Universidade de São Paulo
Instituto de Física

Física Estatística para Compressão e Ocultação de Dados

Antonio André Monteiro Manoel

Orientador: Prof. Renato Vicente

Tese de doutorado apresentada
ao Instituto de Física para a obtenção
do título de Doutor em Ciências

Banca Examinadora:

Prof. Dr. Renato Vicente (IME-USP)

Prof. Dr. Fábio Gagliardi Cozman (Poli-USP)

Prof. Dr. José Fernando Fontanari (IFSC-USP)

Prof. Dr. Márcio Teixeira do Nascimento Varella (IF-USP)

Prof. Dr. Nestor Felipe Caticha Alfonso (IF-USP)

São Paulo
2015

FICHA CATALOGRÁFICA
Preparada pelo Serviço de Biblioteca e Informação
do Instituto de Física da Universidade de São Paulo

Manoel, Antonio André Monteiro

Física Estatística para Compressão e Ocultação de Dados. –
São Paulo, 2015.

Tese (Doutorado) – Universidade de São Paulo.
Instituto de Física. Depto. de Física Geral

Orientador: Prof. Dr. Renato Vicente (IME-USP)

Área de Concentração: Física Estatística

Unitermos: 1. Mecânica Estatística; 2. Inferência Bayesiana;
3. Aprendizado Computacional

USP/IF/SBI-095/2015

University of São Paulo
Institute of Physics

Statistical Physics for Compressed Sensing and Information Hiding

Antonio André Monteiro Manoel

Supervisor: Prof. Renato Vicente

Doctoral thesis presented to
the Institute of Physics for obtaining
the title of Doctor in Science

Examining Committee:

Renato Vicente (IME-USP)

Fábio Gagliardi Cozman (Poli-USP)

José Fernando Fontanari (IFSC-USP)

Márcio Teixeira do Nascimento Varella (IF-USP)

Nestor Felipe Caticha Alfonso (IF-USP)

São Paulo
2015

Agradecimentos

Gostaria primeiro de agradecer todos aqueles que contribuíram diretamente para este trabalho: Renato Vicente, por todos os bons conselhos durante todos estes anos; Florent Krzakala, por gentilmente me acolher em seu grupo de pesquisa em Paris durante um período curto mas bastante agradável e produtivo; e também Eric Tramel e Lenka Zdeborová, com quem tive o prazer de trabalhar neste período. Agradeço ainda a Nestor Caticha, com quem aprendi que todos os modelos estão errados, mas que os modelos bayesianos estão menos errados.

Agradeço a minha família e especialmente à meus pais por todo o apoio, sem o qual esta tese não teria sido possível, e à todos os amigos e colegas de faculdade com quem mantive contato durante todos estes anos – João e Paulo, com quem compartilhei as alegrias e tristezas de fazer um doutorado; Leonardo e Lucas, que tornaram minha estadia em Paris ainda mais aprazível; e Bruno, Gabriel, Henrique, Michel e Petre, por todos os momentos de diversão. Finalmente, agradeço à todos o estudantes, professores e funcionários de meu grupo – em particular Andre, Carol, Cinthia, Diogo, Felipe, Jonatas e Rafael – e de meu departamento, e à todos que me ajudaram de um jeito ou de outro nestes anos, as vezes até mesmo sem perceber.

O auxílio financeiro para este trabalho foi fornecido pela Fundação de Amparo e Pesquisa do Estado de São Paulo (FAPESP) sob os processos no. 2012/12363-8 e no. 2013/01213-8. Auxílio material foi fornecido também pelo Centro para Sistemas Naturais e Artificiais de Processamento de Informações (CNAIPS-USP).

Acknowledgments

I would first like to thank those who directly contributed to this work: Renato Vicente, for all the good advice through all these years; Florent Krzakala, for kindly hosting me at his research group in Paris during some very exciting and productive few months; and also to Eric Tramel and Lenka Zdeborová, with whom I had the pleasure to work with during that time. I would also like to thank Nestor Caticha, who taught me all models are wrong but Bayesian ones are less so.

I thank my family and especially my parents for all the support, without which this work would not have been possible, and all friends and colleagues from university who have kept in touch through all these years – João and Paulo, with whom I have shared the burdens and joys of doing a doctorate; Leonardo and Lucas, who made my stay in Paris even more pleasant; and Bruno, Gabriel, Henrique, Michel and Petre for all the fun times. Finally, I thank all the students, professors and personnel in my group – in particular Andre, Carol, Cinthia, Diogo, Felipe, Jonatas and Rafael – and in my department, and all other individuals who helped me in way or another through the past few years, sometimes without even realizing they were doing so.

Financial support from the São Paulo Research Foundation (FAPESP) under grants 2012/12363-8 and 2013/01213-8 is acknowledged. Material support was also provided by the Center for Natural and Artificial Information Processing Systems (CNAIPS-USP).

Resumo

Esta tese está dividida em duas partes. Na primeira delas, mostramos como problemas de inferência estatística e de otimização combinatória podem ser abordados sob um *framework* unificado que usa ferramentas de áreas tão diversas quanto o aprendizado de máquina, a física estatística e a teoria de informação, permitindo que i) projetemos algoritmos para resolver os problemas, ii) analisemos a performance destes algoritmos tanto empiricamente como analiticamente, e iii) comparemos os resultados obtidos com os limites teóricos. Na segunda parte, este *framework* é usado no estudo de dois problemas específicos, um de inferência (*compressed sensing*) e outro de otimização (ocultação de dados). Em ambos os casos, revisamos abordagens recentes, identificamos suas falhas, e propomos novos esquemas que visam corrigir estas falhas, baseando-nos sobretudo em algoritmos de troca de mensagens, técnicas de inferência variacional, e modelos de vidro de spin da física estatística.

Abstract

This thesis is divided into two parts. In the first part, we show how problems of statistical inference and combinatorial optimization may be approached within a unified framework that employs tools from fields as diverse as machine learning, statistical physics and information theory, allowing us to i) design algorithms to solve the problems, ii) analyze the performance of these algorithms both empirically and analytically, and iii) to compare the results obtained with the optimal achievable ones. In the second part, we use this framework to study two specific problems, one of inference (*compressed sensing*) and the other of optimization (*information hiding*). In both cases, we review current approaches, identify their flaws, and propose new schemes to address these flaws, building on the use of message-passing algorithms, variational inference techniques, and spin glass models from statistical physics.

Notation

Sets, vectors and matrices

$$U \setminus V = \{x \in U \mid x \notin V\}$$

difference between sets U and V

$$U \setminus i = U \setminus \{i\}$$

all elements of U except i

$$[N] = \{0, 1, 2, \dots, N\}$$

set containing all non-negative integers smaller or equal to N

$$\mathbf{x}_U = (x_{i_1}, x_{i_2}, \dots) \text{ for } U = (i_1, i_2, \dots)$$

components of the vector $\mathbf{x}$ which indices are contained in the set U

$$\mathbf{x}_{\setminus i} = (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N)$$

all components of the vector $\mathbf{x}$ except the i -th one

$$\mathbb{1}_N$$

$N \times N$ identity matrix

Graphs

$$\mathcal{G}(V, E)$$

graph consisting in a set of vertices V and a set of edges $E \subseteq V^2$

$$\mathcal{G}(V_N \cup V_F, E)$$

bipartite *factor graph* consisting in a set of variable nodes V_N , a set of factor nodes V_F , $V_N \cap V_F = \emptyset$, and a set of edges $E \subseteq V_N \times V_F$

$$\partial a = \{i \in V_N \mid (i, a) \in E\} = \{i_a^1, i_a^2, \dots\}$$

neighborhood of factor node a , i.e. set containing the indices of the variable nodes connected to a

$$\partial a \setminus i$$

difference between ∂a and $\{i\}$, i.e. set containing all neighbors of a except i

Random variables and probability distributions

$$P(\mathbf{x}; \boldsymbol{\theta})$$

probability density/mass function of $\mathbf{x}$, parametrized by $\boldsymbol{\theta}$ (the parameter may be omitted)

$$\mathbf{x} \sim P(\mathbf{x})$$

$\mathbf{x}$ is a random variable distributed according to $P(\mathbf{x})$

$$P(\mathbf{x}) \propto f(\mathbf{x})$$

$$\mathbf{x}_1 \stackrel{d}{=} \mathbf{x}_2$$

$$\langle f(\mathbf{x}, \mathbf{y}) \rangle_P = \int d\mathbf{x} P(\mathbf{x}|\mathbf{y}) f(\mathbf{x}, \mathbf{y})$$

$$\mathbb{E}_{\mathbf{y}} f(\mathbf{x}, \mathbf{y}) = \int d\mathbf{y} P(\mathbf{y}) f(\mathbf{x}, \mathbf{y})$$

$$\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}x^2}$$

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma^2) = \frac{1}{(2\pi)^{\frac{N}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}\mathbf{x}^T \Sigma^{-1} \mathbf{x}}$$

$$\begin{aligned} B(x; p) &= p\delta(x-1) + (1-p)\delta(x) \\ &= (1-p) \left(\frac{p}{1-p} \right)^x \end{aligned}$$

$$B_{\pm q}(x; p) = p\delta(x-q) + (1-p)\delta(x+q)$$

$$\text{Poisson}(x; \lambda) = \frac{\lambda^x}{x!} e^{-\lambda}$$

$$\mathcal{W}^{-1}(x; \Sigma, M)$$

the probability distribution is defined up to a normalization constant, i.e. $P(\mathbf{x}) = \frac{1}{Z} f(\mathbf{x})$ with $Z = \int d\mathbf{x} f(\mathbf{x})$

equality in distribution, i.e. given $\mathbf{x}_1 \sim P_1(\mathbf{x}_1)$ and $\mathbf{x}_2 \sim P_2(\mathbf{x}_2)$, $P_1(\mathbf{x}) = P_2(\mathbf{x})$ for all $\mathbf{x}$

average of $f(\mathbf{x}, \mathbf{y})$ with respect to $P(\mathbf{x}|\mathbf{y})$

average of $f(\mathbf{x}, \mathbf{y})$ with respect to $P(\mathbf{y})$

normal distribution of mean μ and variance σ^2

multivariate normal distribution of mean $\boldsymbol{\mu} \in \mathbb{R}^N$ and covariance matrix $\Sigma \in \mathbb{R}^{N \times N}$

Bernoulli distribution with parameter p

bimodal distribution with parameter p

Poisson distribution with parameter λ

inverse-Wishart distribution with scale matrix $\Sigma \in \mathbb{R}^{N \times N}$ and M degrees of freedom

The argument of the distribution is omitted if unambiguous, e.g. the normal distribution may be written as $\mathcal{N}(\mu, \sigma^2)$ instead of $\mathcal{N}(x; \mu, \sigma^2)$.

Miscellaneous

$$F[f]$$

$$\mathbb{I}(x = a) = \begin{cases} 1, & \text{if } x = a, \\ 0, & \text{otherwise.} \end{cases}$$

$$x \oplus y = (x + y) \bmod 2$$

functional mapping a given set of functions to $\mathbb{R}$

indicator function – evaluates to 1 if x is equal to a and to 0 otherwise

sum modulo 2 of $x_i, y_i \in \{0, 1\}$

Contents

0	Introduction	1
I	Theory	9
1	Inference and Optimization on Graphical Models	11
1.1	Probabilistic graphical models	12
1.2	Latent variable estimation	17
1.3	Combinatorial optimization	23
2	Variational Inference and Message-Passing Algorithms	29
2.1	Schemes for approximate inference	29
2.2	Inference with continuous variables	37
2.3	Variational optimization	42
3	The Role of Statistical Physics	47
3.1	Disordered systems in equilibrium	49
3.2	Planted ensemble and the Nishimori line	55
3.3	Looking for ground states	58
II	Applications	61
4	Compressed Sensing	63
4.1	Bayesian approach to linear systems	64
4.2	Variational free energies	75
4.3	Swept approximate message-passing	81
5	Information Hiding	91
5.1	Steganographic algorithms	91
5.2	Exploiting LDPC-based quantizers	105
5.3	Insights from statistical physics	110
6	Conclusion	115

*On ne reçoit pas la sagesse, il faut
la découvrir soi-même après un trajet que
personne ne peut faire pour nous,
ne peut nous épargner.¹*

MARCEL PROUST

¹We don't receive wisdom, we must discover it for ourselves after a journey that no one can take for us or spare us.

Introduction

The interdisciplinary aspect of statistical physics has become more and more prominent throughout the past few decades, and is today widely acknowledged in the field. Models that were initially proposed to explain the macroscopic behavior of materials from their microscopic properties are being used today in the study of a wide range of phenomena in the biological[BW87; FLS13], economical[MS99; DY00; BP03] and social sciences[GG82; CFL09; CV11].

In this context, one of the first connections to be established was that between statistical physics and C. Shannon's *mathematical theory of communication*[Sha48], better known today as information theory[CT91]. Shortly after the publication of work that inaugurated the field in 1948, physicist E.T. Jaynes noticed that many results from statistical physics arise naturally in Shannon's framework[Jay57]. In Jaynes' words

If one considers statistical mechanics as a form of statistical inference rather than as a physical theory, it is found that the usual computational rules, starting with the determination of the partition function, are an immediate consequence of the maximum-entropy principle.

The *maximum-entropy principle* proposed by Jaynes makes use of the following quantity

$$S[P] = - \int d\mathbf{x} P(\mathbf{x}) \log P(\mathbf{x}), \quad (0.1)$$

denoted by Shannon the *entropy* of $\mathbf{x} \sim P(\mathbf{x})$, as it has essentially the same form as the Gibbs entropy from statistical physics. In Shannon's framework, the entropy attempts to measure how uncertain is a random variable. Jaynes noticed that maximizing Shannon's entropy while keeping an arbitrary function of $\mathbf{x}$ fixed in average leads to the Gibbs-Boltzmann distribution from statistical physics

$$\frac{\delta}{\delta P} \left\{ S[P] + \lambda \left(\int d\mathbf{x} P(\mathbf{x}) f(\mathbf{x}) - \bar{f} \right) \right\} = 0 \quad \Rightarrow \quad P(\mathbf{x}) = \frac{1}{Z} e^{\lambda f(\mathbf{x})}. \quad (0.2)$$

Thus, any system – physical or otherwise – that is characterized by a quantity that is kept fixed in average is well described by this distribution; in other words, if the only knowledge we have about a system is that a certain quantity is kept fixed in average, then the distribution that makes less hypotheses while taking this knowledge into account is (0.2).

More recently, the relation between these two disciplines has shown to be reciprocal, as techniques from statistical physics – more specifically, techniques used to approach disordered systems, such as the replica and cavity methods – were also shown to be useful in the treatment of problems in information theory[Sou94; VSK02; KS04].

The convergence point happens to be in the specific problems approached by these two fields – generally speaking, most of them may be classified either as problems of *statistical inference*, or of *combinatorial optimization*. As a consequence, the techniques developed in both statistical mechanics and information theory actually allow us to approach problems in other areas that also belong to one of these classes[MM09].

There are other disciplines that focus mainly on problems of inference and optimization, most notably machine learning[Bis07; Bar11; Mur12]. Each discipline however has a different goal in treating these problems – while machine learning is concerned with developing new approaches to solve them, statistical physics seeks to evaluate the average performance of a particular approach, and information theory investigates the optimal performance that can be obtained by any possible approach. Together, the tools developed in each of these fields constitute a toolbox that allows one to perform an exhaustive analysis of inference and optimization problems.

0.1 A first example: channel and source coding

We now present the two main problems in information theory, both introduced by Shannon in his 1948 paper. While one is a problem of inference, the other is a problem of optimization. From these examples, we should be able to see what are the main properties of these classes of problems, and also what are the main differences between them.

0.1.1 Noisy channel coding

In *noisy channel coding*[Sha48; CT91; Mac03], one is interested in transmitting a message $\mathbf{x} \in \{0, 1\}^N$ through a noisy channel that *corrupts* the message, transforming it into a new message $\mathbf{y} \in \{0, 1\}^N$. The channel is modeled by a distribution $Q(\mathbf{y}|\mathbf{x})$, which gives the probability of receiving a message $\mathbf{y}$ given that a message $\mathbf{x}$ has been sent; for instance, in the so-called binary symmetric channel (BSC), each bit of $\mathbf{x}$ has a probability p of being flipped, so that

$$Q(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^N Q(y_i|x_i), \quad Q(y_i|x_i) = \begin{cases} p, & \text{if } y_i \neq x_i, \\ (1-p), & \text{otherwise.} \end{cases} \quad (0.3)$$

In order for the receiver to be able to identify the corrupted bits, some *redundance* needs to be added to the message $\mathbf{x}$; schemes for adding redundance are known as *error-correcting codes*. If M of the N bits of the message are redundant, then all the possible messages are contained in a small subset $\mathcal{C} \subseteq \{0, 1\}^N$, $|\mathcal{C}| = 2^{N-M}$, known as the *codebook*; the *rate* of the code is defined as $R = (N-M)/N$. We assume the message sent was picked uniformly at random from the codebook, so that the *prior* distribution of $\mathbf{x}$ is

$$P_0(\mathbf{x}) = \mathbb{I}(\mathbf{x} \in \mathcal{C}) = \begin{cases} 1, & \text{if } \mathbf{x} \in \mathcal{C}, \\ 0, & \text{otherwise.} \end{cases} \quad (0.4)$$

The task of the receiver is then to *decode* $\mathbf{x}$ from $\mathbf{y}$, i.e. to determine from $\mathbf{y}$ what message $\mathbf{x} \in \mathcal{C}$ is the most likely to have been sent. This can be done by looking at the posterior distribution $P(\mathbf{x}|\mathbf{y})$, which may be written as a function of $Q(\mathbf{y}|\mathbf{x})$ and $P_0(\mathbf{x})$ by means of Bayes' theorem

$$P(\mathbf{x}|\mathbf{y}) \propto P(\mathbf{y}|\mathbf{x}) P_0(\mathbf{x}), \quad (0.5)$$

where $\propto$ indicates equality up to a normalization constant. Given $P(\mathbf{x}|\mathbf{y})$, the receiver may choose $\mathbf{x}$ in different ways, depending on the error probability to be minimized; for instance by setting

$$\hat{\mathbf{x}}_{\text{MAP}}(\mathbf{y}) = \arg \max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y}), \quad (0.6)$$

the block error probability $P(\hat{\mathbf{x}} \neq \mathbf{x})$ is minimized.

Finally, we briefly enunciate Shannon's channel coding theorem. The *capacity* of a noisy channel is the maximum amount of information that can be recovered per bit of information sent through this channel; it is given by supremum of the mutual information of $\mathbf{x}$ and $\mathbf{y}$

$$C = \max_{P(\mathbf{x})} \sum_{\mathbf{x}_0, \mathbf{x}} Q(\mathbf{y}|\mathbf{x}) P(\mathbf{x}_0) \log \frac{Q(\mathbf{y}|\mathbf{x})}{\sum_{\mathbf{x}'} Q(\mathbf{y}|\mathbf{x}') P(\mathbf{x}')}. \quad (0.7)$$

According to the theorem, there must exist a code with $R < C$ which provides negligible block error probability $P(\hat{\mathbf{x}} \neq \mathbf{x}) \rightarrow 0$ as $N \rightarrow \infty$.

0.1.2 Source coding

Source coding[Sha48; CT91; Mac03] is concerned with the task of compressing a binary string $\mathbf{x}_0 \in \{0, 1\}^N$ into a smaller one, $\mathbf{z} \in \{0, 1\}^M$. This can be done either in a *lossy* or a *lossless* way, i.e. with or without loss of information.

Lossless source coding is achieved by mapping more/less frequent strings to shorter/longer sequences respectively, for instance by using the so-called Huffman coding scheme. Define the rate of compression as $R = M/N$; if we assume the binary strings to be compressed were sampled from a distribution $P(\mathbf{x}_0)$, then following Shannon's source coding theorem, lossless compression should always be possible as long as the average rate is *larger* than the entropy $S[P]$.

When the average rate is *smaller* than $S[P]$, the compression is necessarily lossy. In a subsequent work published in 1959, Shannon determined the minimum rate $R_{\min}$ that needs to be used so as to lose no more than a certain amount of information[Sha59] – more precisely, so that the average distortion between the compressed and decompressed vectors is smaller than D

$$\begin{aligned} R_{\min}(D) = \min_{Q(\mathbf{x}|\mathbf{x}_0)} & \sum_{\mathbf{x}_0, \mathbf{x}} Q(\mathbf{x}|\mathbf{x}_0) P(\mathbf{x}_0) \log \frac{Q(\mathbf{x}|\mathbf{x}_0)}{\sum_{\mathbf{x}'} Q(\mathbf{x}|\mathbf{x}') P(\mathbf{x}')} \\ \text{s.t.} & \sum_{\mathbf{x}_0, \mathbf{x}} Q(\mathbf{x}|\mathbf{x}_0) P(\mathbf{x}_0) d(\mathbf{x}_0, \mathbf{x}) < D, \end{aligned} \quad (0.8)$$

known as the *rate-distortion* curve. Alternatively, the inverse of this function gives the minimum distortion that can be achieved if a certain rate R is being used. For instance, if the source is binary symmetric, $P(x_{0i} = +1) = P(x_{0i} = -1) = 1/2$, and the distortion function considered is the Hamming distance, $d(\mathbf{x}_0, \mathbf{x}) = \sum_i \mathbb{I}[x_{0i} \neq x_i]$, then the rate-distortion curve is given by

$$R_{\min}(D) = 1 - H_2(D), \quad (0.9)$$

where $H_2(D) = -D \log_2 D - (1 - D) \log_2 (1 - D)$ is the binary entropy function.

Decompression is accomplished by mapping $\mathbf{z} \in \{0, 1\}^M$ into a subset $\mathcal{C} \subseteq \{0, 1\}^N$. The particular scheme used to compress and decompress, i.e. to map back and forth from $\{0, 1\}^N$ to $\{0, 1\}^M$, is

known as a code, and $\mathcal{C}$ is again labeled the codebook. Optimal compression for a fixed choice of $\mathcal{C}$ is achieved by solving

$$\mathbf{x}^* = \min_{\mathbf{x} \in \mathcal{C}} d(\mathbf{x}_0, \mathbf{x}). \quad (0.10)$$

In summary, noisy channel coding is characterized by an observed quantity $\mathbf{y}$ generated by a stochastic process that takes as input a quantity $\mathbf{x}$ we would like to estimate – it is essentially a problem of inference. Lossy source coding, on the other hand, consists in determining which of the members $\mathbf{x}$ of a set is closest to the element $\mathbf{x}_0$ in hand, given a certain metric – that is a problem of optimization.

Note however there are many similarities between the two problems – in both cases we are looking for a certain member of a set $\mathcal{C}$ (the codebook). The main difference is that, while in channel coding a member of $\mathcal{C}$ has been slightly altered and given to us, in source coding we are given an arbitrary element which is not in principle related to any member of this set (see Figure 0.1)

Figure 0.1: While in channel coding we are given a slight modification of a member of the codebook $\mathcal{C}$ (black dots) and asked to retrieve that member, in lossy source coding, we are given an arbitrary element and asked to find the closest member of the codebook.

0.2 Main results and organization of this thesis

In the first three chapters of this thesis, we introduce the concepts and techniques that will be used to approach the problems we are interested in. By borrowing tools from machine learning, statistical physics and information theory, we construct a unified framework that will allow us to analyze problems of both inference and optimization.

A concept of particular importance that emerged independently in each of these fields is that of exploiting sparse structures – it leads to the belief propagation algorithm in machine learning, to the Bethe approximation in statistical physics, and to sparse graph codes in information theory. This concept is central to this thesis and will be a recurrent theme in our discussions.

In the last two chapters, we use the toolbox we built to approach two specific problems – one of inference, and the other of optimization. The first is the problem of *compressed sensing*, and the second, of *information hiding*.

0.2.1 Compressed sensing

In compressed sensing[CRT04; CW08], we are given a vector $\mathbf{y} \in \mathbb{R}^M$ that is the product of a known matrix $F \in \mathbb{R}^{M \times N}$ with another vector $\mathbf{x} \in \mathbb{R}^N$ which is known to be sparse, i.e. only a fraction ρ of the elements of $\mathbf{x}$ are different from zero. We are then interested in recovering $\mathbf{x}$; this can be easily done for $M \geq N$, however we shall focus on the $M < N$ case. As we will see, the fact that $\mathbf{x}$ is sparse allows us to find a solution even if the system is underdetermined.

The first practical algorithms for determining sparse solutions to underdetermined systems were based on the minimization of the ℓ_1 norm[Tib96; CDS98], i.e. on solving the following convex problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{x}\|_1 = \sum_i |x_i|, \\ \text{s.t.} \quad & \mathbf{y} = F\mathbf{x}. \end{aligned} \tag{0.11}$$

More recently, approaches based on Bayesian statistics and message-passing algorithms such as belief propagation (BP) have been shown to attain better performance[Mon10]. In this context, the most successful algorithm is arguably AMP[DMM09; Ran10; KMS+12] (short for *approximate message-passing*), which is based on a relaxation of BP. However, even though providing much better results than the ℓ_1 minimization under ideal conditions, AMP present serious problems of convergence whenever such conditions are not met.

Figure 0.2: Convergent alternatives to the approximate message-passing algorithm. (a) By directly optimizing the Bethe free energy, we obtain a practical algorithm that provides mean-squared errors similar to AMP's in the base case. However, differently from AMP, it still converges even under non-ideal conditions. (b) While AMP (top left corner) diverges for certain parameter values, the algorithm we propose, SwAMP (top right corner) does not; furthermore, it provides better results than the classical L1 approach (bottom). More details are given in chapter 4.

Our purpose in chapter 4 is to find convergent alternatives to AMP, i.e. algorithms that perform as well as AMP in the cases where it converges, but that also provide good results in the cases where AMP does *not* converge. We propose two different approaches; the first of them is based on methods for variational inference, and has been published in

Krzakala, F., Manoel, A., Tramel, E. W., & Zdeborova, L. (2014, June). *Variational free energies for compressed sensing*. In Information Theory (ISIT), 2014 IEEE International Symposium on (pp. 1499-1503).

The second is based on the fact that sequential update schedules lead to better convergence properties in message-passing algorithms; it resulted in the following publication

Manoel, A., Krzakala, F., Tramel, E. W., & Zdeborova, L. (2015, July). Swept Approximate Message Passing for Sparse Estimation. In Proceedings of The 32nd International Conference on Machine Learning (pp. 1123-1132).

0.2.2 Information hiding

Digital steganography[Cox08; Fri10] is concerned with the task of embedding secret messages into media files in such a way that any modification to the media is imperceptible. As we will see in chapter 5, it is a problem very similar to the one of lossy compression we described in the last section; it is also a problem of combinatorial optimization.

Embedding a message of M bits into a media file of N necessarily requires us to modify the media in some way; if the distortion we cause is measured using the Hamming distance, then the minimum possible distortion is given by the same rate-distortion theory proposed by Shannon in the case of lossy compression[Cra98; Fri10]. The thick line in Figure 0.3 separates the feasible region from the non-feasible one – i.e., any algorithm that embeds a message of relative size $\alpha = M/N$ must lead to a distortion D above this curve.

Figure 0.3: Average fidelity as a function of the relative message size $\alpha = M/N$ and the distortion D . Classically, only (α, D) pairs above the thick line are achievable; using our scheme, however, one is able to obtain pairs below the line, provided that small embedding errors are allowed (i.e., $f < 1$). More details are given in chapter 5.

We propose a scheme that makes it possible to hide messages while causing distortions *below the curve*, provided that one is willing to allow for errors in the embedding; that is similar to what happens in channel coding, where one can communicate below capacity if a certain error rate can be tolerated. A practical implementation of this scheme using message-passing algorithms is given; we analyze it both empirically and using tools from statistical physics. The results are available in following publication

Manoel, A., & Vicente, R. (2015). *Statistical Physics for the Design and Analysis of Information Hiding Algorithms*, preprint¹.

Two other publications were authored or co-authored by me during my doctorate years. Lecture notes on approximate inference, based on a course given by Manfred Oppel, became a book chapter

Raymond, J., Manoel, A., & Oppel, M. (2015). *Expectation Propagation*. In *Statistical Physics, Optimization, Inference, and Message-Passing Algorithms*. Oxford University Press.

on which parts of the chapter 2 of this thesis are based. Furthermore, the following article, based on my Master's work, inspired the example in subsection 2.1.2

Manoel, A., & Vicente, R. (2013). *Statistical mechanics of reputation systems in autonomous networks*. *Journal of Statistical Mechanics: Theory and Experiment*, 2013(08), P08002.

¹Note of the author: as the time of writing, this work is still not available online.

Part I

Theory

Inference and Optimization on Graphical Models

This first chapter introduces our main object of study throughout this thesis – factorized probability distributions – by presenting mathematical structures that are useful in their description and techniques that enable us to compute many of their properties. Furthermore, we will show how these computations are pertinent to problems in a few different fields.

We will be interested in what follows in computing properties of probability distributions factored as

$$P(\mathbf{x}) = \frac{1}{\mathcal{Z}} \prod_{a=1}^M \psi_a(\mathbf{x}_{\partial a}) \quad (1.1)$$

such that each factor ψ_a depends on a subset $\partial a \subseteq \{1, \dots, N\}$ of the components of $\mathbf{x} \in \chi^N$. The set χ over which the components take their values is known as the *alphabet*, and we will typically be working with either $\chi = \{-1, +1\}$ or $\chi = \mathbb{R}$; the cardinality of the alphabet will be denoted by $q \equiv |\chi|$. Among the properties we want to compute are

1. the mode of $P(\mathbf{x})$, $\mathbf{x}^* = \arg \max_{\mathbf{x}} P(\mathbf{x})$;
2. its marginals $P(x_i) = \int d\mathbf{x}_{\setminus i} P(\mathbf{x})$ and moments $\langle x_i^n \rangle = \int d\mathbf{x} x_i^n P(\mathbf{x})$ (or cumulants);
3. and the normalization factor $\mathcal{Z} = \int d\mathbf{x} \prod_{a=1}^M \psi_a(\mathbf{x}_{\partial a})$.

Each of these properties is related to a particular task in different areas of knowledge. The mode, for instance, provides the MAP and ML estimates to statisticians, the optimal solution to computer scientists and the ground state of the system to physicists. The marginals and moments allow one to obtain the MMSE estimate to an inference problem, and also the thermodynamic properties of a physical system; and the normalization factor $\mathcal{Z}$ gives the evidence of a Bayesian model, as well as the cumulant-generating function (or free energy) whenever the distribution belongs to the exponential family (see Box 1.1). All these concepts will be discussed in detail below.

Box 1.1: Exponential family of distributions

The probability distributions we will be approaching throughout this thesis are not only factored as (1.1), but also usually belong to the *exponential family*. A distribution is said to belong to the exponential family if it has the form[WJ08]

$$P(\mathbf{x}) = h(\mathbf{x}) \exp \{ \boldsymbol{\eta}(\boldsymbol{\theta}) \cdot \boldsymbol{\phi}(\mathbf{x}) - A(\boldsymbol{\theta}) \}, \quad (1.2)$$

with $\boldsymbol{\eta}(\boldsymbol{\theta})$ and $\boldsymbol{\phi}(\mathbf{x})$ vector-valued functions with the same dimension. To make this expression more like (1.1), we may rewrite it as

$$P(\mathbf{x}) = \frac{1}{\mathcal{Z}(\boldsymbol{\theta})} \prod_{a=1}^M \exp \{ \eta_a(\boldsymbol{\theta}) \phi_a(\mathbf{x}) + \log h(\mathbf{x}) \} \quad (1.3)$$

with $\mathcal{Z}(\boldsymbol{\theta}) = e^{A(\boldsymbol{\theta})}$. Some examples are

- the univariate normal distribution of mean μ and variance σ^2 , for $x \in \mathbb{R}$, $\boldsymbol{\theta} = (\mu, \sigma^2)$ and

$$\begin{aligned} h(x) &= \frac{1}{\sqrt{2\pi}} & \phi(x) &= \left(x, -\frac{x^2}{2}\right) \\ \eta(\mu, \sigma) &= \left(\frac{\mu}{\sigma^2}, \frac{1}{\sigma^2}\right) & A(\mu, \sigma^2) &= \frac{1}{2} \log(\sigma^2) + \frac{\mu^2}{\sigma^2} \end{aligned}$$

- the Bernoulli distribution of parameter p , for $x \in \{0, 1\}$, $\theta = p$ and

$$\begin{aligned} h(x) &= 1 & \phi(x) &= x \\ \eta(p) &= \log \frac{p}{1-p} & A(p) &= -\log(1-p) \end{aligned}$$

- the Poisson distribution of parameter λ , for $x \in \mathbb{N}$, $\theta = \lambda$ and

$$\begin{aligned} h(x) &= \frac{1}{x!} & \phi(x) &= x \\ \eta(\lambda) &= \log \lambda & A(\lambda) &= \lambda \end{aligned}$$

The set of functions $\{\phi_a(\mathbf{x})\}$ completely characterizes the sample distribution and is thus known as the set of *sufficient statistics*. Notice that $A(\boldsymbol{\eta})$ – obtained by writing $\boldsymbol{\theta}$ in terms of $\boldsymbol{\eta}(\boldsymbol{\theta})$ – serves as generating function for the cumulants $\{\kappa_m^a\}$ of $\phi_a(\mathbf{x})$, since $\frac{\partial^m A(\boldsymbol{\eta})}{\partial \eta_a^m} = \kappa_m^a$; in particular, $\frac{\partial A(\boldsymbol{\eta})}{\partial \eta_a} = \langle \phi_a \rangle$ and $\frac{\partial^2 A(\boldsymbol{\eta})}{\partial \eta_a \partial \eta_b} = \langle \phi_a \phi_b \rangle - \langle \phi_a \rangle \langle \phi_b \rangle$. Moreover, distributions of the exponential family maximize the Shannon entropy

$$S[P] = - \int d\mathbf{x} P(\mathbf{x}) \log P(\mathbf{x}), \quad (1.4)$$

subject to constraints $\langle \phi_a \rangle = \mu_a$. In particular, the Gibbs-Boltzmann distribution,

$$P(\mathbf{x}|\beta, \mathbf{K}) = \frac{1}{\mathcal{Z}} \exp \{ -\beta \mathcal{H}_{\mathbf{K}}(\mathbf{x}) \},$$

also follows from (1.2), for

$$\begin{aligned} h(\mathbf{x}) &= 1 & \mathbf{K} \cdot \phi(\mathbf{x}) &= \mathcal{H}_{\mathbf{K}}(\mathbf{x}) \\ \eta(\beta, \mathbf{K}) &= -\beta \mathbf{K} & A(\beta, \mathbf{K}) &= \log \mathcal{Z} \end{aligned}$$

Therefore, it could be said that computing the macroscopic properties of a physical system is actually nothing more than performing inference over a distribution of the exponential family – in fact, the cumulant-generating function $A(\boldsymbol{\eta})$ is nothing more than the free energy of the system (apart from a factor $-\frac{1}{\beta}$). On the other hand, many techniques originally developed in the context of statistical physics have recently proved to be useful in the study of inference problems.

1.1 Probabilistic graphical models

Factorized distributions such as (1.1) have been studied in the past under the name of *Markov random fields* by physicists and statisticians, and of *Bayesian networks* by engineers and computer scientists. More recently, the language of probabilistic graphical models [Bis07; KF09; Mur12; Jor04] began to provide a unified perspective on the subject by borrowing tools from these different fields.

Graphical models provide a visual representation of the structure of dependency among a set

Figure 1.1: Factor graph for distribution (1.5). Each square *factor node* represents a factor $\psi_a(\mathbf{x}_{\partial a})$ of the distribution, while circular *variable nodes* are associated to the components of $\mathbf{x}$. An edge is drawn between a and i if $i \in \partial a$, that is, if x_i is among the arguments of compatibility function $\psi_a(\mathbf{x}_{\partial a})$.

of random variables; since the joint distribution of these variables completely characterizes such dependencies, graphical models may also be seen as a way of representing the joint distribution itself. Three different descriptions are usually employed, the main difference between them being whether the graph used is undirected, directed or bipartite[Bar11] – the last of these, known as the *factor graph* representation, will be better suited for our needs.

A factor graph[KFL06] is a bipartite graph $\mathcal{G} = (V, E)$ containing two types of nodes: variable nodes $v \in V_N$ and factor nodes $f \in V_F$, with $V_N \cup V_F = V$ and $V_N \cap V_F = \emptyset$. The set of edges $E \subset \{(v, f) \mid v \in V_N \wedge f \in V_F\}$ connect only nodes in V_N to nodes in V_F . Each node $v \in V_N$ is associated to a component x_v of $\mathbf{x}$, while each node $f \in V_F$ is associated to a factor $\psi_f(\mathbf{x}_{\partial f})$ of $P(\mathbf{x})$; edges connect each of the factor nodes f to the variable nodes ∂f associated to the arguments of ψ_f , so that $E = \{(v, f) \mid f \in V_F \wedge v \in \partial f\}$. Take for instance the following distribution

$$P(x_1, x_2, x_3, x_4) = \frac{1}{Z} \psi_1(x_1, x_2) \psi_2(x_2, x_4) \psi_3(x_2, x_3, x_4) \quad (1.5)$$

for which $V_N = \{i_1, i_2, i_3, i_4\}$ (associated respectively to x_1, x_2, x_3 and x_4) and $V_F = \{a_1, a_2, a_3\}$ (associated to ψ_1, ψ_2 and ψ_3) – the corresponding factor graph is drawn in Figure 1.1. A distinctive feature of factor graphs is that each graph uniquely determines a distribution and vice-versa.

We are interested in designing algorithms that calculate properties of a probability distribution by operating over the associated factor graph. In fact, as we will see, some algorithms are capable of exploiting graphical models with a *sparse structure*, reducing the computational complexity of these calculations.

1.1.1 Exploiting sparse graphical models

The complexity of computing the many properties of a distribution, in particular those we listed in the beginning of this chapter, is directly linked to the structure of the factor graph that represents it[Jor04]. In particular, if the factor graph is sparse – meaning both $|\partial a|$ and $|\partial i|$ are typically $o(1)$ – then we should be able to accomplish these computations in time polynomial in N and $q \equiv |\chi|$; on the other hand, if the graph is complete and interactions are weak, good approximations are obtained by means of the law of large numbers as $N \rightarrow \infty$. In the general case, however, the computations are NP-hard¹.

¹More precisely, since computing the partition function is regarded as a counting problem, it is generally #P-hard – a class of problems considered to be even more difficult than NP-hard[MM11].

Figure 1.2: The factor graph for distribution (1.6) is a tree, since it does not contain cycles.

Let us consider the distribution given by the factor graph in Figure 1.2

$$P(\mathbf{x}) = \frac{1}{\mathcal{Z}} \psi_1(x_0, x_1, x_2, x_3, x_4) \psi_2(x_0, x_5, x_6, x_7) \psi_3(x_7, x_8, x_9) \quad (1.6)$$

and focus on the task of computing the normalization factor (or partition function)

$$\mathcal{Z} = \sum_{\mathbf{x}} P(x_0, \dots, x_9).$$

Assuming each of the variables x_i takes values over an alphabet χ – so that $\mathbf{x} \in \chi^{10}$ – a naïve, brute-force calculation would consist in computing each of the $|\chi|^{10}$ values of $P(\mathbf{x})$ and summing them together. However, by cleverly rearranging the terms of the sum one could write it as

$$\begin{aligned} \mathcal{Z} &= \sum_{x_0} \underbrace{\sum_{x_1, x_2, x_3, x_4} \psi_1(x_0, x_1, x_2, x_3, x_4)}_{m_{a_1 \rightarrow i_0}(x_0)} \sum_{x_5, x_6, x_7} \psi_2(x_0, x_5, x_6, x_7) \underbrace{\sum_{x_8, x_9} \psi_3(x_7, x_8, x_9)}_{m_{a_3 \rightarrow i_7}(x_7)} \\ &= \sum_{x_0} m_{a_1 \rightarrow i_0}(x_0) \underbrace{\sum_{x_5, x_6, x_7} \psi_2(x_0, x_5, x_6, x_7) m_{a_3 \rightarrow i_7}(x_7)}_{m_{a_2 \rightarrow i_0}(x_0)} \\ &= \sum_{x_0} m_{a_1 \rightarrow i_0}(x_0) m_{a_2 \rightarrow i_0}(x_0), \end{aligned} \quad (1.7)$$

where each of the functions $m_{a \rightarrow i_0}$ are obtained respectively in $|\chi|^4$ and $|\chi|^3 + |\chi|^2$ steps, so that the whole calculation takes $|\chi|^4 + |\chi|^3 + |\chi|^2 + |\chi|$ steps; for $|\chi| = 2$, 30 steps would be required in total, while the naïve calculation would take 1024. Analogously, the marginal distribution of x_0 can be obtained from

$$P(x_0) = \sum_{\mathbf{x} \sim 0} P(x_0, \dots, x_9) = \frac{1}{\mathcal{Z}} m_{a_1 \rightarrow i_0}(x_0) m_{a_2 \rightarrow i_0}(x_0). \quad (1.8)$$

The reason why terms may be rearranged as we did in (1.7) is that the factor graph in Figure 1.2 is a *tree*, meaning there is a single path connecting any two vertices; equivalently, there are no *cycles* in the graph – no path that begins and ends in the same vertex.

This rearrangement procedure is known as *elimination*, and may only be employed when the factor graph is a tree. When it is not, a more complicated but similar procedure known as the *junction tree* algorithm[Bar11] can be used. The basic idea is to perform a *tree decomposition* of the original factor graph, by grouping nodes together and linking groups of nodes that have non-zero intersection; then the elimination algorithm may be used in this new graph, while brute-force has to be employed inside each group of nodes. The junction tree algorithm runs however in exponential time in the *treewidth* of a graph, defined as the size of the largest group of nodes in its tree decomposition.

Since for general graphs exact inference is often not feasible, we would like to devise a scheme that runs in polynomial time in the number of nodes while providing approximations to the marginals and to the partition function that are good enough for our purposes.

Figure 1.3: As an additional factor node a_4 is added and connected to variable nodes i_1 and i_9 , a cycle appears, and the factor graph is no longer a tree. Arrows in red represent some of the messages being exchanged between factor and variable nodes, which following (1.9) are computed as $m_{i_1 \rightarrow a_1}(x_1) = \prod_{a \in \partial i_1 \setminus a_1} m_{a \rightarrow i_1} = m_{a_4 \rightarrow i_1}$ and $m_{a_1 \rightarrow i_0}(x_0) = \sum_{\mathbf{x}_{\partial a_1 \setminus i_0}} \psi_{a_1}(\mathbf{x}_{\partial a_1}) \prod_{i \in \partial a_1 \setminus i_0} m_{i \rightarrow a_1}(x_i) = \sum_{x_1, \dots, x_4} \psi_{a_1}(x_0, x_1, \dots, x_4) m_{i_1 \rightarrow a_1}(x_1) \dots m_{i_4 \rightarrow a_1}(x_4)$.

Approximate inference: belief propagation

There are situations where, even though the factor graph is not a tree, it is sparse enough for us to obtain approximations $\hat{\mu}_i(x_i)$ to the marginals by repeatedly applying local elimination steps

$$m_{a \rightarrow i}^{(t+1)}(x_i) \propto \sum_{\mathbf{x}_{\partial a \setminus i}} \psi_a(\mathbf{x}_{\partial a}) \prod_{j \in \partial a \setminus i} \underbrace{\left[\prod_{b \in \partial j \setminus a} m_{b \rightarrow j}^{(t)}(x_j) \right]}_{m_{j \rightarrow a}^{(t)}(x_j)}. \quad (1.9)$$

This set of equations is supposed to be iterated until convergence is eventually reached, at which point the approximated marginals are computed from $\hat{\mu}_i(x_i) = \prod_{a \in \partial i} m_{a \rightarrow i}^*(x_i)$. If the cycles are few

and typically long, the graph is said to be locally tree-like; in this case, one usually verifies empirically that the iteration not only converges but also provides good approximations to the marginals and to the partition function.

This iterative procedure may be seen as a *message-passing* algorithm over the correspondent factor graph, where variable nodes repeatedly send messages $m_{i \rightarrow a}$ to the factor nodes, which are then used to compute messages $m_{a \rightarrow i}$ that will be sent in the opposite direction (see Figure 1.3). The arrow notation is employed with the specific purpose of emphasizing this aspect of the procedure.

Box 1.2: Bethe-Peierls(-Weiss) equation

As an example, let us apply recursion (1.9) to an Ising-like model, with binary variables $x \in \{-1, +1\}^N$ interacting in pairs over a c -regular graph $\mathcal{G} = (V, E)$, i.e. so that each variable interacts with c others

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{(i,j) \in E} e^{\beta J x_i x_j}. \quad (1.10)$$

Each factor is indexed by an unordered pair (ij) , with $\psi_{ij}(x_i, x_j) = e^{\beta J x_i x_j}$. By defining $h_{i \rightarrow j} = \frac{1}{2\beta} \log \frac{m_{i \rightarrow (ij)}(+1)}{m_{i \rightarrow (ij)}(-1)}$, we get

$$h_{i \rightarrow j}^{(t+1)} = \frac{1}{\beta} \sum_{k \in \partial i \setminus j} \tanh^{-1} \left[\tanh(\beta J) \tanh(\beta h_{k \rightarrow i}^{(t)}) \right]. \quad (1.11)$$

Let us assume furthermore that $h_{i \rightarrow j} \equiv h_1$ for every pair (i, j) ; then the fixed points of the iteration are given by

$$\tanh(\beta h_1) = (c - 1) \tanh(\beta J) \tanh(\beta h_1), \quad (1.12)$$

which is essentially the same equation written in 1948 by P.R. Weiss^a, who studied the ferromagnetic Ising model using Bethe and Peierls ideas[Wei48; Bet35]; h_1 in this case acts as the effective field over neighbouring spins.

^aNot to be confused with Pierre Weiss (1865-1940), of *Curie-Weiss equation* fame

Recursions similar in spirit to (1.9) have been used many times in the past. In a work published in 1935 by Hans Bethe[Bet35], a specific case of this recursion is studied so as to compute properties of a model for disordered alloys. Later, in the 1960s, an analogous procedure was devised by information-theorist Robert Gallager in order to decode low-density error-correcting codes in linear time[Gal62], but the limited computational power available at the time did not allow him to explore the idea in its full potential. The procedure was better formalized in the context of artificial intelligence by Judea Pearl, who in 1982 gave the algorithm the name by which it is most widely known today, *belief propagation*[Pea82], or BP. Only in the 2000s however came the realization that this same principle is employed in a number of methods across many different areas[AM00]. The idea of using it on graphs with cycles is also quite recent, and gave rise to the *loopy belief propagation* (LBP) algorithm[MWJ99]. Finally, an alternative variational derivation was provided by Jonathan Yedidia in 2000, once again connecting the procedure to mean-field theories of statistical physics[YFW00].

This variational derivation will be of great importance in what follows, as it allows one to better understand the powers and limitations of BP. One could ask for instance whether iteration (1.9) is guaranteed to converge on sparse graphs; it is known today that this is not the case, as its fixed-points become locally unstable at certain regions of the parameters space[MM09]. Even more interesting

is the fact that this instability is not only related to properties of the algorithm, but to properties of the probability distribution itself. Other questions that arise are: if there is a fixed-point, is it unique? If there are many fixed-points, does every one of them provide good approximations to the marginals? These and other questions are central to this thesis and will be further explored in the next chapters.

We will next present two classes of problems that greatly benefit from being described in terms of graphical models – problems of inference (section 1.2) and of optimization (section 1.3).

1.2 Latent variable estimation

In statistical inference and more specifically estimation theory [Was04], one is interested in estimating parameters θ or hidden (*latent*) variables x given a model relating them to a set of observed variables y . In a Bayesian setting [Jay03; SS06], this model can be either generative or discriminative [BL07]; in the former case, the joint distribution $P(x, y|\theta)$ is assumed to be known, while in the latter only the conditional distribution $P(y|x, \theta)$ is given. We will here focus on generative models, for which the Bayes theorem may be used to write the *posterior distribution* $P(x|y, \theta)$ in terms of the given joint one

$$P(x|y, \theta) = \frac{P(x, y|\theta)}{P(y|\theta)} = \frac{1}{Z_\theta(y)} P(y|x, \theta_\ell) P_0(x|\theta_p), \quad (1.13)$$

with $\theta = \{\theta_\ell, \theta_p\}$. The distribution $P(y|x, \theta_\ell)$ goes by the name of *likelihood*, and its purpose is to enforce the relationship assumed to exist between x and y ; the *prior distribution* $P(x|\theta_p)$, on the other hand, reflects any prior knowledge we have on the structure of x . Finally, $Z_\theta(y) \equiv P(y|\theta) = \int dx P(x, y|\theta)$ gives the marginal likelihood or *evidence*, and allows one to compare the empirical distribution of the data with the one provided by the model.

In this setting, we may also write the posterior distribution of the parameters θ by marginalizing over the latent variables

$$P(\theta|y) \propto \int dx P(x, y|\theta) P_0(\theta). \quad (1.14)$$

Figure 1.4: Sketch of generative model (1.16); $x^{(0)}$ is sampled from prior distribution $P_0(x)$ (left) and gives rise to likelihood $P(y|x^{(0)})$ (right). This model is known as *factor analysis* in the statistical inference literature, and is also related to probabilistic principal component analysis (PPCA). Based on figures 12.1 of [Mur12] and 12.9 of [Bis07].

Generative models are also well described by a set of equations of the following form

$$\begin{cases} \mathbf{y} = f(\mathbf{x}, \boldsymbol{\xi}), \\ \boldsymbol{\xi} \sim P(\boldsymbol{\xi}|\boldsymbol{\theta}_\ell), \\ \mathbf{x} \sim P_0(\mathbf{x}|\boldsymbol{\theta}_p). \end{cases} \quad (1.15)$$

This alternative description better illustrates how measurements $\mathbf{y}$ are generated by a process that depends on a set of latent variables $\mathbf{x}$ being at the same time affected by some noise $\boldsymbol{\xi}$, the distribution of which ultimately determines the likelihood $P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_\ell)$. In this context, it is common to name models such as (1.15) as *latent variable models*[Bis07; Bar11; Mur12], or LVMs. We could moreover put a prior $P(\boldsymbol{\theta}|\boldsymbol{\alpha})$ over the parameters, in which case the model is said to be *hierarchical*, and the parameters $\boldsymbol{\alpha}$ of this distribution are known as hyperparameters. Take for instance the following LVM

$$\begin{cases} \mathbf{y} = \mathbf{a} + \mathbf{b}x + \boldsymbol{\xi}, \\ \boldsymbol{\xi} \sim \mathcal{N}(0, \Delta \mathbb{I}_2), \\ x \sim \mathcal{N}(\mu, \sigma^2). \end{cases} \quad (1.16)$$

where $\mathbf{y} \in \mathbb{R}^2$ is the observed variable, $x \in \mathbb{R}$ is the latent one and $\mathbf{a}, \mathbf{b} \in \mathbb{R}^2$ are given. Whenever we gather a sample $\mathbf{y}^{(0)}$, we assume there is a $x^{(0)} \sim P_0(x)$ that generated that sample according to the model above (see Figure 1.4). It may be the case that the model is incorrect and does not reflect the *true* generative process, or that the model is correct but the vector of parameters $\boldsymbol{\theta} = (\mu, \sigma^2, \Delta)$ is unknown; luckily, there are ways to circumvent these issues. Given a sample $\mathbf{y}$, an estimate to the parameters is obtained by maximizing the evidence

$$\hat{\boldsymbol{\theta}}(\mathbf{y}) = \arg \max_{\boldsymbol{\theta}} \log \mathcal{Z}_{\boldsymbol{\theta}}(\mathbf{y}), \quad (1.17)$$

where the logarithm has been introduced for convenience; this procedure is known as *type-II maximum likelihood estimation* or *empirical Bayes*[Bar11]. Furthermore, the accuracy of different models $m \in \mathcal{M}$ may be tested by looking at

$$P(\mathbf{y}|m) = \int d\boldsymbol{\theta} \mathcal{Z}_{\boldsymbol{\theta}}(\mathbf{y}) P(\boldsymbol{\theta}|m), \quad (1.18)$$

which is maximized by the model that better explains the data; an approximation to the expression above is given by the *Bayesian information criterion*[Mur12].

Box 1.3: Expectation-Maximization (EM) algorithm

Since $\mathcal{Z}_{\boldsymbol{\theta}}(\mathbf{y}) = \int d\mathbf{x} P(\mathbf{x}, \mathbf{y}|\boldsymbol{\theta})$ is often hard to compute, it might not be straightforward to obtain the type-II estimate (1.17). One way of dealing with this is by iterating the following procedure starting from some initial guess $\boldsymbol{\theta}^{(0)}$

$$Q^{(t)} = P(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}^{(t)}), \quad (1.19)$$

$$\boldsymbol{\theta}^{(t+1)} = \arg \max_{\boldsymbol{\theta}} \langle \log P(\mathbf{x}, \mathbf{y}|\boldsymbol{\theta}) \rangle_{Q^{(t)}}, \quad (1.20)$$

which is guaranteed to monotonically increase $\log \mathcal{Z}_{\theta^{(t)}}$ at each iteration until a local extremum is reached. The *maximization* (M) step (1.20) guarantees that $f(\theta^{(t+1)}, \theta^{(t)}) = -\text{KL} [P(\mathbf{x}|\mathbf{y}, \theta^{(t+1)}) \| P(\mathbf{x}, \mathbf{y}|\theta^{(t)})]$ will be maximized, while the *expectation* (E) step (1.19) ensures that $f(\theta^{(t)}, \theta^{(t)}) = \log \mathcal{Z}_{\theta^{(t)}}$, so that

$$\log \mathcal{Z}_{\theta^{(t+1)}} \geq \log \mathcal{Z}_{\theta^{(t)}}. \quad (1.21)$$

This procedure goes by the name of expectation-maximization[Bis07; Bar11; Mur12].

Often we are not interested in the posterior distribution itself, but on *point estimates*[Was04] – that is, a single representative value $\hat{\mathbf{x}}$ or $\hat{\boldsymbol{\theta}}$, ideally accompanied by a confidence interval. For instance in (1.16) we would like to estimate $x^{(0)}$ after observing $\mathbf{y}^{(0)}$. This value may be chosen in different ways, depending on our purpose; more precisely, we may use different estimators depending on the loss function we seek to minimize. In Bayesian theory, in particular, it is the average of a loss function with respect to the posterior distribution that is minimized. Among the most frequently employed estimators are[Mur12]

- the maximum a posteriori (MAP) or word MAP estimator

$$\hat{\mathbf{x}}(\mathbf{y}) = \arg \max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}); \quad (1.22)$$

which minimizes the posterior expected value of the 0-1 loss function, $\mathbb{E}_{\mathbf{x}|\mathbf{y}} \mathbb{I}(\hat{\mathbf{x}} \neq \mathbf{x})$, also known as block error probability. Alternatively, one could minimize the 0-1 loss averaged over the components of $\mathbf{x}$ – the bit error probability $\mathbb{E}_{\mathbf{x}|\mathbf{y}} \frac{1}{N} \sum_i \mathbb{I}(\hat{x}_i \neq x_i)$ – by using the maximum posterior marginal (MPM) or symbol MAP estimator

$$\hat{x}_i(\mathbf{y}) = \arg \max_{x_i} P(x_i|\mathbf{y}, \boldsymbol{\theta}); \quad (1.23)$$

- the minimum mean-squared error (MMSE)

$$\hat{\mathbf{x}}(\mathbf{y}) = \arg \min_{\mathbf{x}'} \mathbb{E}_{\mathbf{x}|\mathbf{y}} \|\mathbf{x}' - \mathbf{x}\|^2 = \mathbb{E}_{\mathbf{x}|\mathbf{y}} \mathbf{x}; \quad (1.24)$$

- the maximum likelihood (ML)

$$\hat{\boldsymbol{\theta}}(\mathbf{y}) = \arg \max_{\boldsymbol{\theta}} P(\mathbf{y}|\boldsymbol{\theta}). \quad (1.25)$$

Notice that the computation of most estimators rely on a few properties of the distributions involved, essentially the same properties we listed in the beginning of this chapter. Figure 1.5 gives an idea of the difference between the MAP and MMSE estimates for a posterior distribution that is the mixture of two Gaussians, i.e.

$$P(\mathbf{x}|\boldsymbol{\theta}) = (1 - \rho) \mathcal{N}(\mathbf{x}; 0, 1) + \rho \mathcal{N}(\mathbf{x}; \mu, \sigma^2), \quad (1.26)$$

with $\boldsymbol{\theta} = \{\rho, \mu, \sigma\}$ and ρ small; when sampling from the posterior, one is far more likely to obtain values close to $\hat{\mathbf{x}}_{\text{MMSE}} = \rho\mu \approx 0$ than to $\hat{\mathbf{x}}_{\text{MAP}} = \mu$.

Figure 1.5: MAP and MMSE estimates given a posterior distribution that is the mixture of two Gaussians, (1.26). While the MMSE estimate is very close to the mean of the wider Gaussian, the MAP estimate is given by the mean of the narrow one.

Box 1.4: Bayesian derivation of classical estimation methods

Classical methods for parameter fitting may in general be obtained from generative models with a uniform prior[SS06]. In linear regression, for instance, we are given a vector $\mathbf{Y} \in \mathbb{R}^N$ of observed variables and a set of explanatory variables $\mathbf{X} = [x_1 x_2 \cdots x_N]^T$, $\mathbf{X} \in \mathbb{R}^{N \times P}$, related to each other by means of a vector of parameters $\boldsymbol{\theta} \in \mathbb{R}^P$

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\theta}, \quad (1.27)$$

and we would like to estimate $\boldsymbol{\theta}$ so as to minimize some loss function, a frequent choice being the least-squares estimator (LSE), $\boldsymbol{\theta} = \arg \min_{\boldsymbol{\theta}} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\theta}\|_2^2$. The same results are obtained however by employing the MAP estimator to the following LVM

$$\begin{cases} \mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\eta}, \\ \boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_N), \\ \boldsymbol{\theta} \sim \text{Uniform}(\cdot). \end{cases} \quad (1.28)$$

Notice that in this case $\mathbf{X}$ is given, implying the model does not have hidden or latent variables.

It is often reasonable to assume that each of the measurements are statistically independent of each other, so that $P(\mathbf{y}|\mathbf{x}) = \prod_{\mu=1}^M P(y_{\mu}|\mathbf{x})$. Furthermore, in many cases of interest the components of $\mathbf{x}$ are also independent, $P_0(\mathbf{x}) = \prod_{i=1}^N P_0(x_i)$, so that the posterior distribution may be written as

$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{\mathcal{Z}(\mathbf{y})} \prod_{\mu=1}^M P(y_{\mu}|\mathbf{x}) \prod_{i=1}^N P_0(x_i), \quad (1.29)$$

where we have made implicit the dependence of the different factors on the vector of parameters $\boldsymbol{\theta}$. Given that the posterior distribution has now a form similar to (1.1), its factor graph representation may be used, and computing estimates is now a matter of running algorithms over this factor graph.

1.2.1 Examples

We proceed with some concrete examples of inference problems.

Figure 1.6: Examples of inference problems: (a) binary perceptron, (b) binary symmetric channel, (c) reputation systems.

Example 1: binary perceptron

Given a training set consisting of inputs $\mathbf{X} = [\mathbf{x}^{(1)} \mathbf{x}^{(2)} \dots \mathbf{x}^{(P)}]$, $\mathbf{x} \in \mathbb{R}^N$, and outputs $\boldsymbol{\sigma} \in \{-1, +1\}^P$, one may want to *learn* the rule for generating the outputs so as to correctly predict them for new inputs; that is the main task of a neural network[Bis96; Ami92]. In particular, a perceptron assumes the rule has the following form (see Figure 1.6a)

$$\sigma_i = \text{sgn}(\boldsymbol{\omega} \cdot \mathbf{x}^{(i)}), \quad i = 1, \dots, P, \quad (1.30)$$

and $\boldsymbol{\omega} \in \mathbb{R}^N$ is known as the weight vector. We may speculate that the output is not always generated like this, and that sometimes mistakes occur, specially if $\boldsymbol{\omega} \cdot \mathbf{x} \approx 0$; we will thus include a noise term inside the sign function. By further assigning Gaussian priors to the weights, the following LVM is obtained

$$\begin{cases} \sigma_i = \text{sgn}(\boldsymbol{\omega} \cdot \mathbf{x}^{(i)} + \xi_i), & i = 1, \dots, P, \\ \xi_i \sim \mathcal{N}(0, \Delta), \\ \boldsymbol{\omega} \sim \mathcal{N}(0, \sigma^2 \mathbb{1}_P), \end{cases} \quad (1.31)$$

where we have assumed the weights to be independent random variables. The noise term ξ leads to a probit likelihood

$$\begin{aligned} P(\boldsymbol{\omega} | \mathbf{X}, \boldsymbol{\sigma}) &\propto P(\boldsymbol{\sigma} | \boldsymbol{\omega}, \mathbf{X}) P_0(\boldsymbol{\omega}) = \prod_{i=1}^P P(\sigma_i | \boldsymbol{\omega}, \mathbf{x}^{(i)}) \prod_{\mu=1}^N P_0(\omega_\mu) \\ &= \prod_{i=1}^P \frac{1}{2} \text{erfc} \left(-\sigma_i \frac{\boldsymbol{\omega} \cdot \mathbf{x}^{(i)}}{\sqrt{2\Delta}} \right) \prod_{\mu=1}^N e^{-\frac{\omega_\mu^2}{2\sigma^2}}. \end{aligned} \quad (1.32)$$

Given posterior (1.32), one may then compute different estimates $\hat{\boldsymbol{\omega}}$; it can be shown that some estimators go through a phase transition as $\alpha = \frac{P}{N}$ increases, with the average loss steadily reaching smaller values[EB01].

Example 2: low-density parity-check (LDPC) codes

A model for the communication between two points is provided by information theory[Sha48; CT91]: the message $\mathbf{x} \in \{0, 1\}^N$ being conveyed goes through a noisy channel that corrupts this message

into $\mathbf{y} \in \{0, 1\}^N$; for instance, the binary symmetric channel depicted in Figure 1.6b has

$$\begin{cases} y_i = x_i \oplus \xi_i, & i = 1, \dots, N, \\ \xi_i \sim \text{B}(p), \end{cases} \quad (1.33)$$

where the symbol $\oplus$ is used to represent the sum modulo 2. In order to identify and correct the errors introduced by the channel, the original message $\mathbf{x}$ needs to have some redundancy; this may be done by setting some of its bits as linear combinations of the remaining ones[Mac03]

$$x_a = x_{i_1^a} \oplus x_{i_2^a} \oplus \dots \oplus x_{i_{p-1}^a}, \quad a = 1, \dots, M. \quad (1.34)$$

The prior is then a uniform measure over all $\mathbf{x}$ which satisfy the above (*parity-check*) constraints

$$P_0(\mathbf{x}) = \prod_{a=1}^M \mathbb{I}(x_a \oplus x_{i_1^a} \oplus \dots \oplus x_{i_{p-1}^a} = 0), \quad (1.35)$$

so that

$$\begin{aligned} P(\mathbf{x}|\mathbf{y}) &\propto \prod_{i=1}^N P(y_i|x_i) \prod_{a=1}^M P_0(\mathbf{x}_{\partial a}) \\ &= \prod_{i=1}^N p \left(\frac{1-p}{p} \right)^{x_i \oplus y_i} \prod_{a=1}^M \mathbb{I}(x_a \oplus x_{i_1^a} \oplus \dots \oplus x_{i_{p-1}^a} = 0), \end{aligned} \quad (1.36)$$

since $\xi_i \sim \text{B}(p) = p\delta_{\xi_i,1} + (1-p)\delta_{\xi_i,0} = p\left(\frac{1-p}{p}\right)^{\xi_i}$.

Two estimators are typically employed in this case: the word MAP, $\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y})$, and the symbol MAP, $\hat{x}_i = \arg \max_{x_i} P(x_i|\mathbf{y}) \forall i$. Both may be computed with low computational cost whenever the factor graph associated to (1.36) is sparse; that is why the code is said to be *low-density*.

For small p and large $R = \frac{M}{N}$, both estimators are bound to provide good results[VSK02; RU08], i.e. the error probability asymptotically goes to zero as N goes to infinity; as p increases and R decreases, a sharp transition occurs, after which the probability of error remains finite. This transition point is known as the channel capacity in the information theory literature.

Example 3: reputation systems

In scenarios where agents interact with each other, it is often important for an agent to quantify how much the others can be trusted[RKZF00]. That is the case for instance in sensor networks and e-commerce websites such as eBay. While trust is generally hard to quantify, we may assume it to be correlated with reputation, e.g. whenever a number of agents publicly express their confidence in another one, that agent is considered to be trustworthy.

We denote the reputations in a network of N agents by a vector $\mathbf{r} \in \{-1, +1\}^N$, such that $r_i = +1$ if agent i has a good reputation and $r_i = -1$ otherwise. Furthermore, a set of ratings $J \in \{-1, 0, 1\}^{N \times N}$ is available, $J_{ij} = \pm 1$ depending on whether agent i emitted a positive/negative

rating to agent j and $J_{ij} = 0$ if no rating was emitted at all. With that in mind, a LVM may be formulated[MV13]

$$\begin{cases} J_{ij} = \xi_{ij} r_i r_j, & \forall (i, j) \in \Omega, \\ \xi_{ij} \sim p \delta_{\xi_{ij}, +1} + (1 - p) \delta_{\xi_{ij}, -1}, \\ r_i \sim q \delta_{r_i, +1} + (1 - q) \delta_{r_i, -1}, & i = 1, \dots, N, \end{cases} \quad (1.37)$$

with the noise term ξ being related to uncertainties affecting opinion formation. Notice that a colluding behavior is assumed to exist (see Figure 1.6c), since two untrustworthy agents will typically emit each other favorable opinions, i.e. in the noiseless case $J_{ij} = +1$ if $r_i = r_j = -1$. The posterior is then written as

$$P(\mathbf{r}|\mathbf{J}) \propto \prod_{(i,j) \in \Omega} P(J_{ij}|r_i, r_j) \prod_{i=1}^N P_0(r_i) \quad (1.38)$$

$$= \prod_{(i,j) \in \Omega} \sqrt{p(1-p)} \left(\frac{p}{1-p} \right)^{\frac{1}{2} J_{ij} r_i r_j} \prod_{i=1}^N \sqrt{q(1-q)} \left(\frac{q}{1-q} \right)^{\frac{1}{2} r_i}. \quad (1.39)$$

and if we are able to compute the marginals $P(r_i|\mathbf{J}) = \sum_{\mathbf{r}_{\setminus i}} P(\mathbf{r}|\mathbf{J})$, the symbol MAP estimate is easily obtained from $\hat{r}_i = \arg \max_{r_i} P(r_i|\mathbf{J}) \forall i$.

Box 1.5: Average performance of an estimator

The average performance of an estimator $\hat{\mathbf{x}}(\mathbf{y})$ may be assessed empirically through the following procedure: one samples $\mathbf{x}^{(0)} \sim P_0(\mathbf{x})$, then applies the generative process described by the LVM, effectively sampling $\mathbf{y}^{(0)} \sim P(\mathbf{y}|\mathbf{x}^{(0)})$, and finally computes $\hat{\mathbf{x}}(\mathbf{y}^{(0)})$. The error in the estimate is then given by the loss function associated to the estimator, e.g. if the MMSE estimator was used then $\ell(\hat{\mathbf{x}}(\mathbf{y}), \mathbf{x}) = \|\hat{\mathbf{x}}(\mathbf{y}) - \mathbf{x}\|_2^2$. After repeating these steps for many samples $\{\mathbf{x}^{(k)}, \mathbf{y}^{(k)}\}$, the average loss is obtained from

$$\bar{\ell} = \frac{1}{N_{\text{exp}}} \sum_{k=1}^{N_{\text{exp}}} \ell(\hat{\mathbf{x}}(\mathbf{y}^{(k)}), \mathbf{x}^{(k)}). \quad (1.40)$$

However, this procedure may be computationally expensive depending on the complexity of computing the estimator. One could alternatively perform the average analytically

$$\mathbb{E}_{\mathbf{x}, \mathbf{y}} \ell(\hat{\mathbf{x}}(\mathbf{y}), \mathbf{x}) = \mathbb{E}_{\mathbf{y}} \int d\mathbf{x} P(\mathbf{x}|\mathbf{y}) \ell(\hat{\mathbf{x}}(\mathbf{y}), \mathbf{x}), \quad (1.41)$$

and $\bar{\ell}$ should converge to $\mathbb{E}_{\mathbf{x}, \mathbf{y}} \ell(\hat{\mathbf{x}}(\mathbf{y}), \mathbf{x})$ as $N_{\text{exp}} \rightarrow \infty$. In some cases this calculation can be done by means of standard techniques from the statistical physics of disordered systems, such as the replica and cavity methods, as we will see in section 3.2.

1.3 Combinatorial optimization

Optimization problems are ubiquitous throughout science, and can be written in their most general form as

$$\begin{aligned} \min \quad & f(\mathbf{x}), \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, P, \end{aligned} \tag{1.42}$$

where f and $\{g_i\}$ are any functions of $\mathbf{x} \in \chi^N$. Since in the general case no guarantees are provided, one usually focuses on convex functions[BV04], for which local optima are also global. Problems such as (1.42) are usually solved by means of iterative algorithms that try to improve on an initial guess; in particular, whenever the functions are linear or quadratic, there are algorithms that can provably find the solution in polynomial time in N .

In this section we will focus on problems for which the set of solutions is finite, known as combinatorial optimization problems[PS98]. Such problems are defined by a set of variables $\{x_i\}$ and a cost function $C : \chi^N \rightarrow \mathbb{R}$ associated to each combination of values or *configuration* $\mathbf{x} \in \chi^N$ assumed by these variables, χ being a finite set. We may then ask different questions, each of them related to a specific computational task

1. what is a minimum-cost configuration $\mathbf{x}^* = \arg \min_{\mathbf{x}} C(\mathbf{x})$? (optimization)
2. is there a configuration with cost smaller than C_0 , that is, is there $\mathbf{x} \in \chi^N$ such that $C(\mathbf{x}) < C_0$? (decision)
3. what is the number of configurations with a given cost? (counting)

The cost function $C(\mathbf{x})$ may be seen as the Hamiltonian of a physical system which microstates follow a Gibbs-Boltzmann distribution

$$P_\beta(\mathbf{x}) = \frac{1}{Z_\beta} e^{-\beta C(\mathbf{x})}, \tag{1.43}$$

and the optimization problem above amounts to looking for the ground states of this system[MM09], that is, we are interested in the $\beta \rightarrow \infty$ limit.

A whole area of computer science is devoted to determining the complexity of performing the tasks above for a number of problems[MM11]. It is remarkable how very similar problems can have very different properties; for instance, determining whether there is a path linking two vertices in a graph can be done in polynomial time, while if we require the path to go through each vertex once, the problem becomes NP-complete (the *Hamiltonian cycle* problem).

As a matter of fact, it often happens that tuning a single parameter makes a problem in P become NP. That is the case with the k -SAT problem when going from $k = 2$ to $k = 3$, and with the q -coloring problem when going from $q = 2$ to $q = 3$ – both these problems shall be defined in the next subsection. This sudden behavior can effectively be seen as a phase transition, which happens as the geometric structure of the configuration space is modified, making it harder to look for solutions. Good examples come from constraint-satisfaction problems, which we shall study next.

1.3.1 Constraint-satisfaction problems

The forementioned phase transition phenomena is better understood if we restrict ourselves to a subclass of combinatorial optimization problems known as constraint-satisfaction problems (CSPs)[MM11; KMR+07]. In this case, there are a number of constraints

$$f_a(\mathbf{x}_{\partial a}) = y_a, \quad a = 1, \dots, M, \tag{1.44}$$

to be satisfied and the cost function evaluates how many of them are *not* satisfied by a given configuration $\mathbf{x}$, that is

$$C(\mathbf{x}) = \sum_{a=1}^M \mathbb{I}[f_a(\mathbf{x}_{\partial a}) \neq y_a]. \quad (1.45)$$

so that $C(\mathbf{x}) = 0$ if and only if $f_a(\mathbf{x}_{\partial a}) = y_a$ for all a . We are then interested in 1) what is a configuration that satisfies the most constraints; 2) if there is a configuration that satisfies all constraints and; 3) what is the number of configurations for which a given fraction of constraints is satisfied.

Following (1.43), we may write

$$P_\beta(\mathbf{x}) = \frac{1}{\mathcal{Z}_\beta} \prod_{a=1}^M e^{-\beta \mathbb{I}[f_a(\mathbf{x}_{\partial a}) \neq y_a]}, \quad (1.46)$$

so that in the $\beta \rightarrow \infty$ limit

$$P_\infty(\mathbf{x}) = \begin{cases} \frac{1}{\mathcal{Z}_\infty} & \text{if } f_a(\mathbf{x}_{\partial a}) = y_a \ \forall a \\ 0 & \text{otherwise,} \end{cases} \quad (1.47)$$

and $\mathcal{Z}_\infty = \lim_{\beta \rightarrow \infty} \sum_{\mathbf{x}} \prod_{a=1}^M e^{-\beta \mathbb{I}[f_a(\mathbf{x}_{\partial a}) \neq y_a]}$ counts the number of configurations that satisfy all constraints – note that we have assumed at least one such configuration exists, otherwise $P_\infty(\mathbf{x})$ concentrates on configurations that satisfy as many constraints as possible.

The set of constraints is also well described by a latent variable model

$$\begin{cases} y_a = f_a(\mathbf{x}_{\partial a}) + \eta_a, & i = 1, \dots, M \\ \boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 \mathbb{1}_M), \\ \mathbf{x} \sim \text{Uniform}(\xi^N), \end{cases} \quad (1.48)$$

in the limit $\sigma^2 \rightarrow 0$. Given (1.46) and (1.48), it seems natural to represent CSPs using factor graphs; in particular, we will introduce the notion of *soft* constraints to designate factor nodes that allow for noisy outputs, and otherwise denote them by *hard* constraints. We have already seen in subsection 1.2.1 an example of a LVM that contains both soft and hard constraints: the factor graph associated to the LDPC code (1.36) can be seen in Figure 1.7.

Some examples of CSPs are [MM09]

Example 1: k -XOR-SAT

Given $\mathbf{b} \in \{0, 1\}^M$, solve the following system of Boolean linear equations for $\mathbf{x} \in \{0, 1\}^N$

$$x_{i_1^a} \oplus x_{i_2^a} \oplus \dots \oplus x_{i_K^a} = b_a, \quad a = 1, \dots, M. \quad (1.49)$$

By mapping $\{0, 1\} \rightarrow \{+1, -1\}$, so that $x \oplus y \rightarrow x \cdot y$, we may write the cost function as

$$C(\mathbf{x}) = \frac{1}{2} \sum_a \left(1 - b_a \prod_{i \in \partial a} x_i \right), \quad (1.50)$$

Figure 1.7: Factor graph corresponding to the LDPC code (1.36), containing both soft (left) and hard (right) constraints.

where $\partial a = \{i_1^a, i_2^a, \dots, i_K^a\}$. Dropping constant terms and rescaling the expression above lead us to

$$P_\beta(\mathbf{x}) = \frac{1}{Z_\beta} \prod_{a=1}^M \exp \left\{ \beta b_a \prod_{i \in \partial a} x_i \right\}, \quad (1.51)$$

i.e. solving the k -XOR-SAT problem is analogous to finding the ground states of a p -spin model from statistical physics.

Example 2: k -SAT

Given $J \in \{0, 1\}^{M \times N}$, find an assignment $\mathbf{x}$ that satisfies the following Boolean formula

$$\bigwedge_{a=1}^M \left[(J_{i_1^a}^a \oplus x_{i_1^a}) \vee (J_{i_2^a}^a \oplus x_{i_2^a}) \vee \dots \vee (J_{i_K^a}^a \oplus x_{i_K^a}) \right], \quad (1.52)$$

For $\partial a = \{i_1^a, i_2^a, \dots, i_K^a\}$

$$C(\mathbf{x}) = \sum_{a=1}^M \left[1 - \prod_{i \in \partial a} \mathbb{I}(x_i \neq J_{i_1^a}^a) \right], \quad (1.53)$$

that is, the a -th constraint is not satisfied if and only if $\mathbf{x}_{\partial a} = J_{\partial a}^a$.

Example 3: q -coloring

Represent each vertex of a given graph $\mathcal{G} = (V, E)$ by a component of $\mathbf{x} \in \{1, \dots, q\}^N$; we would like the value assigned to each vertex to be different from those assigned to its neighbors, that is

$$\delta(x_i, x_j) = 0, \quad \forall (i, j) \in E, \quad (1.54)$$

so that $C(\mathbf{x}) = \sum_{(i,j) \in E} \delta(x_i, x_j)$ and

$$P_\beta(\mathbf{x}) = \frac{1}{Z_\beta} \prod_{(i,j) \in E} e^{-\beta \delta(x_i, x_j)} \quad (1.55)$$

also characteristic of the antiferromagnetic Potts model from statistical physics.

Historically, an algorithm that has had great success in solving optimization problems is the *simulated annealing*[KGV83], that perform Monte Carlo steps at steadily decreasing temperatures. However, whenever the constraints depend only on a few variables – i.e., the factor graph associated to the problem is sparse – *local algorithms* that perform better than simulated annealing may be devised[GHH14].

Using arguments similar from those used at deriving (1.9), we may formulate a variant of the belief propagation algorithm to compute

$$\mathbf{x}^* = \arg \max \prod_{a=1}^M \psi(\mathbf{x}_{\partial a}), \quad (1.56)$$

known as *max-product*[Bar11; Mur12]; the idea is to reorganize the factors as done in (1.7), but using the max operator instead of a sum, to obtain

$$\begin{aligned} m_{a \rightarrow i}(x_i) &= \max_{\mathbf{x}_{\partial a \setminus i}} \psi(\mathbf{x}_{\partial a}) \prod_{j \in \partial a \setminus i} m_{j \rightarrow a}(x_j) \\ &= \max_{\mathbf{x}_{\partial a \setminus i}} \mathbb{I}[f_a(\mathbf{x}_{\partial a}) = y_a] \prod_{j \in \partial a \setminus i} m_{j \rightarrow a}(x_j). \end{aligned} \quad (1.57)$$

Iterating this set of equations until convergence should yield a satisfying assignment

$$x_i^* = \arg \max_{x_i} \mu(x_i) = \arg \max_{x_i} \prod_{a \in \partial i} m_{a \rightarrow i}(x_i), \quad (1.58)$$

provided that i) the underlying factor graph is a tree and ii) the maximum is not degenerate. In case the maximum is degenerate, the decimation procedure described in Box 1.6 should be employed.

Box 1.6: Decimation procedure

The *max-marginals* of a probability distribution $P(\mathbf{x})$ are defined as[MM09]

$$\mu(x_i = x_i^*) = \max_{\mathbf{x} : x_i = x_i^*} P(\mathbf{x}), \quad (1.59)$$

that is, as the maximum probability given all configurations $\mathbf{x}$ that have $x_i = x_i^*$. If the factor graph associated to $P(\mathbf{x})$ is a tree, then the relation used in (1.58)

$$\mu(x_i) \propto \prod_{a \in \partial i} m_{a \rightarrow i}(x_i), \quad (1.60)$$

must be satisfied after the convergence of max-product equations (1.57). Furthermore, if we are able to obtain the correct max-marginals of any *unimodal* distribution $P(\mathbf{x})$, then the mode of this distribution is given by

$$x_i^* = \arg \max_{x_i} \mu(x_i) \quad \forall i. \quad (1.61)$$

However, if the distribution is multimodal, a decimation procedure must be used where the max-marginals are recomputed after each variable is set[MM09; RS09]

Sample a permutation σ of $\{1, 2, \dots, N\}$

Initialize $D = \emptyset$

Iterate through $k \leftarrow \{1, 2, \dots, N\}$

Compute max-marginals $\mu(x_i = x_i^*) = \max_{\mathbf{x}_{\setminus D}: x_i = x_i^*} P(\mathbf{x}_{\setminus D} | \mathbf{x}_D)$ for $i = 1, \dots, N$.

Decimate $x_{\sigma_k}^* = \arg \max_{x_{\sigma_k}} \mu(x_{\sigma_k}); D \leftarrow D \cup \{\sigma_k\}$.

As an example, consider a probability distribution $P(\mathbf{x})$, $\mathbf{x} \in \{0, 1\}^4$, that has two modes, $\mathbf{x}^{(1)} = 1010$ and $\mathbf{x}^{(2)} = 0011$. In this case, there are ties in positions 1 and 4 – $\mu(x_1 = 0) = \mu(x_1 = 1)$ and $\mu(x_4 = 0) = \mu(x_4 = 1)$ – so that assigning (1.61) can yield incorrect results such as 1011 or 0010. By following the procedure above, we would first set x_1 to either 0 or 1, and then work only with the configurations that have x_1 fixed to that value, thus resolving the ambiguity at position 4.

Naturally, the max-product equations (1.57) may be used to obtain approximations to the max-marginals at each decimation step.

The max-product equations should yield good approximations whenever the factor graph is not a tree but is *locally tree-like*, i.e. contains only a few long cycles. For certain problems, however, it might happen that even though the cycles are long, variables that are far apart from each other in the factor graph become correlated[MM09] – in the statistical physics jargon, we would say the system has undergone a *glassy* transition. In this case, one verifies that the belief propagation algorithm either does not converge or that it converges to incorrect max-marginals that do not provide a satisfying assignment. This sort of phenomena shall be further investigated in section 3.3, where the connection of optimization problems to statistical mechanics will be established.

Variational Inference and Message-Passing Algorithms

Tasks such as computing the mode, the marginals and the normalization factor of a probability distribution are often intractable, in the sense that there is no algorithm capable of performing them in polynomial time in the number of variables. In that case, one has no option but to resort to approximations. In this chapter, we describe techniques that allow one to systematically develop algorithms for approximate inference.

While in the last chapter we have introduced a structure that allows us to more easily compute properties of distributions in some specific cases, in the general case these computations become either too costly (for junction trees) or inaccurate (in the case of belief propagation). We will now present a scheme that allows one to systematically derive algorithms that, as belief propagation, can often be seen as *message-passing* procedures over a factor graph.

Even though these algorithms also yield approximate results, through this approach there should be more control over the assumptions being made. In fact, we will be able to rederive the belief propagation equations, but now bringing more insight into its powers and limitations.

2.1 Schemes for approximate inference

We will focus on a typical inference problem[Mur12; Bar11; Bis07] where a set of variables $\mathbf{y} \in \mathbb{R}^M$ has been observed and, given a model that relates these observed variables to a set of latent variables $\mathbf{x} \in \mathbb{R}^N$ by means of a vector of parameters $\boldsymbol{\theta} \in \mathbb{R}^P$, we would like to study the posterior distribution of $\mathbf{x}$ given $\mathbf{y}$, written by Bayes theorem as

$$P(\mathbf{x}|\mathbf{y}; \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{y})} P(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta}_\ell) P_0(\mathbf{x}; \boldsymbol{\theta}_p), \quad (2.1)$$

with $\boldsymbol{\theta} = \{\boldsymbol{\theta}_\ell, \boldsymbol{\theta}_p\}$. A model that is of particular importance in Physics is that with binary latent variables, $\mathbf{x} \in \{+1, -1\}^N$, and pairwise interactions, defined by a set E of tuples (see Appendix)

$$P(\mathbf{x}|\mathbf{y}; \beta, \boldsymbol{\theta}) = \frac{1}{Z(\mathbf{y})} \prod_{(i,j) \in E} e^{\beta y_{ij} x_i x_j} \prod_{i=1}^N e^{\beta \theta_i x_i}, \quad (2.2)$$

rising for instance from the following latent variable model

$$\begin{cases} y_{ij} = \eta_{ij} x_i x_j, & \forall (i, j) \in E, \\ \eta_{ij} \sim \text{B}_{\pm 1}(p) = p \delta_{\eta_{ij}, +1} + (1 - p) \delta_{\eta_{ij}, -1}, \\ x_i \sim \text{B}_{\pm 1}(q_i), \end{cases} \quad (2.3)$$

for which $\beta = \frac{1}{2} \log \frac{p}{1-p}$ and $\theta_i = \frac{1}{2\beta} \log \frac{q_i}{1-q_i}$. It is well known by physicists that for N large and the vast majority of graph ensembles, computing the partition function $\mathcal{Z}(\mathbf{y})$ and the moments and cumulants of this posterior distribution can be quite difficult. For this purpose they have developed, ever since the beginning of the 20th century, techniques that allow one to execute these tasks approximately known as mean-field theories[Sal01].

In what follows, we will present a scheme for approximating the posterior $P(\mathbf{x}|\mathbf{y})$ by a *tractable* trial distribution $Q(\mathbf{x}; \lambda)$, for which these tasks are easy to execute. Determining Q usually involves minimizing some sort of distance between Q and P ; as we will see, minimizing a particular distance leads to results very similar to those provided by the mean-field theories from statistical physics, but on a more general context.

2.1.1 KL-variational approximate inference

The Kullback-Leibler (KL) divergence is an information-theoretical measure of the distance between two distributions, and is defined as

$$\begin{aligned} \text{KL}[Q\|P] &= \int d\mathbf{x} Q(\mathbf{x}) \log \frac{Q(\mathbf{x})}{P(\mathbf{x}|\mathbf{y})} \\ &= \int d\mathbf{x} Q(\mathbf{x}) \log Q(\mathbf{x}) - \langle \log P(\mathbf{x}, \mathbf{y}) \rangle_Q + \log \mathcal{Z} \geq 0, \end{aligned} \quad (2.4)$$

the inequality coming from the concavity of the logarithm function. Notice that since $\text{KL}[Q\|P] \neq \text{KL}[P\|Q]$, the KL divergence is not properly a metric, but only a premetric. The first term is recognized as (minus) the Shannon entropy of Q , $S[Q] = - \int d\mathbf{x} Q(\mathbf{x}) \log Q(\mathbf{x})$, while the second may be regarded as (minus) a generalized energy $E[Q]$, since for $P(\mathbf{x}, \mathbf{y}) = e^{-\mathcal{H}(\mathbf{x})}$, $\langle \log P(\mathbf{x}|\mathbf{y}) \rangle_P = -\langle \mathcal{H} \rangle_P$. Since the last term is constant, the sum of the first two is the quantity we are interested in minimizing; in analogy with the Helmholtz free energy from statistical physics, we will call it a generalized or *variational* free energy[OS01]

$$F[Q] \equiv E[Q] - S[Q] \quad (2.5)$$

and for now we do not introduce the notion of temperature, or equivalently, we set $\beta = 1$. In order to proceed, one needs to specify a form for Q – a factorization and/or parametrization. The simplest factorization one can choose is

$$Q(\mathbf{x}) = \prod_i q_i(x_i), \quad (2.6)$$

so that $S[Q] = \sum_i \int dx_i q_i(x_i) \log q_i(x_i)$, and by calculating the functional derivative of $F[Q]$ with respect to q_i and setting it to zero one obtains

$$\left. \frac{\delta F}{\delta q_i} \right|_{q_i^*} = 0 \quad \Rightarrow \quad q_i^*(x_i) = \frac{1}{z_i} e^{\langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{q_{\sim i}}}, \quad (2.7)$$

where $\langle \cdot \rangle_{q_{\sim i}} = \int \cdot \left[\prod_{j \neq i} dx_j q_j(x_j) \right]$ denotes the mean is over all factors q_j with j different of i . For instance, applying this formula to (2.2) would yield

$$q_i^*(x_i) = \frac{1}{z_i} e^{\beta \left(\theta_i + \sum_{j \in \partial i} y_{ij} \langle x_j \rangle \right) x_i}, \quad (2.8)$$

so that

$$\langle x_i \rangle_Q = \tanh \left[\beta \left(\theta_i + \sum_{j \in \partial i} y_{ij} \langle x_j \rangle_Q \right) \right]. \quad (2.9)$$

A very similar equation is obtained in the Curie-Weiss model from statistical physics, where the $J_{ij} = J$ is assumed to be uniform so that every spin variable has the same mean $\langle x_i \rangle = m$, which also gives the magnetization of the system; in our case, however, that would be too strong of an assumption. In order to find the values of $\{\langle x_i \rangle_Q\}$ that satisfy the fixed-point conditions (2.9), one needs to solve a non-linear system of N equations; luckily, in this case, a fixed-point iteration is guaranteed to obtain a solution (see Box 2.1).

Box 2.1: Fixed-point iterations

Suppose we want to find a solution of

$$\mathbf{x} - f(\mathbf{x}) = 0, \quad (2.10)$$

where $\mathbf{x} \in \mathbb{R}^N$ and f is a non-linear function. A fixed-point iteration is a root-finding algorithm that consists in repeatedly iterating the following map

$$\mathbf{x}^{(t+1)} = f(\mathbf{x}^{(t)}), \quad (2.11)$$

hoping that convergence is eventually achieved given a choice of $\mathbf{x}^{(0)}$. Variations of these scheme can be considered, for instance

$$x_i^{(t+1)} = f(x_1^{(t+1)}, \dots, x_{i-1}^{(t+1)}, x_i^{(t)}, x_{i+1}^{(t)}, \dots, x_N^{(t)}) \quad \forall i, \quad (2.12)$$

that is, a *sequential* update schedule in which the variables that already have been updated at this step are used. Iterative algorithms with sequential update schedules are known to have better properties than their parallel equivalent – for instance, the Gauss-Seidl algorithm for solving linear systems converges faster than the Jacobi algorithm.

Thus in order to find a set of values that satisfy (2.9), one could iterate the following equations

$$m_i^{(t+1)} = \tanh \left[\beta \left(\theta_i + \sum_{\substack{j \in \partial i \\ \sigma_j < \sigma_i}} y_{ij} m_j^{(t+1)} + \sum_{\substack{j \in \partial i \\ \sigma_j > \sigma_i}} y_{ij} m_j^{(t)} \right) \right]. \quad (2.13)$$

given a permutation σ of $\{1, 2, \dots, N\}$ that indicates the order of the updates (i.e., $\sigma_i = 1$ and $\sigma_j = N$ indicates i and j will be the first and variables to be updated, respectively). Such an algorithm is guaranteed to converge to a local minimum of F_{MF} – see discussion regarding (2.82) below.

Since in our example x_i is binary, we can parametrize the factors $q_i(x_i)$ by their means $m_i = \langle x_i \rangle_Q$ so that $q_i(x_i) = \frac{1+m_i x_i}{2}$, thus yielding the following expression for the free energy

$$F_{\text{MF}}(\mathbf{m}) = \sum_i \left\{ \frac{1+m_i}{2} \log \frac{1+m_i}{2} + \frac{1-m_i}{2} \log \frac{1-m_i}{2} \right\} - \beta \sum_{(i,j) \in E} y_{ij} m_i m_j - \beta \theta_i \sum_i m_i. \quad (2.14)$$

We could work directly with this expression and employ general-purpose optimization algorithms in order to extremize it, instead of solving the fixed-point conditions (2.9).

The approach derived from the use of the form (2.6) goes by the name of *naive mean-field*, in reference to the mean-field techniques used by physicists. This approximation, even though tractable, often do not provide good results, since it assumes variables to be independent and thus neglects any correlations between them, that is, $\langle x_i x_j \rangle_Q = \langle x_i \rangle_Q \langle x_j \rangle_Q$. One could include first-order *linear response* corrections by using that

$$\chi_{ij} = \frac{\partial m_i}{\partial(\beta\theta_j)} = \langle x_i x_j \rangle - m_i m_j \quad (2.15)$$

and then computing the partial derivative using the expression obtained within the mean-field approximation; for the model (2.2), we obtain

$$\frac{\partial m_i}{\partial(\beta\theta_j)} = \frac{\partial m_i}{\partial m_j} \frac{\partial m_j}{\partial(\beta\theta_j)} = \beta y_{ij} (1 - m_i^2)(1 - m_j^2), \quad (2.16)$$

leading to the following corrected free energy

$$F_{\text{TAP}}(\mathbf{m}) = F_{\text{MF}}(\mathbf{m}) - \beta^2 \sum_{(i,j) \in E} y_{ij}^2 (1 - m_i^2)(1 - m_j^2). \quad (2.17)$$

By deriving this expression with respect to m_i , we are able to determine the respective fixed-point conditions

$$m_i = \tanh \left[\beta \left(\theta_i + \sum_{j \in \partial i} y_{ij} m_j \right) - \beta^2 \sum_{j \in \partial i} y_{ij}^2 (1 - m_j^2) m_i \right]. \quad (2.18)$$

This is known as the TAP approximation, as this same equation was derived in the context of statistical physics by Thouless, Anderson and Palmer[TAP77]. A more accurate, systematic derivation than the one presented here can be performed by means of the cavity method or of Plefka's expansion[OS01].

2.1.2 Bethe approximation

$$\begin{aligned} P(x_1, \dots, x_5) &= P(x_1, x_2, x_3) P(x_4, x_5 | x_3) \\ &= P(x_1, x_2, x_3) \frac{P(x_3, x_4, x_5)}{P(x_3)} \end{aligned}$$

Figure 2.1: Any distribution that is represented by a tree in the factor graph may be written in form (2.19).

One may improve on the mean-field approximation by including factors that reflect the dependence between variables in the original factor graph, that is, for each factor $P(y_a | \mathbf{x}_{\partial a})$ in the likelihood we include a factor $q_a(\mathbf{x}_{\partial a})$ in the approximation so that

$$Q(\mathbf{x}) = \frac{\prod_a q_a(\mathbf{x}_{\partial a})}{\prod_i q_i^{|\partial i|-1}(x_i)}, \quad (2.19)$$

and the product in the denominator can be understood as maintaining the distribution dimensionally consistent, i.e. since variables appear more than once in the numerator we compensate by making them appear in the denominator[Bar11]. As a matter of fact, one can show that any distribution that has a tree representation in the factor graph may be written as (2.19) (see Figure 2.1).

Replacing (2.19) in (2.5) yields

$$\begin{aligned} F_{\text{Bethe}}[\{q_a\}, \{q_i\}] &= - \sum_a \langle \log P(y_a | \mathbf{x}_{\partial a}) \rangle_{q_a} - \sum_i \langle \log P_0(x_i) \rangle_{q_i} \\ &\quad + \sum_a \sum_{\mathbf{x}_{\partial a}} q_a(\mathbf{x}_{\partial a}) \log q_a(\mathbf{x}_{\partial a}) - \sum_i (\underbrace{|\partial i|}_{c_i} - 1) \sum_{x_i} q_i(x_i) \log q_i(x_i) \\ &= - \sum_a \left\langle \log \frac{\tilde{P}(\mathbf{x}_{\partial a})}{q_a(\mathbf{x}_{\partial a})} \right\rangle_{q_a} + \sum_i c_i \left\langle \log \frac{P_0(x_i)}{q_i(x_i)} \right\rangle_{q_i} \end{aligned} \quad (2.20)$$

where we have defined $\tilde{P}(\mathbf{x}_{\partial a}) = P(y_a | \mathbf{x}_{\partial a}) \prod_{i \in \partial a} P_0(x_i)$. While this expression is correct if the factor graph associated to the posterior is a tree, in the general case one must ensure that, for every pair (i, a) , the consistency condition $q_i(x_i) = \sum_{\mathbf{x}_{\partial a}} q_a(\mathbf{x}_{\partial a})$ is verified. In order to do that, we introduce the following Lagrangian[YFW00; YFW01]

$$\mathcal{L}[\{q_a\}, \{q_i\}, \{\lambda_{ia}\}] = F_{\text{Bethe}}[\{q_a\}, \{q_i\}] + \sum_{(i,a) \in E} \sum_{x_i} \lambda_{ia}(x_i) \left\{ q_i(x_i) - \sum_{\mathbf{x}_{\partial a \setminus i}} q_a(\mathbf{x}_{\partial a}) \right\}, \quad (2.21)$$

and equate to zero the functional derivatives with respect to all factors $\{q_a\}$ and $\{q_i\}$

$$\begin{aligned} \frac{\delta \mathcal{L}}{\delta q_i} &= \log P_0 - \log q_i + \sum_{a \in \partial i} \lambda_{ia} = 0 \quad \Rightarrow \quad q_i(x_i) = \frac{1}{z_i} P_0(x_i) \left[\prod_{a \in \partial i} e^{\lambda_{ia}(x_i)} \right]^{\frac{1}{c_i}}, \\ \frac{\delta \mathcal{L}}{\delta q_a} &= -\log \tilde{P} + \log q_a - \sum_{i \in \partial a} \lambda_{ia} = 0 \quad \Rightarrow \quad q_a(\mathbf{x}_{\partial a}) = \frac{1}{z_a} \tilde{P}(\mathbf{x}_{\partial a}) \prod_{i \in \partial a} e^{\lambda_{ia}(x_i)}. \end{aligned} \quad (2.22)$$

The normalization factors z_a and z_i guarantee all q_a and q_i are normalized to 1. If we further define $\hat{m}_{i \rightarrow a}(x_i) = e^{\lambda_{ia}(x_i)}$ and introduce $\{m_{a \rightarrow i}\}$ such that $\hat{m}_{i \rightarrow a} = \prod_{b \in \partial i \setminus a} m_{a \rightarrow i}$, these expressions become

$$q_i(x_i) = \frac{1}{z_i} P_0(x_i) \prod_{a \in \partial i} m_{a \rightarrow i}(x_i), \quad q_a(\mathbf{x}_{\partial a}) = \frac{1}{z_a} \tilde{P}(\mathbf{x}_{\partial a}) \prod_{i \in \partial a} \hat{m}_{i \rightarrow a}(x_i), \quad (2.23)$$

and finally, by setting $\frac{\delta \mathcal{L}}{\delta \lambda_{ia}} = 0$ one enforces the consistency constraints to obtain

$$m_{a \rightarrow i}(x_i) = \frac{z_i}{z_a} \sum_{\mathbf{x}_{\partial a \setminus i}} P(y_a | \mathbf{x}_{\partial a}) \prod_{j \in \partial a \setminus i} [P_0(x_j) \hat{m}_{j \rightarrow a}(x_j)], \quad \forall (a, i), \quad (2.24)$$

which can be recognized as the belief propagation equations (1.9). While in chapter 1 we derived these equations on a tree as the result of a dynamic programming procedure, they have now been obtained by means of a variational argument. This allows us to see how close this technique is to other mean-field methods; in particular, one could use better mean-field approximations – known in statistical physics as cluster variational methods – to propose better, *generalized* belief

propagation algorithms[YFW00] that take into account correlations between larger sets of variables. This derivation also enables us to understand that, throughout its iterations, belief propagation is essentially solving an optimization problem.

Box 2.2: Bethe approximation for the Ising model

It is instructive to derive the Bethe free energy for the Ising model (2.2); since $x_i \in \{+1, -1\}$, the messages can be parametrized by

$$h_{i \rightarrow j} = \theta_i + \frac{1}{2\beta} \log \frac{\hat{m}_{i \rightarrow (ij)}(+1)}{\hat{m}_{i \rightarrow (ij)}(-1)}, \quad u_{j \rightarrow i} = \frac{1}{2\beta} \log \frac{m_{(ij) \rightarrow i}(+1)}{m_{(ij) \rightarrow i}(-1)}, \quad (2.25)$$

so that

$$q_i(x_i) = \frac{1}{z_i} e^{\beta \theta_i x_i} \prod_{j \in \partial i} \frac{e^{\beta u_{j \rightarrow i} x_i}}{2 \cosh(\beta u_{j \rightarrow i})},$$

$$q_{ij}(x_i, x_j) = \frac{1}{z_{ij}} e^{\beta y_{ij} x_i x_j} \frac{e^{\beta h_{i \rightarrow j} x_i} e^{\beta h_{j \rightarrow i} x_j}}{4 \cosh(\beta h_{i \rightarrow j}) \cosh(\beta h_{j \rightarrow i})}, \quad (2.26)$$

and the BP equations become

$$h_{i \rightarrow j} = \theta_i + \sum_{k \in \partial i \setminus j} u_{k \rightarrow i}, \quad u_{j \rightarrow i} = \frac{1}{\beta} \tanh^{-1} [\tanh(\beta y_{ij}) \tanh(\beta h_{j \rightarrow i})]. \quad (2.27)$$

By inserting the parametrized marginals (2.26) in the general expression (2.20) we obtain

$$F_{\text{Bethe}}(\mathbf{h}, \mathbf{u}) = - \sum_{(i,j) \in E} \log z_{ij} - \sum_i \log z_i + 2 \sum_{(i,j) \in E} \log \frac{z_i}{z_{i \rightarrow j}}, \quad (2.28)$$

where $z_{i \rightarrow j} = \sum_{x_i} e^{\beta \theta_i x_i} \prod_{k \in \partial i \setminus j} \frac{e^{\beta u_{k \rightarrow i} x_i}}{2 \cosh(\beta u_{k \rightarrow i})}$, and

$$z_{ij} = \cosh(\beta y_{ij}) [1 + \tanh(\beta h_{i \rightarrow j}) \tanh(\beta u_{j \rightarrow i})],$$

$$z_i = \frac{2 \cosh[\beta(\theta_i + \sum_{j \in \partial i} u_{j \rightarrow i})]}{\prod_{j \in \partial i} 2 \cosh(\beta u_{j \rightarrow i})}, \quad \frac{z_i}{z_{i \rightarrow j}} = \frac{1}{2} [1 + \tanh(\beta h_{i \rightarrow j}) \tanh(\beta u_{j \rightarrow i})]. \quad (2.29)$$

Broadly speaking, the Bethe free energy may be expressed as a function of the messages using a generalization of (2.28) (see e.g. [MM09])

$$\begin{aligned} F_{\text{Bethe}}[\mathbf{m}, \hat{\mathbf{m}}] &= - \sum_a \log z_a - \sum_i \log z_i + \sum_{(i,a) \in E} \log \frac{z_i}{\hat{z}_{i \rightarrow a}} \\ &= - \sum_a \log \left[\sum_{\mathbf{x}_{\partial a}} P(y_a | \mathbf{x}_{\partial a}) \prod_{i \in \partial a} \hat{m}_{i \rightarrow a}(x_i) \right] - \sum_i \log \left[\sum_{x_i} P_0(x_i) \prod_{a \in \partial i} m_{a \rightarrow i}(x_i) \right] \\ &\quad + \sum_{(i,a) \in E} \log \left[\sum_{x_i} \hat{m}_{i \rightarrow a}(x_i) m_{a \rightarrow i}(x_i) \right] \\ &\equiv - \sum_i F_i - \sum_a F_a + \sum_{(i,a) \in E} F_{ia}. \end{aligned} \quad (2.30)$$

Finally, in order to demonstrate and compare the different approximations we have introduced, we will try to compute the MPM (or symbol MAP) estimator for a model of reputation systems, (1.37). In order to obtain the MPM estimate, one should compute the marginals of the posterior distribution associated to the model and assign to each variable the value that maximizes its marginal; since the posterior associated to this model

$$P(\mathbf{r}|J) = \prod_{(i,j) \in \Omega} \sqrt{p(1-p)} \left(\frac{p}{1-p} \right)^{\frac{1}{2} J_{ij} r_i r_j} \prod_{i=1}^N \sqrt{q(1-q)} \left(\frac{q}{1-q} \right)^{\frac{1}{2} r_i},$$

$$\propto \prod_{(i,j) \in \Omega} e^{\alpha_p J_{ij} r_i r_j} \prod_{i=1}^N e^{\alpha_q r_i}, \quad \alpha_\theta = \frac{1}{2} \log \frac{\theta}{1-\theta}, \quad (2.31)$$

is not tractable, the marginals cannot be written in closed-form and we must rely on approximations. Following the derivations above, the equations to be iterated for each of the schemes are

$$\text{mean-field:} \quad m_i^{(t+1)} = \tanh \left(\alpha_q + \alpha_p \sum_{j \in \partial i} J_{ij} m_j^{(t)} \right); \quad (2.32)$$

$$\text{TAP:} \quad m_i^{(t+1)} = \tanh \left(\alpha_q + \alpha_p \sum_{j \in \partial i} J_{ij} m_j^{(t)} - \alpha_p^2 \sum_{j \in \partial i} (1 - (m_j^{(t)})^2) m_i^{(t)} \right); \quad (2.33)$$

$$\text{Bethe:} \quad \begin{cases} h_{i \rightarrow j}^{(t+1)} = \alpha_q + \sum_{k \in \partial i \setminus j} \tanh^{-1} \left[\tanh(\alpha_p J_{ki}) \tanh(h_{k \rightarrow i}^{(t)}) \right], \\ m_i^{(t+1)} = \tanh \left(\alpha_q + \sum_{j \in \partial i} \tanh^{-1} \left[\tanh(\alpha_p J_{ij}) \tanh(h_{j \rightarrow i}^{(t+1)}) \right] \right); \end{cases} \quad (2.34)$$

where $m_i = \langle r_i \rangle_Q$ is the variable we are interested in – since r_i is binary, the maximum of the marginals is given by $\hat{r}_i = \text{sgn}(m_i)$. We can test the performance of the different algorithms by generating many instances of the problem, running all the algorithms over each of the instances and measuring the error $\ell_{\text{MPM}}(\hat{\mathbf{r}}, \mathbf{r}) = \frac{1}{2} \left(1 - \frac{1}{N} \sum_i \hat{r}_i r_i \right)$ obtained in each run (see Box 1.5). The results are showed in Figure 2.2 – as expected, TAP yields better approximations than the naive mean-field approximation, while Bethe has an improved performance over both, specially for small noise levels.

2.1.3 Other schemes

While we have used the inverse KL divergence, $\text{KL}[Q||P]$, to derive the approximations presented in this section, one may wonder why not use the direct one

$$\text{KL}[P||Q] = \int d\mathbf{x} P(\mathbf{x}) \log \frac{P(\mathbf{x})}{Q(\mathbf{x})}. \quad (2.35)$$

An obvious drawback is that, since we do not know in general how to compute averages with respect to $P(\mathbf{x})$, then the expression above might not have a closed-form expression, thus making

Figure 2.2: Average error of MPM estimates obtained with the naive mean-field, TAP and Bethe approximations respectively, for a model of reputation systems (1.37) and different values of the noise level p . The results were obtained by averaging over 500 instances of size $N = 1024$; the reputation bias q is fixed to 0.7.

more difficult to solve the minimization problem. If we require $Q(\mathbf{x})$ to be a distribution of the exponential family

$$Q_{\text{exp}}(\mathbf{x}) = h(\mathbf{x}) \exp\{\eta(\boldsymbol{\theta}) \cdot \phi(\mathbf{x}) - A(\boldsymbol{\theta})\}, \quad (2.36)$$

then

$$\begin{aligned} \frac{\partial}{\partial \eta_a} \text{KL}[P \| Q_{\text{exp}}] &= - \int d\mathbf{x} P(\mathbf{x}) \phi_a(\mathbf{x}) + \int d\mathbf{x} P(\mathbf{x}) \frac{\partial A}{\partial \eta_a} \\ &= -\langle \phi_a \rangle_P + \langle \phi_a \rangle_{Q_{\text{exp}}} = 0, \end{aligned} \quad (2.37)$$

that is, minimizing the KL divergence implies on matching the moments of the two distributions; still, computing $\langle \phi_a \rangle_P$ might not be possible.

We will once again work with probability distributions factored as

$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{\mathcal{Z}(\mathbf{y})} P_0(\mathbf{x}) \prod_{a=1}^M P(y_a | \mathbf{x}_{\partial a}), \quad (2.38)$$

and approximate them by

$$Q(\mathbf{x}) = \frac{1}{\tilde{\mathcal{Z}}} P_0(\mathbf{x}) \prod_a q_a(\mathbf{x}), \quad (2.39)$$

but in order to avoid intractable computations, instead of minimizing $\text{KL}[P \| Q]$ we will introduce the factors of $P(\mathbf{x}|\mathbf{y})$ one at a time through the following procedure

Initialize $Q(\mathbf{x}) = P_0(\mathbf{x})$ and $q_a(\mathbf{x}) = 1$ for $a = 1, \dots, M$.

Repeat until convergence – pick $a \in \{1, \dots, M\}$ uniformly at random and

Update the tilted distribution

$$\tilde{Q}_a(\mathbf{x}) \propto \frac{P(y_a|\mathbf{x})}{q_a(\mathbf{x})} Q(\mathbf{x}), \quad (2.40)$$

replacing the approximated likelihood $q_a(\mathbf{x})$ by the exact one $P(y_a|\mathbf{x}_{\partial a})$ and thus taking Q outside the exponential family.

Project the tilted distribution back to the exponential family

$$Q^{\text{new}}(\mathbf{x}) = \arg \min_Q \text{KL} [\tilde{Q}_a \| Q], \quad (2.41)$$

effectively by updating the parameters of Q so that its moments match the moments of the tilted distribution $\tilde{Q}_a$.

Refine the approximated likelihoods

$$q_a^{\text{new}}(\mathbf{x}) \propto \frac{Q^{\text{new}}(\mathbf{x})}{Q(\mathbf{x})} q_a(\mathbf{x}), \quad \text{for } a = 1, \dots, M. \quad (2.42)$$

At convergence obtain tractable approximation from (2.39).

The approximated posterior $Q(\mathbf{x})$ and the tilted distributions $\tilde{Q}_a(\mathbf{x}) \propto \frac{P(y_a|\mathbf{x})}{q_a(\mathbf{x})} Q(\mathbf{x})$ will have matching moments, $\langle \phi \rangle_Q = \langle \phi \rangle_{\tilde{Q}_a}$ for $a = 1, \dots, M$.

In order to proceed, a specific form/parametrization needs to be chosen for $Q(\mathbf{x})$. The procedure we have just described constitutes the so-called *expectation propagation* (EP) algorithm [Min01; RMO15]. It provides an alternative to the algorithms derived in the last subsections and has experienced some success in applications. While we have mentioned it for completeness, it will not be used throughout this thesis; instead, we shall mostly focus on the Bethe approximation and the belief propagation (BP) algorithm.

In the same way BP fixed-points are described by the minima of the Bethe free energy, EP fixed-points come from the minima of the closely related expectation-consistent free energy [OW05]. It is worth mentioning that subtle relations exist between the BP and EP algorithms, as well as between the Bethe and expectation-consistent free energies – one is often able to obtain one scheme's equations from the other's by enforcing the appropriate limits.

2.2 Inference with continuous variables

Even though the techniques for approximate inference introduced in the previous section can be used for any $\mathbf{x} \in \chi^N$, the examples we gave focused mostly on cases in which the values assumed are discrete, typically $\chi = \{-1, +1\}$. Additional complications arise whenever the values are continuous, i.e. $\chi = \mathbb{R}$; for instance, the sums that need to be performed at each iteration of the belief propagation algorithm (1.9) become integrals

$$m_{a \rightarrow i}^{(t+1)}(x_i) = \frac{1}{z_{a \rightarrow i}} \int d\mathbf{x}_{\partial a \setminus i} P(y_a|\mathbf{x}_{\partial a}) \prod_{j \in \partial a \setminus i} \hat{m}_{j \rightarrow a}^{(t)}(x_j), \quad (2.43)$$

$$\hat{m}_{i \rightarrow a}^{(t)}(x_i) = \frac{1}{\hat{z}_{i \rightarrow a}} P_0(x_i) \prod_{b \in \partial i \setminus a} m_{b \rightarrow i}^{(t)}(x_i), \quad (2.44)$$

that in general cannot be performed analytically. In this case, we may proceed in one of a few different ways

- by discretizing x_i , i.e. using a histogram to represent each of the messages; this is done for instance in [BMP08; BSB10];
- by parametrizing the messages, e.g. $\hat{m}_{i \rightarrow a}(x_i) = e^{-\frac{(x_i - \mu_i)^2}{2\sigma_i^2}}$ [WF01];
- by using Monte Carlo to perform the integrals, either resampling from Gaussian mixtures at each iteration[SII+10] or keeping sets of particles that are repeatedly updated according to our current estimate of the marginals[IM09].

The performance of these different tricks will ultimately depend on how well are we approximating the messages, e.g. on whether the parametrization is close enough to the actual form of the messages.

We will here focus on an alternative technique that has been recently developed, known as *approximate message-passing*[DMM09; Ran10; KMS+12]. This technique is particularly suited for posterior distributions of the form

$$P(\mathbf{x}|\mathbf{y}, F) = \frac{1}{\mathcal{Z}(\mathbf{y})} \prod_{\mu=1}^M P(y_\mu | z_\mu \equiv \mathbf{F}_\mu \cdot \mathbf{x}) \prod_i P_0(x_i). \quad (2.45)$$

where $\mathbf{y} \in \mathbb{R}^M$ and $F \in \mathbb{R}^{M \times N}$ are given, and $\mathbf{F}_\mu$ denotes the μ -th row of F ; such posterior comes up for instance when looking at the following latent variable model

$$\begin{cases} y_\mu = f(z_\mu, \xi_\mu), & \mu = 1, \dots, M, \\ z_\mu = \mathbf{F}_\mu \cdot \mathbf{x}, \\ \xi_\mu \sim P(\xi_\mu), \\ x_i \sim P_0(x_i), & i = 1, \dots, N. \end{cases} \quad (2.46)$$

2.2.1 Approximate message-passing

As a starting point, we will study the case where the likelihood is Gaussian

$$P(y_\mu | F, \mathbf{x}) \propto \exp \left\{ -\frac{1}{2\Delta} \left(y_\mu - \sum_i F_{\mu i} x_i \right)^2 \right\}, \quad (2.47)$$

so that (2.43) becomes

$$m_{\mu \rightarrow i}(x_i) = \frac{1}{z_{i \rightarrow a}} \int \prod_{j \neq i} [dx_j \hat{m}_{j \rightarrow \mu}(x_j)] e^{-\frac{1}{2\Delta} (y_\mu - \sum_i F_{\mu i} x_i)^2}. \quad (2.48)$$

Following [KMS+12], we will assume $F_{\mu i} \sim o\left(\frac{1}{\sqrt{N}}\right)$, so that the exponential in (2.48) may be expanded up to second order and the integral can be performed analytically, yielding

$$m_{\mu \rightarrow i}(x_i) = \frac{1}{z_{\mu \rightarrow i}} e^{-A_{\mu \rightarrow i} \frac{x_i^2}{2} + B_{\mu \rightarrow i} x_i}, \quad (2.49)$$

where

$$A_{\mu \rightarrow i} = \frac{F_{\mu i}^2}{\Delta + \sum_{j \neq i} F_{\mu j}^2 c_{j \rightarrow \mu}}, \quad B_{\mu \rightarrow i} = \frac{F_{\mu i} (y_\mu - \sum_{j \neq i} F_{\mu j} a_{j \rightarrow \mu})}{\Delta + \sum_{j \neq i} F_{\mu j}^2 c_{j \rightarrow \mu}}, \quad (2.50)$$

are functions of the first and second cumulants of the variable-to-node messages

$$a_{i \rightarrow \mu} = \int dx_i x_i \hat{m}_{i \rightarrow \mu}(x_i), \quad c_{i \rightarrow \mu} = \int dx_i x_i^2 \hat{m}_{i \rightarrow \mu}(x_i) - a_{i \rightarrow \mu}^2. \quad (2.51)$$

Finally, from (2.44) we obtain

$$\hat{m}_{i \rightarrow \mu}(x_i) = \frac{1}{z_{i \rightarrow \mu}} P_0(x_i) e^{-\frac{x_i^2}{2} \sum_{\nu \neq \mu} A_{\nu \rightarrow i} + x_i \sum_{\nu \neq \mu} B_{\nu \rightarrow i}}, \quad (2.52)$$

and after iterating (2.50)-(2.52) until convergence, the marginal of x_i can be computed from

$$P(x_i) = \frac{1}{z_i} P_0(x_i) e^{-\frac{x_i^2}{2} \sum_{\mu} A_{\mu \rightarrow i} + x_i \sum_{\mu} B_{\mu \rightarrow i}}. \quad (2.53)$$

This algorithm is sometimes called *relaxed belief propagation* [GW06], or rBP. Note that (2.52) has the form

$$Q_x(x; R, \Sigma^2) = \frac{1}{z(\Sigma^2, R)} P_0(x) e^{-\frac{(x-R)^2}{2\Sigma^2}}, \quad (2.54)$$

so that if we are able to obtain closed-form expressions to the first and second moments of this distribution, then integrals (2.51) can be easily evaluated from these expressions; otherwise, the integrals need to be performed numerically. In what follows, we will denote the mean and variance of $Q_x(x; R, \Sigma^2)$ by $f_a(R, \Sigma^2)$ and $f_c(R, \Sigma^2)$, respectively; the following identities shall be useful

$$f_a(\Sigma^2, R) = R + \Sigma^2 \frac{\partial}{\partial R} \log z(\Sigma^2, R), \quad f_c(\Sigma^2, R) = \Sigma^2 \frac{\partial}{\partial R} f_a(\Sigma^2, R). \quad (2.55)$$

For instance, the mean of the marginal distribution (2.53) is given by

$$a_i = f_a\left(\underbrace{\frac{1}{\sum_{\mu} A_{\mu \rightarrow i}}}_{\Sigma_i^2}, \underbrace{\frac{\sum_{\mu} B_{\mu \rightarrow i}}{\sum_{\mu} A_{\mu \rightarrow i}}}_{R_i}\right). \quad (2.56)$$

If we are working with a latent variable model such as (2.46), then a_i provides an approximation to the MMSE estimator of x_i .

Box 2.3: Using rBP with Gaussian priors

In order to evaluate rBP performance, we can iterate its equations while using a Gaussian prior

$$P_0(\mathbf{x}; \mu, \sigma^2) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x_i - \mu_i)^2}, \quad (2.57)$$

and then compare the approximate MMSE estimate obtained from (2.56) to the true one $\hat{\mathbf{x}}_{\text{MMSE}} = \int d\mathbf{x} P(\mathbf{x}|\mathbf{y})$, that in this case can be computed analytically

$$\hat{\mathbf{x}}_{\text{MMSE}} = \left[\frac{\mathbf{F}^T \mathbf{F}}{\Delta} + \frac{1}{\sigma^2} \mathbb{I}_N \right]^{-1} \left(\frac{\mathbf{F}^T \mathbf{y}}{\Delta} + \frac{1}{\sigma^2} \boldsymbol{\mu} \right). \quad (2.58)$$

Given the prior above, the normalization factor z and the first two cumulants of $Q_x(x; R, \Sigma^2)$ read

$$z(\Sigma^2, R) = \sqrt{\frac{\Sigma^2}{\Sigma^2 + \sigma^2}} e^{-\frac{(\mu - R)^2}{2(\Sigma^2 + \sigma^2)}}, \quad f_a(\Sigma^2, R) = \frac{\sigma^2 R + \Sigma^2 \mu}{\Sigma^2 + \sigma^2}, \quad f_c(\Sigma^2, R) = \frac{\Sigma^2 \sigma^2}{\Sigma^2 + \sigma^2}. \quad (2.59)$$

Each step of rBP consists in iterating equations (2.50) – which are independent of the prior – and (2.51), that in this case can be written as

$$a_{i \rightarrow \mu}^{(t+1)} = \frac{\frac{\mu}{\sigma^2} + \sum_{\nu \neq \mu} B_{\nu \rightarrow \mu}^{(t)}}{\frac{1}{\sigma^2} + \sum_{\nu \neq \mu} A_{\nu \rightarrow \mu}^{(t)}}, \quad c_{i \rightarrow \mu}^{(t+1)} = \frac{1}{\frac{1}{\sigma^2} + \sum_{\nu \neq \mu} A_{\nu \rightarrow \mu}^{(t)}}. \quad (2.60)$$

while the approximate MMSE estimate comes from

$$a_i = \frac{\frac{\mu}{\sigma^2} + \sum_{\mu} B_{\mu \rightarrow i}^*}{\frac{1}{\sigma^2} + \sum_{\mu} A_{\mu \rightarrow i}^*}. \quad (2.61)$$

In Figure 2.3, we show the MSE of the current estimate throughout rBP iterations for matrices of different sizes and compare them to the true MMSE estimate (2.58).

Figure 2.3: Mean-squared error of estimate provided by rBP throughout iterations, while using a Gaussian prior. For both curves, single instances of size $N = 1024$, $M = \alpha N$ were generated, with $\mathbf{x} \in \mathbb{R}^N$, $F \in \mathbb{R}^{M \times N}$ and $\mathbf{y} \in \mathbb{R}^M$ respectively sampled from $\mathbf{x} \sim P_0(\mathbf{x}; \mu, \sigma^2)$, $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$ and $\mathbf{y} \sim \mathcal{N}(F\mathbf{x}, \Delta \mathbb{I}_M)$ – parameters' values are $\mu = 0$, $\sigma^2 = 1$ and $\Delta = 10^{-8}$. Dotted lines give true MMSE estimate (2.58), which in both cases is in good agreement with the estimate provided by rBP after convergence.

At each iteration of the rBP algorithm, we need to compute $o(MN)$ messages – just as in usual BP; as M and N become larger, the computational cost of executing this task may become prohibitive. Thus, it would be interesting to have a scheme to reduce the number of messages at each iteration. In particular, notice that all the messages $\{m_{\mu \rightarrow i}\}$ sent by factor node μ are very similar to each other, as are the messages $\{m_{i \rightarrow \mu}\}$ sent by variable node i , specially as $N \rightarrow \infty$ since $F_{\mu i} \sim o\left(\frac{1}{\sqrt{N}}\right)$.

We first define

$$\omega_\mu \equiv \sum_i F_{\mu i} a_{i \rightarrow \mu}, \quad V_\mu \equiv \sum_i F_{\mu i}^2 c_{i \rightarrow \mu}, \quad (2.62)$$

so that by neglecting all terms proportional to $\frac{1}{N}$ in (2.50)

$$A_{\mu \rightarrow i} \approx \frac{F_{\mu i}^2}{\Delta + V_\mu} \quad B_{\mu \rightarrow i} \approx \frac{F_{\mu i}^2}{\Delta + V_\mu} a_{i \rightarrow \mu} + F_{\mu i} \frac{y_\mu - \omega_\mu}{\Delta + V_\mu}. \quad (2.63)$$

and

$$\Sigma_i^2 = \frac{1}{\sum_\mu A_{\mu \rightarrow i}} = \left[\sum_\mu \frac{F_{\mu i}^2}{\Delta + V_\mu} \right]^{-1}, \quad R_i = \frac{\sum_\mu B_{\mu \rightarrow i}}{\sum_\mu A_{\mu \rightarrow i}} = a_i + \frac{\sum_\mu F_{\mu i} \frac{y_\mu - \omega_\mu}{\Delta + V_\mu}}{\sum_\mu F_{\mu i}^2 \frac{1}{\Delta + V_\mu}}. \quad (2.64)$$

Finally, we need to get rid of the $a_{i \rightarrow \mu}$ and $c_{i \rightarrow \mu}$ terms in (2.62) and write them in terms of $\{\omega_\mu, V_\mu\}$ and $\{\Sigma_i^2, R_i\}$. This can be done by expanding the f_a and f_c functions while neglecting terms of $o(F_{\mu i}^2)$ or higher

$$a_{i \rightarrow \mu} = f_a \left(\frac{1}{\sum_{\nu \neq \mu} A_{\nu \rightarrow i}}, \frac{\sum_{\nu \neq \mu} B_{\nu \rightarrow i}}{\sum_{\nu \neq \mu} A_{\nu \rightarrow i}} \right) \approx f_a(\Sigma_i^2, R_i) - B_{\mu \rightarrow i} \underbrace{\Sigma_i^2 \frac{\partial f_a}{\partial R}}_{f_c(\Sigma_i^2, R_i)} \quad (2.65)$$

$$c_{i \rightarrow \mu} = f_c \left(\frac{1}{\sum_{\nu \neq \mu} A_{\nu \rightarrow i}}, \frac{\sum_{\nu \neq \mu} B_{\nu \rightarrow i}}{\sum_{\nu \neq \mu} A_{\nu \rightarrow i}} \right) \approx f_c(\Sigma_i^2, R_i) \quad (2.66)$$

and the linear correction term in (2.65) is sometimes called Onsager reaction term, as it plays the same role as the correction term in the TAP equations from statistical physics. By putting together equations (2.62)-(2.66), we may prescribe the following iterative algorithm[KMS+12]

$$V_\mu^{(t+1)} = \sum_i F_{\mu i}^2 c_i^{(t)}, \quad (2.67)$$

$$\omega_\mu^{(t+1)} = \sum_i F_{\mu i} a_i^{(t)} - \frac{y_\mu - \omega_\mu^{(t)}}{\Delta + V_\mu^{(t)}} V_\mu^{(t+1)}, \quad (2.68)$$

$$\Sigma_i^{2(t+1)} = \left[\sum_\mu \frac{F_{\mu i}^2}{\Delta + V_\mu^{(t)}} \right]^{-1}, \quad (2.69)$$

$$R_i^{(t+1)} = a_i^{(t)} - \Sigma_i^{2(t+1)} \sum_\mu F_{\mu i} \frac{y_\mu - \omega_\mu^{(t)}}{\Delta + V_\mu^{(t)}}, \quad (2.70)$$

$$a_i^{(t+1)} = f_a(R_i^{(t+1)}, \Sigma_i^{2(t+1)}), \quad (2.71)$$

$$c_i^{(t+1)} = f_c(R_i^{(t+1)}, \Sigma_i^{2(t+1)}). \quad (2.72)$$

known as *approximate message-passing* (AMP). In principle, these iterations could be performed either in parallel – i.e. for each of the equations, all coefficients are updated at once – or in sequential fashion; however, there is no clear sequential schedule available, since the first two equations run on $\mu = 1, \dots, M$ and the remaining ones run on $i = 1, \dots, N$, with each μ variable depending on all i

ones and vice-versa. Thus, we will only consider the parallel schedule for now. Note that apart from the two last iterations, all the other ones consist only on basic operations between vectors and can be performed with low computational cost.

If we assume that $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$, we may use the central limit theorem to further approximate some of the expressions

$$V_\mu = \sum_i F_{\mu i}^2 c_i \approx \frac{1}{N} \sum_i c_i^{(t)} = \bar{c} \Rightarrow \Sigma_i^2 = \frac{\Delta + \bar{c}}{\alpha}, \quad (2.73)$$

where $\alpha = \frac{M}{N}$; by doing that, we end up with scalar variances, i.e. $V_\mu = \bar{c}$ for all μ and $\Sigma_i^2 = \frac{\Delta + \bar{c}}{\alpha}$ for all i , thus obtaining an algorithm with an even lower computational complexity. This is in fact how AMP was first derived[DMM09].

Finally, we mention that a similar algorithm may be derived given any likelihood $P(\mathbf{y}|\mathbf{z} \equiv F\mathbf{x}) = \prod_{\mu=1}^M P(y_\mu|z_\mu \equiv F_\mu \cdot \mathbf{x})$, i.e. not necessarily a Gaussian one as we assumed while deriving AMP. If we define

$$Q_z(z; y, \omega, V) = \frac{1}{Z(y, \omega, V)} P(y|z) e^{-\frac{1}{2V}(y-\omega)^2}, \quad (2.74)$$

and

$$g(y, \omega, V) = \frac{\partial}{\partial \omega} \log Z(y, \omega, V) = \left\langle \frac{y - \omega}{V} \right\rangle_{Q_z}, \quad (2.75)$$

then replacing (2.68)-(2.70) by

$$\omega_\mu^{(t+1)} = \sum_i F_{\mu i} a_i^{(t)} - V_\mu^{(t+1)} g(y_\mu, \omega_\mu^{(t)}, V_\mu^{(t)}), \quad (2.76)$$

$$\Sigma_i^{2(t+1)} = - \left[\sum_\mu F_{\mu i}^2 \frac{\partial g}{\partial \omega}(y_\mu, \omega_\mu^{(t)}, V_\mu^{(t)}) \right]^{-1}, \quad (2.77)$$

$$R_i^{(t+1)} = a_i^{(t)} - \Sigma_i^{2(t+1)} \sum_\mu F_{\mu i} g(y_\mu, \omega_\mu^{(t)}, V_\mu^{(t)}), \quad (2.78)$$

yield the so-called *generalized approximate message-passing* (GAMP) algorithm[Ran10]. For Gaussian likelihoods

$$P(y|z) = \frac{1}{\sqrt{2\pi V}} e^{-\frac{1}{2V}(y-z)^2} \Rightarrow g(y, \omega, V) = \frac{y - \omega}{\Delta + V}, \quad (2.79)$$

thus retrieving AMP equations.

2.3 Variational optimization

The variational free energies we have derived in section 2.1 are usually highly non-convex, having multiple local extrema. Whenever employing iterative algorithms to minimize these free energies, we will have no guarantees on whether the solution obtained is global or only local. However, one verifies empirically that the message-passing algorithms we have derived as fixed-point iterations to solve this problem are often effective in finding good solutions; that is perhaps unexpected, since fixed-point iterations are not a very sophisticated optimization algorithm. In this section, we will briefly investigate this issue, trying to elucidate such unreasonable effectiveness.

2.3.1 Fixed-point iterations

Sometimes the variational free energy $F[Q]$ obtained happens to upper bound the true free energy of the system, since

$$\text{KL}[Q||P] = F[Q] + \log \mathcal{Z} \geq 0.$$

That is the case for the naive mean-field free energy

$$\begin{aligned} F_{\text{MF}}[\mathbf{q}] &= -\langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{q}} - \sum_j \sum_{x_j} q_j(x_j) \log q_j(x_j) \\ &= -\langle \langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{q}_{\setminus i}} \rangle_{q_i} + \sum_{j \neq i} \langle \log q_j(x_j) \rangle_{q_j} + \langle \log q_i(x_i) \rangle_{q_i}, \end{aligned} \quad (2.80)$$

but unfortunately, this is not true in general for other approximations. The Bethe approximation, for instance, provides an upper bound to the true free energy only in a few cases[YFW05; SWW08]; that is because, in order to write the Bethe entropy $S_{\text{Bethe}}[Q]$, we assume the factor graph associated to Q is a tree, which is not usually true.

Another advantage of the naive mean-field approach is that sequential updates

$$q_i^{(t+1)}(x_i) = \frac{1}{z_i} \exp \langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{q}_{\setminus i}}, \quad (2.81)$$

are guaranteed to converge to local optima of F_{MF} , since[Bar11]

$$\begin{aligned} \Delta F_{\text{MF}} &= F_{\text{MF}}[\mathbf{q}_{\setminus i}, q_i^{(t+1)}] - F_{\text{MF}}[\mathbf{q}_{\setminus i}, q_i^{(t)}] \\ &= -\langle \langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{q}_{\setminus i}} \rangle_{q_i^{(t+1)}} + \langle \langle \log P(\mathbf{x}, \mathbf{y}) \rangle_{\mathbf{q}_{\setminus i}} \rangle_{q_i^{(t)}} + \langle \log q_i^{(t+1)} \rangle_{q_i^{(t+1)}} - \langle \log q_i^{(t)} \rangle_{q_i^{(t)}} \\ &= \langle \log q_i^{(t+1)} \rangle_{q_i^{(t)}} - \langle \log q_i^{(t)} \rangle_{q_i^{(t)}} = -\text{KL}[q_i^{(t)} || q_i^{(t+1)}] \leq 0, \end{aligned} \quad (2.82)$$

that is, such update schedule guaranteedly decreases the free energy at each iteration.

As soon as we parametrize the factors of the approximating distribution $Q(\mathbf{x})$, for instance by their means $\langle \mathbf{x} \rangle_Q$ as we did in 2.14 and in general by any vector $\mathbf{a}$, the fixed-point iterations assume the following form

$$\frac{\partial F}{\partial a_i} = \kappa(\mathbf{a}) [a_i - f_i(\mathbf{a})] = 0 \quad \Rightarrow \quad a_i = f_i(\mathbf{a}). \quad (2.83)$$

For the Bethe approximation in particular not only the factors' parameters need to be updated but also the Lagrangian multipliers that ensure the marginals are locally consistent (see 2.21). As discussed in Box 2.1, the relation above may be iterated either in parallel or sequential fashion, with sequential updates providing in general better results; while this is well understood for the mean-field approximation, in the general case it is not.

Whenever updates are performed in parallel

$$a_i^{(t+1)} = f_i(\mathbf{a}^{(t)}), \quad (2.84)$$

we may gain some insight on what is happening by employing a dynamical systems approach, i.e. we treat (2.83) as a discrete map, and by linearizing this map at one of its fixed points $\mathbf{a}^*$ we obtain the Jacobian matrix

$$\mathbb{J}_{ij}(\mathbf{a}^*) = \left. \frac{\partial f_i}{\partial a_j} \right|_{\mathbf{a}^*}. \quad (2.85)$$

We can then perform a linear stability analysis of the fixed points of the map by looking at the spectral radius of the Jacobian

$$\rho(\mathbf{a}^*) = \max_{\lambda_i} \{ |\lambda_i(\mathbb{J}(\mathbf{a}^*))| \}, \quad (2.86)$$

where $\{\lambda_i(\mathbb{J})\}$ is the set of Jacobian's eigenvalues. The fixed point $\mathbf{a}^*$ is stable with respect to small perturbations if and only if $\rho(\mathbf{a}^*) < 1$. Furthermore, $\rho(\mathbf{a}^*)$ is related to the rate of convergence of the dynamical system – the smaller its value, the faster the convergence happens.

Box 2.4: Density evolution

In order to study the asymptotic behavior of fixed-point iteration (2.84), we may consider a to be a random variable that evolves according to

$$a^{(t+1)} \stackrel{d}{=} f(a_1^{(t)}, a_2^{(t)}, \dots) \quad (2.87)$$

where $\{a_1^{(t)}, a_2^{(t)}, \dots\}$ are independent and identically distributed copies of $a^{(t)}$, and $\stackrel{d}{=}$ indicates the equality is in distribution. Our goal is then to determine $P_t(a^{(t)})$ for each time step t ; this may be done by means of a Monte Carlo algorithm known as *population dynamics* [MP01; MM09].

Take for instance the fixed-point iteration associated to the Bethe approximation in the case of a binary model with pairwise interactions (2.2)

$$h^{(t+1)} \stackrel{d}{=} \theta + \frac{1}{\beta} \sum_{i=1}^{c-1} \tanh^{-1} \left[\tanh(\beta J) \tanh(\beta h_i^{(t)}) \right] \quad (2.88)$$

where θ and J are random variables and c is a fixed parameter. The population dynamics algorithm consists in generating a set (or population) of variables $\{h_i\}_{i=1, \dots, N_{\text{pop}}}$ and, at each time step, iterating the relation above for each of the members of the population. At each iteration, θ and J are sampled from their distributions and the h_i are picked uniformly at random from the population.

For $t = 1, \dots, t_{\text{max}}$:

For $i = 1, \dots, N_{\text{pop}}$:

Sample $\theta \sim P(\theta)$, $J \sim P(J)$ and $\{j_1, j_2, \dots, j_{c-1}\}$ uniformly at random from $[N]$

Iterate $h_i = f(\theta, J, \{h_{j_k}\}_{k=1, \dots, c-1})$

At the end of each time step t , $\{h_i\}_{i=1, \dots, N_{\text{pop}}}$ provides a sample of $P_t(h^{(t)})$

In Figure 2.4, we compare the mean magnetization $m = \tanh(\beta h)$ obtained both by this procedure and by averaging over many runs of the belief propagation algorithm over different instances of the model.

Figure 2.4: Comparison between the mean magnetization of a binary model with pairwise interactions obtained both by the density evolution procedure describe in Box 2.4 and by averaging over many runs of the belief propagation algorithm over different instances of the model. Parameters are $c = 3$, $P(J) = B_{\pm 1}(0.9)$ and $P(\theta) = B_{\pm 0.2}(0.7)$.

There are some tricks one may resort to in order to improve the convergence properties of fixed-point iterations; one that is often used consists in *damping* the iteration[Mur12; VSR+14], by updating each a_i using a linear combination of the current value and the new one

$$a_i^{(t+1)} = (1 - \eta) a_i^{(t)} + \eta f_i(\mathbf{a}^{(t)}), \quad (2.89)$$

for a chosen $\eta \in (0, 1]$; for $\eta = 1$ we recover the usual fixed-point iteration, while as η approaches 0 the updates become very small in magnitude, effectively making $\rho(\mathbf{a}^*) \approx 1 - \eta$ so that $\mathbf{a}^*$ is stable but converging towards it is very slow.

2.3.2 General-purpose algorithms for convex optimization

Instead of using fixed-point iterations, we could also employ other optimization algorithms such as gradient descent and Newton's method[BV04]. The gradient descent algorithm consists in iterating

$$a_i^{(t+1)} = a_i^{(t)} - \alpha(t) \left. \frac{\partial F}{\partial a_i} \right|_{\mathbf{a}^{(t)}}, \quad (2.90)$$

where $F(\mathbf{a})$ is the function to be minimized (in this case, the free energy) and $\alpha(t)$ determines the step size. One must be careful in choosing $\alpha(t)$, as too small values may slow down the convergence and too large ones will make a_i oscillate around the minimum. Note that since $\frac{\partial F}{\partial a_i} = \kappa(\mathbf{a})[a_i - f_i(\mathbf{a})]$, the gradient descent iteration above may be written as a damped fixed-point iteration[Hes03]

$$a_i^{(t+1)} = (1 - \alpha\kappa(\mathbf{a})) a_i + \alpha\kappa(\mathbf{a}) f_i(\mathbf{a}), \quad (2.91)$$

with damping parameter $\eta = \alpha\kappa(\mathbf{a})$. Newton's method, on the other hand, prescribes

$$a_i^{(t+1)} = a_i^{(t)} - \alpha(t) \sum_j [H^{-1}]_{ij} \left. \frac{\partial F}{\partial a_j} \right|_{\mathbf{a}^{(t)}} \quad (2.92)$$

where H is the Hessian computed at $\mathbf{a}^{(t)}$, i.e. $H_{ij} = \frac{\partial^2 F}{\partial a_i \partial a_j} \big|_{\mathbf{a}^{(t)}}$. Newton's method may be seen as minimizing a 2nd order Taylor expansion of the function around the minimum. Inverting the Hessian in the expression above is often computationally expensive, especially as the number of coefficients gets larger; in this case, the Hessian may be approximated, as it is done by the so-called quasi-Newton methods.

For appropriately chosen $\alpha(t)$, both algorithms are guaranteed to reach the minimum of convex functions; as we mentioned though, the free energies we are working with are not in general convex, and even though the algorithms may ultimately lead to a local minimum, they are not guaranteed to decrease the free energy at each iteration. For this purpose, a *double-loop*, or *convex-concave* procedure[Yui02], may be employed. The basic idea is that any function, and in particular the free energies we are working with, can be decomposed as the sum of two parts that are respectively convex and concave, $F(\mathbf{a}) = F_{\text{vex}}(\mathbf{a}) + F_{\text{cave}}(\mathbf{a})$; one then shows that solving the following equation for $\mathbf{a}^{(t+1)}$ is guaranteed to decrease $F(\mathbf{a})$ at each iteration

$$\nabla F_{\text{vex}}(\mathbf{a}^{(t+1)}) = -\nabla F_{\text{cave}}(\mathbf{a}^{(t)}). \quad (2.93)$$

If additionally there are K linear constraints $\phi_i \cdot \mathbf{a} = c_i$ we want to satisfy, the following equation can be used

$$\nabla F_{\text{vex}}(\mathbf{a}^{(t+1)}) = -\nabla F_{\text{cave}}(\mathbf{a}^{(t)}) - \sum_{i=1}^K \lambda_i \phi_i \quad (2.94)$$

so that we end up with two loops: an outer one, where we solve the equation above for $\mathbf{a}^{(t+1)}$, and an inner one, where the values of $\{\lambda_i\}$ that satisfy the constraints are to be determined (thus the double-loop denomination).

The Role of Statistical Physics

Determining the thermodynamic properties of a physical system in thermal equilibrium is a problem akin to those introduced in the first chapter, and may also be approached using probabilistic graphical models. In this chapter, we discuss the relationship between problems in statistical physics and problems of inference and optimization, and how techniques developed independently in one area may be useful in the other.

Many of the inference and optimization problems we have seen in chapter 1 are described by probability distributions of the form $P(\mathbf{x}|J) \propto \exp\{-\beta\mathcal{H}_J(\mathbf{x})\}$. In particular if the components of $\mathbf{x}$ are binary, $\mathbf{x} \in \{-1, +1\}^N$, the Hamiltonian $\mathcal{H}_J(\mathbf{x})$ can always be written as

$$\mathcal{H}_J(\mathbf{x}) = - \sum_{a=1}^M J_a \prod_{i \in \partial a} x_i. \quad (3.1)$$

given an appropriate factor graph $\mathcal{G}(V_N \cup V_F, E)$ that specifies the neighborhoods $\partial a = \{i \mid i \in V_N \wedge (i, a) \in E\}$. This form encompasses many classical models from statistical physics, from prototypical Ising model to models of disordered systems such as Edwards-Anderson's, Sherrington-Kirkpatrick's and the p -spin glass[Nis01; MM09]. Each $a = 1, \dots, M$ is associated to a set ∂a of spins interacting, with J_a modulating the strength of the interaction. For instance in the Ising model[Sal01]

$$\mathcal{H}_J(\mathbf{x}) = -J \sum_{(ij) \in \mathbb{Z}^2} x_i x_j, \quad (3.2)$$

all interactions have the same strength J , while in the more general Edwards-Anderson model

$$\mathcal{H}_J(\mathbf{x}) = - \sum_{(ij) \in \mathbb{Z}^2} J_{ij} x_i x_j, \quad (3.3)$$

the $\{J_{ij}\}$ are sampled from a certain distribution, usually plus-minus one or Gaussian. Whenever the coupling strengths are random variables, one says the system is disordered; also, since these models were introduced so as to model glasses, they go by the name of *spin glass* models[FH93].

The analytical treatment of models in lattices like the two above is hard in general, so physicists usually recur to mean-field models with no geometric structure[Zam10]. The two classes of mean-field models that have been most widely studied are the fully-connected models, where each spin interacts (weakly) with every other spin; and the diluted ones, where each spin interacts with a finite number of spins picked at random. An example is the Sherrington-Kirkpatrick (SK) model

$$\mathcal{H}_J(\mathbf{x}) = -\frac{1}{2N} \sum_{ij} J_{ij} x_i x_j, \quad (3.4)$$

which is the fully connected mean-field analog of the Edwards-Anderson model (3.3) – notice that in order for its energy to be extensive, the interactions need to be proportional to $\frac{1}{N}$. The model above has been studied in detail since its introduction [SK75] in 1975, and can be considered the fruit fly of disordered systems in equilibrium, for it is simple enough to be carefully analyzed in first approximation but also complicated enough to present all plethora of phenomena usually associated to spin glasses, such as the need of employing full replica symmetry breaking in order to obtain exact results.

Given the Hamiltonian $\mathcal{H}_J(\mathbf{x})$ that describes a physical system in thermal equilibrium, one would proceed by computing the partition function $\mathcal{Z}_J(\beta) = \sum_{\mathbf{x}} e^{-\beta \mathcal{H}_J(\mathbf{x})}$, from which the *free entropy density*¹

$$\phi_J(\beta) = \frac{1}{N} \log \mathcal{Z}_J(\beta), \quad (3.5)$$

is obtained; the derivatives of ϕ_J then yield, in the thermodynamic limit $N \rightarrow \infty$, all the thermodynamic properties of this system. Notice that this problem belongs to the same wide class of problems introduced in chapter 1 – computing thermodynamic properties is akin to computing the moments of the Gibbs-Boltzmann distribution, which can be obtained from the derivatives of the cumulant generating-function, in this context given by the free entropy.

We see thus that while many of the tools introduced in chapter 1 can be adapted to the context of statistical physics – for instance, factor graphs can be used to represent Hamiltonians (see Figure 3.1) – the opposite is also true: tools developed in Physics can be used to analyze this wider class of problems, and in particular problems of inference and optimization.

Figure 3.1: Factor graphs associated to typical models from statistical physics, for $P(\mathbf{x}|J) \propto \exp \{-\beta \mathcal{H}_J(\mathbf{x})\}$. Left: Ising model on a square lattice, $\mathcal{H}_J(\mathbf{x}) = -\sum_{(i,j) \in \mathbb{Z}^2} J_{ij} x_i x_j - \sum_i \theta_i x_i$. Right: p -spin model on a random lattice, with $p = 3$ and $\mathcal{H}_J(\mathbf{x}) = -\sum_{(i,j,k) \in E} J_{ijk} x_i x_j x_k$.

Up to this point, we have been concerned with the *single instance* scenario, where we want to compute the properties of $P(\mathbf{x}|J)$ given a sample $J = \{J_a\}$. In Physics, on the other hand, one is usually concerned with the *typical case* scenario, where given $P(J)$ we want to study the ensemble of distributions $P(\mathbf{x}|J)$ as N goes to infinity – for instance, we may want to determine what is the average over J of a certain property of $P(\mathbf{x}|J)$. As an example, consider the SK model (3.4) under an external field

$$P(\mathbf{x}|J; \beta, B) = \frac{1}{\mathcal{Z}_J(\beta, B)} e^{-\beta \mathcal{H}_J(\mathbf{x})}, \quad \mathcal{H}_J(\mathbf{x}) = -\frac{1}{2N} \sum_{ij} J_{ij} x_i x_j - B \sum_i x_i, \quad (3.6)$$

¹Equivalently, one could work with the free energy density $f_J(\beta) = -\frac{1}{\beta} \phi_J(\beta)$.

and say the property we are interested in is the mean of $\mathbf{x}$, i.e. the magnetization $m_J = \frac{1}{N} \sum_i \langle x_i \rangle_P = \frac{\partial \phi_J}{\partial (\beta B)}$. In order to determine the average of this and other quantities over J , it suffices to compute the average free entropy density

$$\bar{\phi}(\beta, B) = \lim_{N \rightarrow \infty} \frac{1}{N} \mathbb{E}_J \log \mathcal{Z}_J(\beta, B) = \lim_{N \rightarrow \infty} \frac{1}{N} \int \prod_{ij} [dJ_{ij} P(J_{ij})] \log \mathcal{Z}_J(\beta, B) \quad (3.7)$$

since $\mathbb{E}_J m_J = \mathbb{E}_J \frac{\partial \phi_J}{\partial (\beta B)} = \frac{\partial \bar{\phi}}{\partial (\beta B)}$. However, since in general it is not possible to obtain $\log \mathcal{Z}_J$ in closed-form for a given J , $\bar{\phi}$ may be hard to compute.

In the next section, we will discuss approximations that allow us to compute $\bar{\phi}$ either analytically or numerically; the two most prominent of these approximations, the replica and cavity methods, were initially developed in the context of statistical physics and later found great success in the study of problems in other areas. The application of these techniques to the study of inference and optimization problems is the subject of sections 3.2 and 3.3 respectively.

3.1 Disordered systems in equilibrium

Given any distribution

$$P(\mathbf{x}|J; \beta, \boldsymbol{\theta}) = \frac{1}{\mathcal{Z}_J(\beta, \boldsymbol{\theta})} e^{-\beta \mathcal{H}_J(\mathbf{x}; \boldsymbol{\theta})}, \quad (3.8)$$

we will denote by *dynamical* the $\mathbf{x}$ variables, and by *quenched* the $J \sim P(J)$ ones; β and any other parameters $\boldsymbol{\theta}$ of the Hamiltonian $\mathcal{H}_J(\mathbf{x}; \boldsymbol{\theta})$ are called *control parameters*, while derivatives of the average free entropy density $\bar{\phi}(\beta, \boldsymbol{\theta}) = \lim_{N \rightarrow \infty} \frac{1}{N} \mathbb{E}_J \log \mathcal{Z}_J(\beta, \boldsymbol{\theta})$ go by the name of *order parameters*. We are typically interested in evaluating the order parameters for a range of control parameters, as they allow us to distinguish between different thermodynamic phases of the system and draw its phase diagram.

3.1.1 Computing the average free entropy

Annealed approximation

The simplest way of approximating $\mathbb{E}_J \log \mathcal{Z}_J$ is by using the so-called *annealed approximation*

$$\mathbb{E}_J \log \mathcal{Z}_J \approx \log \mathbb{E}_J \mathcal{Z}_J, \quad (3.9)$$

which treats the dynamical and quenched variables on the same footing. The annealed denomination stands for models where the time scales are the same for both the disorder and the dynamical variables – in opposition to the quenched models we approach here, where the disorder evolves in a much slower time scale and may thus be considered fixed. While calculating $\mathbb{E}_J \mathcal{Z}_J$ is often a much easier problem, this approximation is usually quite inaccurate, as rare samples of $\mathcal{Z}_J$ will dominate the mean – one says that $\log \mathcal{Z}_J$ is a self-averaging quantity, since fluctuations around its mean are exponentially rare, while $\mathcal{Z}_J$ is not.

Even though $\log \mathbb{E}_J \mathcal{Z}_J$ provides a crude approximation, its calculation might be useful as it also gives a lower-bound on $\mathbb{E}_J \log \mathcal{Z}_J$ due to Jensen's inequality, so that

$$\mathbb{E}_J \log \mathcal{Z}_J \geq \log \mathbb{E}_J \mathcal{Z}_J. \quad (3.10)$$

The replica method

The replica method[Nis01] is based on the following identity

$$\mathbb{E}_J \log \mathcal{Z}_J = \lim_{n \rightarrow 0} \frac{\mathbb{E} \mathcal{Z}_J^n - 1}{n}, \quad (3.11)$$

sometimes called the *replica trick* in statistical physics, as the limit $n \rightarrow 0$ is taken even though n is assumed to be a positive integer. The replicated partition function

$$\mathbb{E}_J \mathcal{Z}_J^n = \mathbb{E}_J \prod_{\alpha=1}^n \sum_{\mathbf{x}^\alpha} e^{-\beta \mathcal{H}_J(\mathbf{x}^\alpha)} = \mathbb{E}_J \sum_{\mathbf{x}^1, \dots, \mathbf{x}^n} e^{-\beta \sum_{\alpha=1}^n \mathcal{H}_J(\mathbf{x}^\alpha)} \quad (3.12)$$

emulates n distinct interacting copies (or *replicas*) of the original system. After computing the average over J , order parameters $\mathbf{q}$ related to properties of these systems are introduced; for the SK model, for instance, two quantities naturally emerge throughout the calculations

$$m_\alpha = \frac{1}{N} \sum_{i=1}^N x_i^\alpha, \quad q_{\alpha\beta} = \frac{1}{N} \sum_i x_i^\alpha x_i^\beta, \quad (3.13)$$

while other quantities should be considered depending on the model being studied. After some work, one is able to write

$$\mathbb{E}_J \mathcal{Z}_J^n = \int d\mathbf{q} e^{Nn f(\mathbf{q})} \approx 1 + Nn f(\mathbf{q}^*), \quad (3.14)$$

where $\mathbf{q}^* = \arg \min_{\mathbf{q}} f(\mathbf{q})$ following from the integral being computed using the steepest descent approximation in the $N \rightarrow \infty$ limit. Thus

$$\bar{\phi} = \lim_{N \rightarrow \infty} \frac{1}{N} \mathbb{E}_J \log \mathcal{Z}_J \approx \text{extr}_{\mathbf{q}} f(\mathbf{q}), \quad (3.15)$$

and, as the order parameters $\mathbf{q}$, the function $f(\mathbf{q})$ depends on the particular model we are looking at. In order to solve for the saddle-point conditions

$$\left. \frac{\partial f}{\partial q_\alpha} \right|_{\mathbf{q}^*} = 0, \quad (3.16)$$

one usually recurs to approximations; the simplest one, known as replica-symmetric (RS), assumes $q_\alpha \equiv q_0$ for all α . It has been recently proved[GT02] that, for a series of models, a hierarchy of approximations proposed by Giorgio Parisi[Par83] in the 1980s leads to exact results; the calculations that follow from employing these approximations are however very intricate.

Cavity approach

Throughout this thesis, we will usually employ the cavity approach whenever averages over the disorder are to be computed. In its original formulation[MPV87], this method follows from computing the shift in the free energy due to removing the spin at the root of a tree (thus forming a *cavity*); we will however follow a modern derivation[MM09] that uses the belief propagation equations as a starting point, once again connecting approximate inference techniques to statistical physics.

Given the Hamiltonian (3.1) and $P(\mathbf{x}|J) = \frac{1}{\mathcal{Z}_J(\beta, \theta)} e^{-\beta \mathcal{H}_J(\mathbf{x}; \theta)}$, the belief propagation equations (1.9) read

$$\begin{aligned}
m_{a \rightarrow i}^{(t)}(x_i) &\propto \sum_{\mathbf{x}_{\partial a \setminus i}} e^{\beta J_a \prod_{i \in \partial a} x_i} \prod_{j \in \partial a \setminus i} \hat{m}_{j \rightarrow a}^{(t)}(x_j), \\
\hat{m}_{i \rightarrow a}^{(t+1)}(x_i) &\propto \prod_{b \in \partial i \setminus a} m_{b \rightarrow i}^{(t)}(x_i),
\end{aligned} \tag{3.17}$$

and since $x_i \in \{+1, -1\}$, we may use the change of variables introduced in (2.25) to write them as

$$\begin{aligned}
u_{a \rightarrow i}^{(t)} &= \frac{1}{\beta} \tanh^{-1} \left[\tanh(\beta J_a) \prod_{j \in \partial a \setminus i} \tanh(\beta h_{i \rightarrow a}^{(t)}) \right], \\
h_{i \rightarrow a}^{(t+1)} &= \sum_{b \in \partial i \setminus a} u_{b \rightarrow i}^{(t)}.
\end{aligned} \tag{3.18}$$

We have already seen in section 2.3 that, in order to perform the asymptotic analysis of a message-passing algorithm, it is useful to consider the messages as random variables, thus obtaining their distribution at each iteration by means of a Monte Carlo procedure (see Box 2.4). Beside considering h and u as random variables, we will now also take the BP equations (3.18) at their fixed points

$$\begin{aligned}
u &\stackrel{d}{=} \frac{1}{\beta} \tanh^{-1} \left[\tanh(\beta J) \prod_{i=1}^{p-1} \tanh(\beta h_i) \right], \\
h &\stackrel{d}{=} \sum_{a=1}^{c-1} u_a,
\end{aligned} \tag{3.19}$$

where as before, h_i and u_a represent i.i.d. copies of h and u , and $\stackrel{d}{=}$ indicates that the equalities are in distribution. Furthermore, c and p are random variables respectively distributed according to the degree distributions of the variable and factor nodes.

Box 3.1: Random factor graphs

Since on diluted spin models the spins involved in each interaction are picked at random, random graphs are usually employed in their description. We define an ensemble of factor graphs by the number of variable and factor nodes N and M and by their respective degree distributions P_v and P_f ; for instance, in cavity equation (3.19), $c \sim P_v$ and $p \sim P_f$. It is interesting to represent these distributions by their moment-generating functions

$$G_v(x) = \sum_{d=0}^{d_v^{\max}} P_v(d) x^d, \quad G_f(x) = \sum_{d=0}^{d_f^{\max}} P_f(d) x^d, \tag{3.20}$$

so that the mean degrees are given by $G'_v(1)$ and $G'_f(1)$ respectively and the following relation must be satisfied

$$n_{\text{edges}} \equiv N G'_v(1) = M G'_f(1) \quad \Rightarrow \quad \frac{M}{N} = \frac{G'_v(1)}{G'_f(1)}. \tag{3.21}$$

In general, averages over many instances of the factor graph are going to be denoted by $\mathbb{E}_{\mathcal{G}}$.

An approximation to the thermodynamical free energy is obtained by extremizing the variational free energies derived before. In particular, for the cavity approach one needs to use the Bethe free energy, that has been derived in Box 2.2 in the case of pairwise interactions and can be easily generalized to the current Hamiltonian (3.1).

$$F_{\text{Bethe}}(\mathbf{h}, \mathbf{u}) = - \sum_a \log \cosh(\beta J_a) + \sum_a (|\partial a| - 1) \log \left[1 + \tanh(\beta J_a) \prod_{i \in \partial a} \tanh(\beta h_{i \rightarrow a}) \right] - \sum_i \left[\log \cosh \left(\beta \sum_{a \in \partial i} u_{a \rightarrow i} \right) - \sum_{a \in \partial i} \log \cosh(\beta u_{a \rightarrow i}) \right]. \quad (3.22)$$

We will consider the large system limit and assume that the $\mathbf{h}$ and $\mathbf{u}$ that minimize this expression converge in distribution to the solution of (3.19) – this is called the replica-symmetric (RS) hypothesis, since it makes essentially the same assumptions as the replica method in the RS approximation. By computing the average of the expression above with respect to these distributions, we obtain

$$\bar{\phi}_{\text{RS}} = \frac{\bar{c}}{\bar{p}} \mathbb{E}_J \log \cosh(\beta J) - \frac{\bar{c}}{\bar{p}} \mathbb{E}_{J,h,p} (p-1) \log \left[1 + \tanh(\beta J) \prod_{i=1}^p \tanh(\beta h_i) \right] + \mathbb{E}_{u,c} \left[\log \cosh \left(\beta \sum_{a=1}^c u_a \right) - \sum_{a=1}^c \log \cosh(\beta u_a) \right]. \quad (3.23)$$

and the sign is flipped since we are now considering the free entropy instead of the free energy.

As an example, consider the cavity equation for the SK model (3.4), but replacing the $\frac{1}{N}$ term in the Hamiltonian by $\frac{1}{\sqrt{k}}$, where $k = c - 1$. For $p = 2$, u is a function of a single h , and we may choose to work only with h ; by keeping only terms that are linear in $\frac{1}{\sqrt{k}}$ we obtain

$$h \stackrel{d}{=} \frac{1}{\beta} \sum_{i=1}^{c-1} \tanh^{-1} \left[\tanh\left(\frac{\beta}{\sqrt{k}} J_i\right) \underbrace{\tanh(\beta h_i)}_{m_i} \right] \approx \frac{1}{\beta} \sum_{i=1}^k \tanh^{-1} \left[\frac{\beta}{\sqrt{k}} J_i m_i \right] \approx \frac{1}{\sqrt{k}} \sum_{i=1}^k J_i m_i, \quad (3.24)$$

or in terms of $m = \tanh(\beta h)$

$$m \stackrel{d}{=} \tanh \left(\frac{\beta}{\sqrt{k}} \sum_{i=1}^k J_i m_i \right), \quad (3.25)$$

Assume $J \sim \mathcal{N}(\mu_J, \sigma_J^2)$, and denote the mean and variance of m by $\bar{m}$ and $\bar{q}$ respectively; in the $N, k \rightarrow \infty$ limit

$$\frac{1}{\sqrt{k}} \sum_{i=1}^k J_i m_i \stackrel{d}{=} \mathcal{N}(\mu_J \bar{m}, \sigma_J^2 \bar{q}), \quad (3.26)$$

and thus

$$\begin{aligned} \bar{m} &= \frac{1}{\sqrt{2\pi}} \int dz e^{-\frac{z^2}{2}} \tanh(\beta \mu_J \bar{m} + \beta \sigma_J \sqrt{\bar{q}} z), \\ \bar{q} &= \frac{1}{\sqrt{2\pi}} \int dz e^{-\frac{z^2}{2}} \tanh^2(\beta \mu_J \bar{m} + \beta \sigma_J \sqrt{\bar{q}} z), \end{aligned} \quad (3.27)$$

which are precisely the saddle-point equations obtained by the replica method in the RS approximation.

Figure 3.2: Distribution of the cavity fields h described by (3.28) for $J \sim B_{\pm 1}(0.5)$ and $c = 3$ ($k = 2$), obtained by means of the population dynamics algorithm at different temperatures $T = \beta^{-1}$. The histograms were made using samples of size $N_{\text{pop}} = 1 \cdot 10^4$ averaged over $8 \cdot 10^3$ time steps.

More generally, in cases where the central limit theorem cannot be applied, the cavity equation needs to be solved numerically; consider the diluted analog of the SK model, for which the cavity equation reads

$$h \stackrel{d}{=} \frac{1}{\beta} \sum_{i=1}^k \tanh^{-1}[\tanh(\beta J_i) \tanh(\beta h_i)], \quad (3.28)$$

and in order to approximately determine $P(h)$, the population dynamics algorithm introduced in Box 2.4 can be used. More precisely, the algorithm provides us with a sample $\{h_i\}_{i=1, \dots, N_{\text{pop}}}$ of $P(h)$; averages of any function $\phi(h_1, h_2, \dots)$ of multiple copies of h can then be computed by picking elements from this sample at random, i.e.

$$\mathbb{E}_h \phi(h_1, h_2, \dots) \approx \sum_{a=1}^{N_{\text{samples}}} \phi(h_{i_a^1}, h_{i_a^2}, \dots), \quad (3.29)$$

with $i_a^k \sim \text{Uniform}(\{1, \dots, N_{\text{pop}}\})$. In particular, the average magnetization of the diluted SK model under consideration is given by

$$\bar{m} = \mathbb{E}_h \tanh(\beta h) \approx \sum_{i=1}^{N_{\text{pop}}} \tanh(\beta h_i). \quad (3.30)$$

Histograms of the samples of cavity fields obtained by the population dynamics algorithm are given in Figure 3.2 for $J \sim B_{\pm 1}(0.5)$, $c = 3$ ($k = 2$) and different temperatures $T = \beta^{-1}$.

3.1.2 The glassy phase

Thermodynamic phases are determined from the order parameters, which in turn can be obtained from derivatives of the free entropy. Following the cavity approach, these quantities become functions of the cavity fields and biases h and u . Consider the ferromagnetic Ising model, for which $P(J) = \delta(J - 1)$. In that case, we may assume $P(h) = \delta(h - h_0)$, and the cavity equation becomes

$$h_0 = \frac{k}{\beta} \tanh^{-1}[\tanh(\beta) \tanh(\beta h_0)] \equiv k u_0(h_0). \quad (3.31)$$

This equation has a single, *paramagnetic* solution at $h_0 = 0$ for $\tanh(\beta) < 1/k$ or

$$\beta < \beta_c \equiv \tanh^{-1}\left(\frac{1}{k}\right), \quad (3.32)$$

but for $\beta > \beta_c$, two other solutions appear. Furthermore, we can see that the $h_0 = 0$ fixed-point is not stable for $\beta > \beta_c$, since

$$\left. \frac{\partial u_0}{\partial h_0} \right|_{h_0=0} = k \frac{\tanh(\beta) (1 - \tanh^2(\beta h_0))}{1 - \tanh^2(\beta) \tanh^2(\beta h_0)} \Big|_{h_0=0} = k \tanh(\beta) > 1. \quad (3.33)$$

Alternatively, one could look at the ferromagnetic susceptibility

$$\chi = \frac{1}{N} \sum_{ij} (\langle x_i x_j \rangle - \langle x_i \rangle \langle x_j \rangle) \approx \sum_{r=0}^{\infty} k^r (\langle x_0 x_r \rangle - \langle x_0 \rangle \langle x_r \rangle), \quad (3.34)$$

where we assumed the factor graph to be locally tree-like in order to write the rightmost expression. Using the fluctuation-dissipation theorem

$$\langle x_0 x_r \rangle - \langle x_0 \rangle \langle x_r \rangle = \frac{\partial m_r}{\partial h_0} = \frac{\partial m_r}{\partial h_r} \frac{\partial h_r}{\partial h_{r-1}} \cdots \frac{\partial h_1}{\partial h_0} \sim \left(\frac{\partial u_0}{\partial h_0} \right)^r \quad (3.35)$$

so that $\chi \approx \sum_{r=0}^{\infty} \left(k \frac{\partial u_0}{\partial h_0} \right)^r$ only remains finite for $h_0 = 0$ if

$$k \frac{\partial u_0}{\partial h_0} \Big|_{h_0=0} = k \tanh(\beta) < 1 \quad \Rightarrow \quad \beta < \beta_c \equiv \tanh^{-1}\left(\frac{1}{k}\right). \quad (3.36)$$

Thus, for $\beta > \beta_c$, the paramagnetic solution is wrong, and one must take into account the other solutions at $h \neq 0$. In physical terms, what happens effectively at β_c is a phase transition, from para- to ferromagnetic.

Figure 3.3: Free energy density and squared magnetization for $J \sim \mathcal{B}_{\pm 1}(0.5)$ and $c = 3$ ($k = 2$), averaged over the cavity fields (3.37) using the samples yielded by the population dynamics algorithm. The phase transition from paramagnetic to spin glass happens at $T_c = 1.13$ and is signaled by a positive squared magnetization (the first moment of the overlap distribution); moreover, the paramagnetic free energy computed at $h = 0$ (dotted line) is not correct for $T < T_c$.

A thermodynamic phase that is distinctive of disordered systems, only appearing once quenched disorder is introduced, is the so-called *glassy phase*. This phase is characterized by a rugged free energy landscape containing multiple local minima, so that most dynamics ends up stuck in metastable

states for long periods of time and therefore are not ergodic. The replica-symmetric approximation to the free energy, obtained either through the replica or cavity methods, is known to be wrong at this phase.

We now consider $J \sim B_{\pm 1}(0.5) \equiv \frac{1}{2} \delta(J-1) + \frac{1}{2} \delta(J+1)$, and go back to cavity equation (3.28)

$$h = \sum_{i=1}^k u(J_i, h_i), \quad u(J, h) \equiv \frac{1}{\beta} \tanh^{-1} [\tanh(\beta J) \tanh(\beta h)]. \quad (3.37)$$

In order to study the stability of the paramagnetic solution in this case, we consider averages of $u(J, h)$ with respect to J and h ; more specifically, we introduce the spin glass susceptibility

$$\chi_{SG} = \frac{1}{N} \mathbb{E}_{J, \mathcal{G}} \sum_{ij} (\langle x_j x_i \rangle - \langle x_i \rangle \langle x_j \rangle)^2 \approx \sum_{r=0}^{\infty} k^r \mathbb{E}_{J, \mathcal{G}} \left(\frac{\partial \mathbb{E}_h u}{\partial \mathbb{E}_h h} \right)^{2r}. \quad (3.38)$$

As before, the paramagnetic solution is correct if χ_{SG} remains finite for $h = 0$, so that the stability condition reduces to

$$k \mathbb{E}_{J, \mathcal{G}} \left(\frac{\partial \mathbb{E}_h u}{\partial \mathbb{E}_h h} \right)^2 \Big|_{h=0} = k \mathbb{E}_{J, \mathcal{G}} \tanh^2(\beta J) < 1 \quad \Rightarrow \quad \beta < \beta_c \equiv \tanh^{-1} \left(\frac{1}{\sqrt{k}} \right). \quad (3.39)$$

In particular, in the $N, k \rightarrow \infty$ limit, we may use (3.25) and (3.26) to obtain

$$k \mathbb{E}_{J, \mathcal{G}} \left(\frac{\partial \mathbb{E}_m u}{\partial \mathbb{E}_m m} \right)^2 \Big|_{h=0} = \frac{\beta^2}{\sqrt{2\pi}} \int dz e^{-\frac{z^2}{2}} \operatorname{sech}^4(\beta \mu_J \bar{m} + \beta \sigma_J \sqrt{q} z) < 1,$$

which is the same expression obtained by de Almeida and Thouless for the fully-connected SK model using the replica method.

Another quantity useful to our analysis is the distribution of the overlaps between two configurations $\mathbf{x}^a$ and $\mathbf{x}^b$, $q = \frac{1}{N} \sum_i x_i^a x_i^b$. Under the replica-symmetric assumption, this distribution is assumed to be trivial, i.e. $P(q) = \delta(q - q_0)$. However, as the system undergoes a glassy transition, the true overlap distribution becomes more complicated. We might assume the distribution is bimodal, hypothesis that goes by the name of one-step replica-symmetry breaking (1RSB). In the full replica-symmetry breaking (FRSB) scenario, this distribution is continuous. We may thus define RS, 1RSB and FRSB phases, for regions of the diagram where these assumptions are correct.

The first and second (central) moments of the distribution of overlaps are $\langle q \rangle = \frac{1}{N} \mathbb{E}_{J, \mathcal{G}} m^2$ and $\operatorname{Var}(q) = \frac{1}{N} \chi_{SG}$. In order to identify a para-glass transition, it suffices to look at the first moment, since it is expected to be zero in the paramagnetic phase and positive in the spin glass phase (see Figure 3.3). In the general case, one would measure $\frac{1}{N} \chi_{SG}$, and if a positive value were to be obtained in the thermodynamic limit $N \rightarrow \infty$, we would have an indication that the RS hypothesis is not true. One would then have to rederive all the equations assuming a bimodal overlap distribution, i.e. using a 1RSB *ansatz*; this is a fairly intricate procedure that, for the cavity method, has only been accomplished in the last decade[MP01].

3.2 Planted ensemble and the Nishimori line

Given any latent variable model that gives rise to a distribution of the form

$$P(\mathbf{x}|\mathbf{y}) = \frac{1}{\mathcal{Z}(\mathbf{y})} \prod_{\mu=1}^M P(y_\mu | \mathbf{x}_{\partial \mu}) \prod_{i=1}^N P_0(x_i), \quad (3.40)$$

we may associate a Hamiltonian

$$\mathcal{H}_{\mathbf{y}}(\mathbf{x}) = - \sum_{a=1}^M \log P(y_\mu | \mathbf{x}_{\partial\mu}) - \sum_{i=1}^N \log P_0(x_i), \quad (3.41)$$

provided the factors are strictly positive for all y_μ, x_i . In this context, the observed variables $\mathbf{y}$ can be seen as quenched disorder; furthermore, we know from the model that the y_μ are functions of a specific configuration $\mathbf{x}_0$ (which we are trying to estimate) and the noise ξ . From this scenario emerges the concept of *planted disorder*[KZAC15] – quenched variables that are sampled from a very particular distribution, namely one that depends on a *planted* configuration. In other words, one picks a configuration $\mathbf{x}_0$ and generates the bonds in such a way that $\mathbf{x}_0$ is a typical configuration of the system

$$P(\mathbf{x}_0 | \mathbf{y}) \propto e^{-\beta \mathcal{H}_{\mathbf{y}}(\mathbf{x}_0)}. \quad (3.42)$$

Once the Hamiltonian is written down, our main interest is in computing the average of the loss function $\mathbb{E}_{\mathbf{x}_0, \mathbf{y}} \ell(\hat{\mathbf{x}}(\mathbf{y}), \mathbf{x}_0)$ given the chosen estimator $\hat{\mathbf{x}}(\mathbf{y})$. Again, since $\mathbf{y}$ is a function of $\mathbf{x}_0$ and ξ , this quantity may be written as

$$\mathbb{E}_{\mathbf{x}_0, \xi} \ell(\hat{\mathbf{x}}(\mathbf{x}_0, \xi), \mathbf{x}_0). \quad (3.43)$$

This is the primary order parameter to be considered when studying inference problems, i.e. phase transitions are identified by sudden changes in this quantity. These ideas shall become clearer with the examples we present next.

Example: reputation systems

Consider the latent variable model (1.37)

$$\begin{cases} J_{ij} = \xi_{ij} r_i r_j, & \forall (i, j) \in \Omega, \\ \xi_{ij} \sim \text{B}_{\pm 1}(p) \equiv p \delta_{\xi_{ij}, +1} + (1-p) \delta_{\xi_{ij}, -1}, \\ r_i \sim \text{B}_{\pm 1}(q), & i = 1, \dots, N, \end{cases} \quad (3.44)$$

for $J_{ij}, r_i \in \{+1, -1\}$, which following Bayes' theorem yields the following posterior

$$P(\mathbf{s} | J) = \frac{1}{\mathcal{Z}(J)} \exp \left\{ \alpha_p \sum_{(i,j) \in \Omega} J_{ij} s_i s_j + \alpha_q \sum_i s_i \right\} \quad (3.45)$$

$$= \frac{1}{\mathcal{Z}(\mathbf{r}, \xi)} \exp \left\{ \alpha_p \sum_{(i,j) \in \Omega} \xi_{ij} \underbrace{r_i s_i}_{\sigma_i} r_j s_j + \alpha_q \sum_i s_i \right\} \quad (3.46)$$

$$= \frac{1}{\mathcal{Z}(\mathbf{r}, \xi)} \exp \left\{ \alpha_p \sum_{(i,j) \in \Omega} \xi_{ij} \sigma_i \sigma_j + \alpha_q \sum_i r_i \sigma_i \right\}, \quad (3.47)$$

where $\alpha_p = \frac{1}{2} \log \frac{p}{1-p}$. From the first to the second line we have used the generative model to write the observed variables J in terms of the reference configuration $\mathbf{r}$ and the noise ξ , while from the second to the third, we have introduced a new variable $\sigma = \mathbf{r} \cdot \mathbf{s}$ related to the overlap between $\mathbf{s}$ and the reference configuration. Estimators will depend on properties of

$P(s|J)$; the MPM (or symbol MAP) estimate for instance is given by $\hat{x}_i = \text{sgn}(\langle s_i \rangle)$, while its loss function reads

$$\ell_{\text{MPM}}(\hat{\mathbf{x}}, \mathbf{r}) = \frac{1}{2} \left(1 - \frac{1}{N} \sum_i r_i \text{sgn}(\langle s_i \rangle) \right) \Rightarrow \mathbb{E}_{\mathbf{r}, \xi, \mathcal{G}} \ell_{\text{MPM}} = \frac{1 - \mathbb{E}_{\mathbf{r}, \xi, \mathcal{G}} \text{sgn}(\langle \sigma \rangle)}{2}. \quad (3.48)$$

Example: ridge regression

As an introduction on what follows, it is useful to give an example with continuous variables. Take the problem of doing linear regression while using a Gaussian prior over the parameters

$$\begin{cases} \mathbf{y} = F\mathbf{x}_0 + \boldsymbol{\xi}, \\ \boldsymbol{\xi} \sim \mathcal{N}(0, \Delta \mathbb{1}_M), \\ \mathbf{x}_0 \sim \mathcal{N}(0, \sigma^2 \mathbb{1}_N). \end{cases} \quad (3.49)$$

for $\mathbf{y} \in \mathbb{R}^M$, $\mathbf{x}_0 \in \mathbb{R}^N$ and where $F \in \mathbb{R}^{N \times P}$ is given. The posterior distribution in this case reads

$$\begin{aligned} P(\mathbf{x}|\mathbf{y}) &= \frac{1}{\mathcal{Z}(\mathbf{y})} \exp \left\{ -\frac{1}{2\Delta} \|\mathbf{y} - F\mathbf{x}\|_2^2 - \frac{1}{2\sigma^2} \|\mathbf{x}\|_2^2 \right\} \\ &= \frac{1}{\mathcal{Z}(\mathbf{x}_0, \boldsymbol{\xi})} \exp \left\{ -\frac{1}{2\Delta} \|\boldsymbol{\xi} - F(\underbrace{\mathbf{x} - \mathbf{x}_0}_{\boldsymbol{\delta}})\|_2^2 - \frac{1}{2\sigma^2} \|\mathbf{x}\|_2^2 \right\} \\ &= \frac{1}{\mathcal{Z}(\mathbf{x}_0, \boldsymbol{\xi})} \exp \left\{ -\frac{1}{2\Delta} \|\boldsymbol{\xi} - F\boldsymbol{\delta}\|_2^2 - \frac{1}{2\sigma^2} \|\boldsymbol{\delta} + \mathbf{x}_0\|_2^2 \right\}, \end{aligned} \quad (3.50)$$

and for the MMSE estimator, $\hat{\mathbf{x}} = \langle \mathbf{x} \rangle$

$$\ell_{\text{MMSE}}(\hat{\mathbf{x}}, \mathbf{x}_0) = \|\langle \mathbf{x} \rangle - \mathbf{x}_0\|_2^2 \Rightarrow \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\xi}} \ell_{\text{MMSE}} = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\xi}} \|\langle \boldsymbol{\delta} \rangle\|^2. \quad (3.51)$$

In this case $\langle \mathbf{x} \rangle$ may be written in closed-form (2.58), as does the mean-squared error

$$\begin{aligned} \ell_{\text{MMSE}} &\stackrel{d}{=} \left\| \left[\sqrt{\frac{\sigma^2}{\Delta}} F^T F + \sqrt{\frac{\Delta}{\sigma^2}} \mathbb{1}_N \right]^{-1} \left(\sqrt{\frac{\sigma^2}{\Delta}} F^T \boldsymbol{\xi} - \sqrt{\frac{\Delta}{\sigma^2}} \mathbf{x}_0 \right) \right\|_2^2 \\ &\approx \frac{\sigma^2}{\Delta} \left\| \mathcal{W}_N^{-1} \left(\frac{1}{N} \mathbb{1}_N, M \right) \mathcal{N}(0, (\alpha\sigma^2 + \Delta) \mathbb{1}_N) \right\|_2^2. \end{aligned} \quad (3.52)$$

where we have assumed $\Delta \ll \sigma^2$.

While in the last example the average of the loss function could in principle be obtained in closed-form, in the more general case this is not possible, since the estimator itself often has no analytic expression. In these cases, employing the replica and cavity methods might be useful[KZAC15].

When the analysis is performed assuming the values of the parameters are the same used to generate the quenched variables, then one says to be working in the *Nishimori condition*[Iba99]. For

instance, in the first example above we assume that α_p and α_q are computed using the same values of p and q that have been used to generate the quenched variables, $\xi_{ij} \sim B_{\pm 1}(p)$ and $r_i \sim B_{\pm 1}(q)$. The Nishimori condition simplifies a number of calculations; average quantities may be in general obtained by means of annealed averages, and for any function f of two configurations the following property holds

$$\mathbb{E}_{\xi, x_0} \langle f(\mathbf{x}_a, \mathbf{x}_b) \rangle = \mathbb{E}_{\xi, x_0} \langle f(\mathbf{x}_0, \mathbf{x}) \rangle. \quad (3.53)$$

In particular, the average overlap between two replicas $\mathbf{x}_a$ and $\mathbf{x}_b$, $\bar{q} = \mathbb{E}_{\xi, x_0} \langle \mathbf{x}_a \cdot \mathbf{x}_b \rangle$ is the same as the average overlap between any configuration $\mathbf{x}$ and the planted one $\bar{m} = \mathbf{x}_0$, $\mathbb{E}_{\xi, x_0} \langle \mathbf{x}_0 \cdot \mathbf{x} \rangle$, so that the overlap distribution is trivial and the system never undergoes a glassy transition[NS01].

3.3 Looking for ground states

Statistical physics also brings some insight into combinatorial optimization problems[Zde09]. As in section 1.3, we associate the cost function $C_I(\mathbf{x})$ to a Gibbs-Boltzmann distribution

$$P(\mathbf{x}|I) = \frac{1}{Z_I(\beta)} e^{-\beta C_I(\mathbf{x})} \quad (3.54)$$

and we are interested in the $\beta \rightarrow \infty$ limit. The subscript is used to indicate that we are focusing on a particular instance I of an optimization problem; for instance, in the K -XOR-SAT problem

$$\prod_{i \in \partial a} x_i = b_a, \quad a = 1, \dots, M, \quad \Rightarrow \quad C_I(\mathbf{x}) = \frac{1}{2} \sum_{a=1}^M \left(1 - b_a \prod_{i \in \partial a} x_i \right), \quad (3.55)$$

an instance is defined by a vector $\mathbf{b} \in \{+1, -1\}^M$ and by a factor graph $\mathcal{G}$ that specifies the neighborhoods $\partial a = \{i_1^a, i_2^a, \dots, i_K^a\}$, i.e. on which variables each constraint depend.

The cost function may be seen as the Hamiltonian of a physical system, and we are interested in answering a number of questions about what happens in the zero-temperature limit, in particular

- over which configurations the measure $P(\mathbf{x}|I)$ concentrates in this limit?
- what is the ground-state energy? is the ground-state frustrated, i.e. is its energy larger than zero?
- what is the ground-state entropy, that is, over how many configurations does $P(\mathbf{x}|I)$ concentrates?

Note that these are related to the different tasks we want to perform when treating combinatorial optimization problems, specifically: optimization, satisfiability and counting.

In the spirit of statistical physics, we shall consider I picked uniformly at random from a chosen set of instances and try to explain what happens in average in the $N \rightarrow \infty$ limit, for instance what is the average of the ground-state energy over all instances. In particular, if the problem at hand is a CSP and the average of the ground-state energy is zero, then we know that all instances from that set are satisfiable.

The space of solutions for constraint-satisfaction problems has a very rich structure that can be studied within this framework. Both in K -SAT and K -XOR-SAT, we can define $\alpha = \frac{M}{N}$ as the ratio between the number of constraints/clauses and the number of variables, and compute averages over

all instances with a given α . For α small both problems are satisfiable – i.e. the ground-state is not frustrated – but as its value increases, different kinds of phase transitions occur. At a first moment, the system undergoes a glassy transition, and as the RS solution loses its validity, the marginals provided by belief propagation are no longer correct. However, one is able to obtain good results even in the 1RSB regime by employing the closely related survey propagation algorithm.

As the optimization problem we treat in this thesis (chapter 5) is not a CSP, it is not of our interest to delve into specifics of the structure associated to this class of problems. Instead, we shall focus on the more general problem of computing the average ground-state energy density ϵ_0 for a given model. In the $\beta \rightarrow \infty$ limit, the cavity equations become

$$\begin{aligned} u &\stackrel{d}{=} \operatorname{sgn} \left(J \prod_{i=1}^{p-1} h_i \right) \min (|J|, \{|h_i|\}_{i=1, \dots, p-1}), \\ h &\stackrel{d}{=} \sum_{a=1}^{c-1} u_a. \end{aligned} \tag{3.56}$$

Alternatively, these equations may be obtained from the distributional fixed-point of the max-product iteration (1.57). Since $\epsilon_0 = \lim_{\beta \rightarrow \infty} -\frac{1}{\beta N} \log \mathcal{Z}(\beta)$, the RS cavity prediction of this quantity is easily obtained by taking the zero-temperature limit of expression (3.23)

$$\begin{aligned} \epsilon_0^{\text{RS}} &= -\frac{\bar{c}}{\bar{p}} \mathbb{E}_J |J| + 2 \frac{\bar{c}}{\bar{p}} \mathbb{E}_{J, h, p} (p-1) \min (|J|, \{|h_i|\}_{i=1, \dots, p}) \mathbb{I} \left[J \prod_{i=1}^p h_i < 0 \right] \\ &\quad - \mathbb{E}_{u, c} \left\{ \left| \sum_{a=1}^c u_a \right| - \sum_{a=1}^c |u_a| \right\}. \end{aligned} \tag{3.57}$$

After determining $P(h)$ and $P(u)$ using the population dynamics algorithm, this expression can be easily evaluated. We may then validate the performance of a given algorithm by comparing ϵ_0^{RS} to the cost of the solutions this algorithm provides for single instances, since expects the average cost to tend to ϵ_0^{RS} as $N \rightarrow \infty$.

Part II

Applications

Compressed Sensing

Consider the following system of linear equations

$$\mathbf{y} = F\mathbf{x}, \quad (4.1)$$

for a given vector $\mathbf{y} \in \mathbb{R}^M$ and a matrix $F \in \mathbb{R}^{M \times N}$; we denote the ratio of the number of rows to the number of columns in F by $\alpha = M/N$. It is known that, if $M = N$ and F is full-rank, then the system has a unique solution $\mathbf{x}^* \in \mathbb{R}^N$.

If $M > N$, then the system is said to be overdetermined, i.e. there are more equations than unknowns, so the system usually has no solution and is thus deemed inconsistent. More precisely, a system of linear equations has no solution whenever the rank of the augmented matrix $[F|\mathbf{y}]$ is greater than the rank of the coefficient matrix F , following Rouché-Cappelli's theorem[Str03]. If the coefficients are for instance picked at random, then most of the rows of the coefficient matrix will be linearly independent, and the system will be inconsistent with high probability. In this case, an approximate solution may be obtained by means of the ordinary least-squares estimator

$$\hat{\mathbf{x}}^{\text{OLS}}(F, \mathbf{y}) = \arg \min_{\mathbf{x}} \|\mathbf{y} - F\mathbf{x}\|_2^2 = (F^T F)^{-1} F^T \mathbf{y}. \quad (4.2)$$

On the other hand, if $M < N$, the system is underdetermined and usually has infinite solutions, i.e. it is ill-posed; in particular, the set of solutions form an affine subspace of $\mathbb{R}^N$ of dimension $N - \text{rank}(F)$. Also, note that the ordinary least-squares estimator cannot be used in this case, since $F^T F$ is rank-deficient and thus singular. We therefore need to specify another criterion in order to have a unique solution; one that is commonly used is to pick the solution with the least ℓ_2 norm, that is

$$\hat{\mathbf{x}}^{\text{LN}}(F, \mathbf{y}) = \begin{cases} \arg \min_{\mathbf{x}} \|\mathbf{x}\|_2^2, \\ \text{s.t. } \mathbf{y} = F\mathbf{x}. \end{cases} = F^T (F F^T)^{-1} \mathbf{y}. \quad (4.3)$$

These two estimators are summarized by the Moore-Penrose pseudoinverse of F

$$F^+ = \begin{cases} (F^T F)^{-1} F^T, & \text{if } M \geq N, \\ F^T (F F^T)^{-1}, & \text{if } M < N. \end{cases} \Rightarrow F^+ \mathbf{y} = \begin{cases} \hat{\mathbf{x}}^{\text{OLS}}(F, \mathbf{y}), & \text{if } M \geq N, \\ \hat{\mathbf{x}}^{\text{LN}}(F, \mathbf{y}), & \text{if } M < N. \end{cases} \quad (4.4)$$

An alternative way of obtaining a solution to an underdetermined linear system is by introducing a penalty or regularization term in the least-squares estimation[BV04], for instance

$$\begin{aligned}
\hat{\mathbf{x}}_{\lambda}^{\text{ridge}}(F, \mathbf{y}) &= \arg \min_{\mathbf{x}} \|\mathbf{y} - F\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 \\
&= (F^T F + \lambda \mathbb{1}_N)^{-1} F^T \mathbf{y} \\
&= F^T (F F^T + \lambda \mathbb{1}_N)^{-1} \mathbf{y},
\end{aligned} \tag{4.5}$$

known as Tikhonov regularization or ridge regression. Note that

$$\lim_{\lambda \rightarrow 0^+} \hat{\mathbf{x}}_{\lambda}^{\text{ridge}}(F, \mathbf{y}) = F^+ \mathbf{y}. \tag{4.6}$$

4.1 Bayesian approach to linear systems

Suppose that, given a matrix $F \in \mathbb{R}^{M \times N}$ and a vector $\mathbf{x}_0 \in \mathbb{R}^N$, the vector $\mathbf{y}$ in hands has actually been generated by means of $\mathbf{y} = F\mathbf{x}_0$. We are interested in determining under which conditions the *planted* vector $\mathbf{x}_0$ may be recovered.

When $M \geq N$, the system has the planted vector as its unique solution, and the OLS estimator (4.2) should lead to it; for $M < N$ however, there is no reason to believe that for instance the least-norm solution $\hat{\mathbf{x}}^{\text{LN}}(F, \mathbf{y})$ is the same as $\mathbf{x}_0$. Remarkably, as we will see, some estimators allow one to recover $\mathbf{x}_0$ even when $M < N$, under specific conditions. Such procedure is effectively one of decompression, since a vector of dimension N is recovered from a vector of dimension $M < N$.

Recovering the planted vector is essentially a problem of inference. Consider the following latent variable model, or LVM

$$\begin{cases} \mathbf{y} = F\mathbf{x}_0, \\ \mathbf{x}_0 \sim \mathcal{N}(0, \sigma^2 \mathbb{1}_N), \end{cases} \tag{4.7}$$

that through Bayes theorem leads to the following posterior distribution

$$P(\mathbf{x}|F, \mathbf{y}) = \frac{1}{\mathcal{Z}(F, \mathbf{y})} \delta(\mathbf{y} - F\mathbf{x}) \prod_{i=1}^N e^{-\frac{1}{2\sigma^2} x_i^2}. \tag{4.8}$$

The hard constraint $\mathbf{y} = F\mathbf{x}$ can be relaxed – we may assume for instance that there is some noise affecting the measurement process

$$\begin{cases} \mathbf{y} = F\mathbf{x}_0 + \boldsymbol{\xi}, \\ \boldsymbol{\xi} \sim \mathcal{N}(0, \Delta \mathbb{1}_M), \\ \mathbf{x}_0 \sim \mathcal{N}(0, \sigma^2 \mathbb{1}_N), \end{cases} \tag{4.9}$$

and the posterior now reads

$$P(\mathbf{x}|F, \mathbf{y}) = \frac{1}{\mathcal{Z}(F, \mathbf{y})} \exp \left\{ -\frac{1}{2\Delta} \|\mathbf{y} - F\mathbf{x}\|_2^2 - \frac{1}{2\sigma^2} \|\mathbf{x}\|_2^2 \right\}, \tag{4.10}$$

Successfully recovering $\mathbf{x}_0$ means, in this context, having an estimate $\hat{\mathbf{x}}$ that is close to $\mathbf{x}_0$ in some desired metric – the particular choice of metric will determine the estimator that should be used (see section 1.2). Since $P(\mathbf{x}|F, \mathbf{y})$ is a multivariate normal distribution, its mode and mean coincide so that both the MAP and MMSE estimates yield

$$\hat{\mathbf{x}}^{\text{MMSE}}(F, \mathbf{y}) = \hat{\mathbf{x}}^{\text{MAP}}(F, \mathbf{y}) = \left(F^T F + \frac{\Delta}{\sigma^2} \mathbb{1}_N \right)^{-1} F^T \mathbf{y} = \hat{\mathbf{x}}_{\Delta/\sigma^2}^{\text{ridge}}(F, \mathbf{y}). \tag{4.11}$$

This is precisely the quantity computed in Box 2.3, and as we have shown, the estimator fails to obtain the planted solution when $\alpha < 1$, i.e. the mean-squared error (MSE)

$$\ell_{\text{MMSE}} = \frac{1}{N} \|\hat{\mathbf{x}}(F, \mathbf{y}) - \mathbf{x}_0\|_2^2, \quad (4.12)$$

is typically $o(\Delta)$ for $\alpha \equiv \frac{M}{N} > 1$, while for $\alpha < 1$ it is $o(1)$. We will next see under which conditions the vector $\mathbf{x}_0$ can be recovered when $\alpha < 1$, that is, when the system has less equations than variables.

4.1.1 Linear systems with sparse solutions

We now focus on recovering planted vectors $\mathbf{x}_0$ that are *sparse*. Let us define the ℓ_0 (pseudo-)norm of $\mathbf{x}$

$$\|\mathbf{x}\|_0 = \sum_i \mathbb{I}[x_i \neq 0], \quad (4.13)$$

as the number of non-zero components in the vector. Furthermore, for a vector $\mathbf{x}$ with $\|\mathbf{x}\|_0 = K$, we define the sparsity coefficient $\rho = \frac{K}{N}$, and say $\mathbf{x}$ is sparse whenever $\rho \ll 1$.

The planted vector is now assumed to be sampled from the following distribution

$$P_0(\mathbf{x}) = \prod_{i=1}^N [\rho \mathcal{N}(x_i; \mu, \sigma^2) + (1 - \rho) \delta(x_i)] \quad (4.14)$$

that is, each of its components is a normal random variable with probability ρ , and zero with probability $1 - \rho$; (4.14) is known as the Bernoulli-Gaussian or Gauss-Bernoulli distribution. The LVM can now be written as

$$\begin{cases} \mathbf{y} = F\mathbf{x}_0 + \boldsymbol{\xi}, \\ \boldsymbol{\xi} \sim \mathcal{N}(0, \Delta \mathbb{1}_M), \\ x_{0i} \sim \rho \mathcal{N}(x_i; \mu, \sigma^2) + (1 - \rho) \delta(x_i), \quad i = 1, \dots, N, \end{cases} \quad (4.15)$$

and the associated posterior

$$P(\mathbf{x}|F, \mathbf{y}) = \frac{1}{\mathcal{Z}(F, \mathbf{y})} e^{-\frac{1}{2\Delta} \|\mathbf{y} - F\mathbf{x}\|_2^2} \prod_{i=1}^N [\rho e^{-\frac{1}{2\sigma^2} (x_i - \mu)^2} + (1 - \rho) \delta(x_i)]. \quad (4.16)$$

The two control parameters we are primarily concerned about are the sparsity $\rho = \frac{K}{N}$ (the fraction of non-zero coefficients) and the sub-sampling rate $\alpha = \frac{M}{N}$ (the ratio between the numbers of rows and columns); in particular, we are interested in what happens when both ρ and α are much smaller than 1.

Convex surrogates

Since computing Bayesian estimators is intractable in this case – e.g. because marginals cannot be obtained in closed-form – one might choose to ignore the correct model and employ estimators on an empirical basis. We might look for instance for the sparsest solution of the linear system

$$\hat{\mathbf{x}}^{\text{sparse}}(F, \mathbf{y}) = \begin{cases} \arg \min_{\mathbf{x}} & \|\mathbf{x}\|_0, \\ \text{s.t.} & \mathbf{y} = F\mathbf{x}, \end{cases} \quad (4.17)$$

$$M \left\{ \left[\begin{array}{c} \text{ } \\ \text{ } \\ \text{ } \end{array} \right] = \underbrace{\left[\begin{array}{c} \text{ } \\ \text{ } \\ \text{ } \end{array} \right]}_N \underbrace{\left[\begin{array}{c} \text{ } \\ \text{ } \\ \text{ } \end{array} \right]}_K \left[\begin{array}{c} \text{ } \\ 0 \\ 0 \\ 0 \\ \vdots \end{array} \right] \right\} \begin{matrix} K \\ N \end{matrix}$$

Figure 4.1: In order to find the sparsest solution of an underdetermined linear system (4.17), one needs to check for each $K = 1, 2, \dots, M$ whether any of the K -combinations of all the N columns form a consistent system (note that these systems are overdetermined).

This problem however is also intractable, as for each $K < M$, one needs to check whether there is a K -combination of the N columns that makes the system consistent (see figure 4.1). One may thus replace the ℓ_0 by the ℓ_1 norm

$$\hat{\mathbf{x}}^{\text{bp}}(F, \mathbf{y}) = \begin{cases} \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1, \\ \text{s.t. } \mathbf{y} = F\mathbf{x}, \end{cases} \quad (4.18)$$

and hope that it recovers the planted vector in a certain regime. This procedure is known as *basis pursuit*[CDS98], and since it consists in minimizing a convex function subject to linear constraints, it can be done in polynomial time in N . In order to account for noise in the measurements, the following Lagrangian relaxation of (4.18) can be used

$$\hat{\mathbf{x}}_{\lambda}^{\text{lasso}}(F, \mathbf{y}) = \arg \min_{\mathbf{x}} \|\mathbf{y} - F\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1, \quad (4.19)$$

known as basis pursuit denoising (BPDN) or least absolute shrinkage and selection operator[Tib96] (LASSO). In Figure 4.2 we solve the problem above for instances generated using LVM (4.15) and different values of $\rho = \frac{K}{N}$ and $\alpha = \frac{M}{N}$; note that in some cases the recovery is actually successful, as the estimate $\hat{\mathbf{x}}_{\lambda}^{\text{lasso}}(F, \mathbf{y})$ gets very close to the planted signal $\mathbf{x}$.

Box 4.1: On the origins of the name *compressed sensing* – applications to signal processing

The problem of finding sparse solutions to linear systems is of interest to many different fields. While the LASSO algorithm has been developed by machine learning enthusiasts, basis pursuit had its origins within the signal processing community, which also motivated many of the accomplishments of the past decade. We briefly describe the importance of the problem to this particular field[CW08].

Given a discrete signal $\tilde{\mathbf{x}} \in \mathbb{R}^N$, one would like to be able to reconstruct $\tilde{\mathbf{x}}$ by correlating it to a small set of waveforms ϕ_k

$$y_k = \phi_k \cdot \tilde{\mathbf{x}}, \quad i = 1, \dots, M, \quad (4.20)$$

in other words, to *sense* or *sample* the signal by means of sensing waveforms $\Phi = [\phi_1 \ \phi_2 \ \dots \ \phi_M]$.

We may choose to represent the signal $\tilde{\mathbf{x}}$ in any basis Ψ , so that $\tilde{\mathbf{x}} = \Psi\mathbf{x}$, and $\mathbf{y} = (\Phi\Psi)\mathbf{x} \equiv F\mathbf{x}$. In particular, it is known that some choices of basis lead almost always to sparse or compressible representations, that is, for these choices a considerable fraction of the components of $\mathbf{x}$ are zero or close to zero – for instance, expansions in wavelet series usually have this property[Mal08].

Figure 4.2: Left: mean-squared error (MSE) provided by the LASSO estimator (4.19) for many values of λ . A $\mathbf{x}$ vector of size $N = 1024$ is generated with $K = \rho N$ non-zero entries sampled from the standard Gaussian distribution, $\rho = 0.2$; for each of the curves, a new matrix $F \in \mathbb{R}^{M \times N}$, $M = \alpha N$, is created using $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$, and finally $\mathbf{y} \sim \mathcal{N}(F\mathbf{x}, \Delta \mathbb{1}_M)$, $\Delta = 10^{-8}$, is computed. The value reported is that of $N^{-1} \|\hat{\mathbf{x}}_{\lambda}^{\text{lasso}}(F, \mathbf{y}) - \mathbf{x}\|_2^2$ – the estimate $\hat{\mathbf{x}}_{\lambda}^{\text{lasso}}$ was obtained using the SpaSM toolbox for MATLAB[SCLE12]. Right: minimum MSE obtained among all λ for each value of α , and three different values of ρ ; note that a transition occurs, and the algorithm is not able to recover the planted vector as $\alpha = \frac{M}{N}$ gets smaller.

In Figure 4.3, we show how an image remains practically unchanged if we neglect the coefficients of the wavelet expansion that are close to zero – thus obtaining a sparse vector $\mathbf{x}$.

The question then is how many measurements M must be performed in order to reconstruct $\tilde{\mathbf{x}}$, and what properties the sensing waveforms Φ must have with respect to the basis Ψ . As we will see next, this question has been recently answered; in fact, if Φ and Ψ possess certain properties, the number of measurements necessary is relatively small, so that the sensing is said to be *compressed* or *compressive*.

Formal guarantees

From a linear algebra point of view, being able to recover the planted vector means the system has a unique solution among the class of vectors under consideration. Disregarding for a moment the Bernoulli-Gaussian model (4.14), let us consider the class of vectors Σ_K that have exactly K non-zero components; we then want to know under which conditions two distinct vectors $\mathbf{x}_1, \mathbf{x}_2 \in \Sigma_K$ will be uniquely identifiable by their projections $\mathbf{y}_1 = F\mathbf{x}_1$ and $\mathbf{y}_2 = F\mathbf{x}_2$. Moreover, we want to know how good in identifying these signals are the estimators we have introduced.

Given a vector $\mathbf{x} \in \Sigma_K$ and a matrix F that satisfies certain requirements, there are guarantees that some estimators will always be able to recover $\mathbf{x}$ from $\mathbf{y} = F\mathbf{x}$ (or something close to $\mathbf{x}$ whenever noise is present). Define the coherence of a matrix F as

$$\mu(F) = \max_{i \neq j} \frac{|\mathbf{F}_i^T \cdot \mathbf{F}_j^T|}{\|\mathbf{F}_i^T\|_2 \|\mathbf{F}_j^T\|_2}, \quad (4.21)$$

Figure 4.3: Sparse representations using wavelet expansions. The original figure is displayed in the upper left, while the non-zero coefficients of the correspondent wavelet transform are displayed in the bottom left. For the remaining columns, we threshold the smallest coefficients of the transform so that only $K = \rho N$ coefficients are different from zero, and then perform the inverse transform. The coefficients after the thresholding are displayed in the bottom, and the *difference* between the resulting and the original images in the top. Notice that we can achieve small values of ρ without significantly altering the image.

where F_i^T is the i -th row of F^T (or i -th column of F); then if

$$K < \frac{1}{2} \left(1 + \frac{1}{\mu(F)} \right), \quad (4.22)$$

the vector $\mathbf{x}$ is guaranteed to be the sparsest solution of the system [DE03], and is thus recoverable by means of estimator (4.17).

An alternative but equivalent requirement that is more suited for the noisy case is that the matrix must have the *restricted-isometry property* [CRT04], or RIP. A matrix F is said to have the K -RIP if there exists $\delta_K \in \mathbb{R}^+$ such that

$$(1 - \delta_K) \|\mathbf{x}\|^2 \leq \|F_K \mathbf{x}\|^2 \leq (1 + \delta_K) \|\mathbf{x}\|^2, \quad \forall \mathbf{x} \in \mathbb{R}^K, \quad (4.23)$$

for any K -combination $F_K \in \mathbb{R}^{M \times K}$ of the N columns of F – in other words, if every subset of K columns of F is distance-preserving. In particular, let δ_K^* be the smallest value of δ_K such that the condition above is satisfied, and assume that $\delta_K^* < 1$; if $\mathbf{y} = F\mathbf{x} + \boldsymbol{\xi}$, with $\mathbf{x} \in \Sigma_K$ and $\|\boldsymbol{\xi}\|_2 \leq \varepsilon$, then the following inequality holds [DET06]

$$\|\hat{\mathbf{x}}^{\text{sparse}}(F, \mathbf{y}) - \mathbf{x}\|_2^2 \leq \frac{\varepsilon^2}{1 - \delta_K^*}. \quad (4.24)$$

Even though computing δ_K^* is intractable in general, upper bounds can be used[DE11] – it is known for instance[CRT04] that $\delta_K^* \leq (K - 1) \mu(F)$. Furthermore, many random matrices are known to have the RIP with high probability for certain sets of parameters; Gaussian matrices with

$$F_{\mu i} \sim \mathcal{N}\left(0, \frac{1}{N}\right), \quad (4.25)$$

can be shown to be K -RIP with high probability if $M \geq cK \log(N/K)$, with c a small constant[BDDW08].

Figure 4.4: *Donoho-Tanner transition:* above the diagonal line (green and yellow regions), a planted vector $x \in \mathbb{R}^N$ with $K = \rho N$ non-zero elements can always be recovered as the sparsest solution of the system (4.17), provided the coefficient matrix $F \in \mathbb{R}^{M \times N}$, $M = \alpha N$, is K -RIP. Additionally, in the green region, minimizing the ℓ_1 norm, i.e. employing the basis pursuit procedure (4.18) is also guaranteed to yield the correct result with much less computational cost. The line between the green and yellow regions is known as the Donoho-Tanner transition, and has been determined using both combinatorial arguments[DT05] and the replica method[KWT09] which is briefly discussed below (see (4.33)). In the red region sparse solutions are not unique and thus cannot be recovered. These results are independent of the distribution of the non-zero coefficients; we assume however that the measurements are noiseless, i.e. $\Delta = 0$.

Still, since computing $\hat{x}^{\text{sparse}}$ is computationally hard, it would be nice to have guarantees also for $\hat{x}^{\text{bp}}$ and $\hat{x}^{\text{lasso}}$. As a matter of fact, many of the results we have just mentioned are shown to hold for these two estimators as well[CRT04], albeit not for all values of $\alpha = \frac{M}{N}$ and $\rho = \frac{K}{N}$. One can thus define two regions in the α - ρ plane, depending on whether or not the estimators provide (approximately) the same result as $\hat{x}^{\text{sparse}}$; the line that separate these two regions is known as the Donoho-Tanner transition, and its shape has been first derived using combinatorial arguments[DT05; DT09].

Determining phase boundaries using statistical physics

We will now briefly mention how the formal results mentioned above, in particular the Donoho-Tanner transition, may be obtained using tools from statistical physics [KWT09]. First, notice that the ℓ_1 estimator introduced on an empirical basis may actually be obtained in a Bayesian setup; if we suppose the prior is a Laplace distribution with zero mean, $P_0(x_i) \propto \exp(-\beta|x_i|)$, then the

posterior reads

$$P(\mathbf{x}|F, \mathbf{y}) = \frac{1}{\mathcal{Z}(F, \mathbf{y})} e^{-\frac{1}{2\Delta} \|\mathbf{y} - F\mathbf{x}\|_2^2} \prod_{i=1}^N e^{-\beta|x_i|} \quad (4.26)$$

and computing the MAP estimator in this case is the same as solving the LASSO problem (4.19). If the signal has been originally sampled from a Gauss-Bernoulli distribution (4.14) and we compute estimators with respect to the measure above, then there are no guarantees these estimators are *Bayes-optimal*, i.e. that they lead to the global minimum of the loss function; still, they may provide good approximations.

Figure 4.5: Mean-squared error reported at each iteration of the AMP algorithm, using both a Laplace prior (4.26) in the $\beta \rightarrow \infty$ limit (red) and the correct Gauss-Bernoulli prior (4.14) from which the signal has been originally sampled (blue); the dotted line gives the MSE provided by the LASSO estimate (4.19). The results are for a single instance of size $N = 4096$ generated using latent-variable model (4.15), with $\rho = 0.2$ and $\Delta = 10^{-8}$, while the F matrix has been created using (4.25). Notice that for large α , both estimators provide results with low MSE (left), while for smaller values the Laplace prior leads to an estimate with high MSE (right).

Box 4.2: LASSO using AMP

The AMP algorithm introduced in section 2.2 may be used to compute the MAP estimator given posterior distribution (4.26). Note that in the $\beta \rightarrow \infty$ limit the MAP and MMSE estimators coincide, so we will consider

$$\begin{aligned} \lim_{\beta \rightarrow \infty} \frac{1}{\beta} \log z(\Sigma^2, R) &= \lim_{\beta \rightarrow \infty} \frac{1}{\beta} \log \int dx e^{-\frac{1}{2\Sigma^2} (x-R)^2 - \beta|x|} \\ &\approx \max_x \left\{ -\frac{1}{2\Sigma^2} (x-R)^2 - |x| \right\} \\ &= \begin{cases} \frac{\Sigma^2}{2} - |R|, & \text{if } |R| > \Sigma^2, \\ -\frac{R^2}{2\Sigma^2}, & \text{otherwise.} \end{cases} \end{aligned} \quad (4.27)$$

where we have rescaled $\Sigma^2 \rightarrow \Sigma^2/\beta$, and computed the integral using Laplace's method. From this quantity we obtain

$$\begin{aligned}
f_a(R, \Sigma^2) &= R + \Sigma^2 \frac{\partial}{\partial R} \log z(\Sigma^2, R) = \begin{cases} R - \Sigma^2, & \text{if } R > \Sigma^2, \\ R + \Sigma^2, & \text{if } R < -\Sigma^2, \\ 0, & \text{otherwise.} \end{cases} \\
f_c(R, \Sigma^2) &= \Sigma^2 \frac{\partial f_a}{\partial R}(\Sigma^2, R) = \begin{cases} \Sigma^2, & \text{if } |R| > \Sigma^2, \\ 0, & \text{otherwise.} \end{cases}
\end{aligned} \tag{4.28}$$

Replacing these in (2.67)-(2.72) effectively provide another way of solving the ℓ_1 minimization problem – in fact, analogous to the so-called *iterative soft thresholding* algorithm[DDD04]. Results yielded by this procedure over instances with different values of $\alpha = \frac{M}{N}$ can be seen in Figure 4.5.

Within a statistical physics point-of-view, the quantity we are interested in is the average free entropy, which using the replica trick may be written as

$$\bar{\phi} = \lim_{N \rightarrow \infty} \frac{1}{N} \mathbb{E}_{F, x_0, \xi} \log \mathcal{Z}(F, \mathbf{y}(x_0, \xi)) = \lim_{N \rightarrow \infty} \frac{1}{N} \lim_{n \rightarrow 0} \frac{\mathbb{E}_{F, x_0, \xi} \mathcal{Z}^n - 1}{n}, \tag{4.29}$$

with

$$\mathbb{E}_{F, x_0, \xi} \mathcal{Z}^n = \int \prod_{i, \alpha} dx_i^\alpha e^{-\beta |x_i^\alpha|} \prod_{\mu} \mathbb{E}_{F, x_0, \xi} \exp \left\{ -\frac{1}{2\Delta} \sum_{\alpha=1}^n \left[\sum_{i=1}^N F_{\mu i} (x_{0i} - x_i) + \xi_{\mu} \right]^2 \right\}. \tag{4.30}$$

We consider the noiseless case $\Delta = 0$, so that $\exp \left\{ -\frac{1}{2\Delta} \cdot \right\} \rightarrow \delta(\cdot)$; furthermore, in order to study the performance of the MAP estimator, we work in the $\beta \rightarrow \infty$ limit. While $P_0(x_i^\alpha) \propto \exp(-\beta |x_i^\alpha|)$ is the prior assumed by the model, $P(x_0) = \rho \mathcal{N}(x_0; \mu, \sigma^2) + (1 - \rho) \delta(x_0)$ is the distribution actually used to generate the signal we are looking for. Using (3.14) and after some work, one obtains that in the replica-symmetric approximation[KWT09]

$$\bar{\phi} = \min_{\mathbf{q}} f(\mathbf{q}), \tag{4.31}$$

where $\mathbf{q} = \{m, q, Q, \hat{m}, \hat{q}, \hat{Q}\}$ are order parameters defined throughout the calculation. The first three of them read

$$m = \frac{1}{N} \sum_i x_{0i} \langle x_i \rangle, \quad q = \frac{1}{N} \sum_i \langle x_i \rangle^2, \quad Q = \frac{1}{N} \sum_i \langle x_i^2 \rangle, \tag{4.32}$$

while the remaining ones are auxiliary parameters related to the Fourier transform of the three above. An explicit expression for the replica free energy $f(\mathbf{q})$ is given in [KWT09]. After writing the saddle-point equations $\frac{\partial f}{\partial q} = 0$ in the $\beta \rightarrow \infty$ limit, one may obtain the conditions for the planted vector to be recovered

$$\begin{cases} \alpha = 2(1 - \rho) \int_x^\infty \mathcal{D}z (z - x)^2 + \rho(1 + x^2), \\ \alpha > \rho + 2(1 - \rho) \int_x^\infty \mathcal{D}z, \end{cases} \tag{4.33}$$

where $\mathcal{D}z = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz$. These conditions are derived either by testing the stability of the RS solution[KWT09] or by considering a particular scaling of the order parameters[GS10].

Throughout the calculations, one finds out that the actual distribution of the non-zero coefficients is not important in determining the transition[KMS+12], that is, the same results are obtained for a generic distribution $P(x_0) = \rho \tilde{P}(x_0) + (1 - \rho) \delta(x_0)$ provided that $\tilde{P}(x_0) = \tilde{P}(-x_0)$. Furthermore, the same calculations may be done using ℓ_0 and ℓ_2 priors instead[KWT09], i.e., $P_0(x) \propto \exp(-\beta \|x\|_p)$; for $p = 2$, the condition for successful recovery is $\alpha > 1$, while for $p = 0$, the condition is $\alpha > \rho$, but for all these values the RS solution turns out to be unstable.

4.1.2 Bayes-optimal estimation

Ideally, if we had a way of efficiently computing MMSE estimates for the correct model (4.16), there would be no need of employing surrogates as we did in the last subsection. However, as we have already discussed, the correct model is intractable, i.e. it is computationally expensive to compute its marginals and thus any estimates that depend on them.

A possibility is to use the techniques for approximate inference that we introduced in chapter 2, for instance the belief propagation algorithm (1.9). As we saw however it is not adequate to employ standard BP in the treatment of models with continuous variables, so that further approximations need to be considered. The approximate message-passing (AMP) algorithm (2.67)-(2.72) is particularly well-suited for our purpose, since it is derived for distributions (2.45) that have the same form as the one we approach.

In order to use AMP with posterior distribution (4.16), one needs to calculate the $f_a(\Sigma^2, R)$ and $f_c(\Sigma^2, R)$ functions for a Gauss-Bernoulli prior; if we first compute

$$\begin{aligned} z(\Sigma^2, R) &= \int dx \left[\rho \frac{e^{-\frac{1}{2\sigma^2}(x-\mu)^2}}{\sqrt{2\pi\sigma^2}} + (1 - \rho) \delta(x) \right] e^{-\frac{1}{2\Sigma^2}(x-R)^2} \\ &= \underbrace{(1 - \rho) e^{-\frac{R^2}{2\Sigma^2}}}_{z_0} + \underbrace{\rho \sqrt{\frac{\Sigma^2}{\Sigma^2 + \sigma^2}} e^{-\frac{(\mu-R)^2}{2(\Sigma^2 + \sigma^2)}}}_{z_{\neq 0}}, \end{aligned} \quad (4.34)$$

then f_a and f_c are easily obtained from (2.55)

$$f_a(\Sigma^2, R) = R + \Sigma^2 \frac{1}{z} \frac{\partial z}{\partial R} = \frac{1}{1 + \frac{z_0}{z_{\neq 0}}} \cdot \frac{\sigma^2 R + \Sigma^2 \mu}{\Sigma^2 + \sigma^2}, \quad (4.35)$$

$$f_c(\Sigma^2, R) = \Sigma^2 \frac{\partial f_a}{\partial R} = \frac{z_0}{z_{\neq 0}} f_a^2 + \frac{1}{1 + \frac{z_0}{z_{\neq 0}}} \cdot \frac{\Sigma^2 \sigma^2}{\Sigma^2 + \sigma^2}. \quad (4.36)$$

After replacing these expressions in the AMP equations (2.67)-(2.72) and iterating them until convergence is reached, the (approximate) MMSE estimate is provided by $a_i = f_a(\Sigma_i, R_i)$. Figure 4.5 compares the performance of the MMSE estimate provided by AMP using the correct Gauss-Bernoulli prior with the one obtained using a Laplace prior that mimics the LASSO procedure (see Box 4.2).

We may study the typical performance of the MMSE estimate using the replica method, the same way we did in the previous subsection for the surrogate model but now for the correct one. To do this, we replace the prior $P_0(x_i^\alpha)$ in (4.30) by the one actually used to generate the signal $P(x_0)$ – in other words, we are now working in the Nishimori conditions. In particular, we are interested in computing the free entropy as a function of the mean of the squared distance between the estimate

$\mathbf{x}$ and the signal $\mathbf{x}_0$, $D = N^{-1} \sum_i \langle (x_i - x_{0i})^2 \rangle = Q - 2m + \rho \langle x_0^2 \rangle$; this can be done by extremizing the replica free energy $f(\mathbf{q})$ with respect to all order parameters $\{Q, q, \hat{Q}, \hat{q}, \hat{m}\}$ except m , which is fixed to $\frac{1}{2}(Q - D + \rho \langle x_0^2 \rangle)$

$$\bar{\phi}(D) = \text{extr}_{Q, q, \hat{Q}, \hat{q}, \hat{m}} f(Q, q, \frac{1}{2}(Q - D + \rho \langle x_0^2 \rangle), \hat{Q}, \hat{q}, \hat{m}). \quad (4.37)$$

An explicit expression to $f(\mathbf{q})$ is derived in [KMS+12]. Assuming the non-zero coefficients have zero mean and unit variance, i.e. $\mu = 0$ and $\sigma^2 = 1$, (4.37) evaluates to

$$\begin{aligned} \bar{\phi}(E) = & -\frac{\alpha}{2} \left[\log(\Delta + E) + \frac{\Delta}{\Delta + E} \right] + (1 - \rho) \frac{1}{2(\Sigma_0^2 + 1)} \\ & + (1 - \rho) \int \mathcal{D}z \log \left[(1 - \rho) e^{-\frac{z^2}{2(\Sigma_0^2 + 1)}} + \rho \sqrt{\frac{\Sigma_0^2}{\Sigma_0^2 + 1}} \right] + \rho \int \mathcal{D}z \log \left[(1 - \rho) e^{-\frac{z^2}{2\Sigma_0^2}} + \rho \sqrt{\frac{\Sigma_0^2}{\Sigma_0^2 + 1}} \right], \end{aligned} \quad (4.38)$$

for $E = N^{-1} \sum_i (\langle x_i \rangle - x_{0i})^2 = D/2$ and $\Sigma_0^2 = \frac{\Delta + E}{\alpha}$.

In order to obtain the correct value of the average free entropy, one would extremize $\bar{\phi}(E)$ once again. It is useful however to study the landscape of this function so as to provide insight on the performance of inference algorithms, since the algorithms themselves are looking for the maxima. If no local maxima exist, it is straightforward to design a dynamic that leads straight to the global one – that is in fact what the belief propagation algorithm does¹. When local maxima exist, however, the algorithm might get stuck in metastable states.

Figure 4.6: The replica free entropy $\bar{\phi}(E)$, (4.38), as a function of the average squared distance between the estimate and the planted signal, $E = \frac{1}{N} \mathbb{E}_{\mathbf{x}_0, \xi} \|\langle \mathbf{x} \rangle - \mathbf{x}_0\|_2^2$, for $\rho = 0.4$, $\Delta = 10^{-5}$ and different values of α . Notice that, as α decreases, the landscape of the free entropy undergoes significant changes, pointing to the occurrence of phase transitions. For large α , a single local maximum exists at $E \sim o(\Delta)$; in the first transition, deemed *spinodal*, a second local maximum appears at $E \sim o(1)$, and metastable states now exist, leading to the failure of the AMP algorithm. This local maximum eventually becomes global and finally unique at $\alpha \approx \rho$.

The $\bar{\phi}(E)$ function is plotted in Figure 4.6 for $\rho = 0.4$, $\Delta = 10^{-5}$ and different values of α . While for $\alpha > 0.58$ the AMP algorithm provides good estimates, i.e. estimates with MSE of $o(\Delta)$, for $\alpha < 0.58$ the MSE of the estimates is $o(1)$. Equivalently, for $\alpha > 0.58$, $\bar{\phi}(E)$ has a single maximum at $E \sim o(\Delta)$, while for $\alpha < 0.58$ a second maximum appears at $E \sim o(1)$. Thus, *the breakdown of the AMP algorithm can be explained by the existence of metastable states*.

¹One must be careful in that the BP algorithm does not act over the replica free energy, but over the more complicated Bethe free energy; they are however related to each other.

Eventually this 2nd maximum goes from local to global and finally, when $\alpha \approx \rho$, it becomes unique – these different transitions can be observed in Figure 4.6. Between the first transition (in which the 2nd maximum appears, known as *spinodal*) and the second one (in which it becomes global), AMP is sub-optimal: it is not able to lead to the global maximum of the free entropy. One is however able to escape the metastable states by employing a particular design of the F matrix[KMS+12].

Figure 4.7: *Spinodal transition* in the noiseless case ($\Delta = 0$): while in the green region, $\bar{\phi}(E)$ has a single local maximum at $E = 0$ and is thus monotonic, in the yellow region a second local maximum appears at $E \sim o(1)$ as metastable states appear, leading to the failure of the AMP algorithm. The Donoho-Tanner transition is given by the dashed line; notice that even algorithms that fail with the onset of metastability (such as AMP) attain better performance than the ℓ_1 minimization.

In Figure 4.7 we show the spinodal transition in the α - ρ plane for the noiseless case ($\Delta = 0$); the yellow region is determined by the appearance of the second maximum and thus coincides with the breakdown of AMP. Note that, despite AMP being sub-optimal in the yellow region, it improves on the performance of the ℓ_1 procedure, which fails to provide good results beneath the dashed line. This is in accordance with the results of Figure 4.5.

4.1.3 Conditions for the AMP algorithm to work

While analyzing AMP, we have assumed the elements of the F matrix to be sampled from

$$F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N}) \quad (4.39)$$

Through its derivation however, the only hypothesis that is made is that the elements of F must be of $o(\frac{1}{\sqrt{N}})$ (see discussion in section 2.2).

In compressed sensing, random matrices such as the above are widely used due to the fact they have the restricted isometry property, and for this matrix in particular, AMP proves to work well. However, in practical setups, one usually deals with more complicated, structured matrices[DE11]. In these cases, one verifies empirically that AMP often fails, even if the matrix has the RIP and the $F_{\mu i} \sim o(\frac{1}{\sqrt{N}})$ hypothesis is satisfied.

What happens in many situations is that the AMP fixed-point ceases to be stable throughout the iterations[CKZ14], thus making the estimate diverge. This lack of stability shows up for instance if the F matrix has positive mean, e.g. its elements are sampled from

$$F_{\mu i} \sim \mathcal{N}\left(\frac{\gamma}{N}, \frac{1}{N}\right) \quad (4.40)$$

with $\gamma \gtrsim 1$ (see Figure 4.8).

Figure 4.8: The AMP algorithm usually fails to converge whenever the elements of the matrix F are not sampled from $\mathcal{N}(0, \frac{1}{N})$. Here we show the mean-squared error obtained by iteration of the algorithm over single instances of size $N = 4096$ generated according to (4.15), with $F_{\mu i} \sim \mathcal{N}(\frac{\gamma}{N}, \frac{1}{N})$ and $\rho = 0.5$, $\Delta = 10^{-8}$ and different values of α . Notice that for both values of α the algorithm converges when $\gamma = 0$, but diverges already at the first few iterations when $\gamma = 2$.

Thus, even though the AMP algorithm works better than the ℓ_1 minimization under the assumptions we have made, it fails to converge in many other important cases. In the next two sections, we will try to address this problem by introducing alternative approaches. We will take two different aspects into account in order to measure their performance – whether they achieve the same mean-squared error as AMP in the standard case $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$, and whether they converge in cases where AMP does not.

4.2 Variational free energies

In order to derive the AMP algorithm in section 2.2, we started from the belief propagation equations and, since we were not able to integrate them, we employed approximations that allowed us to write closed-form expressions to be iterated. From our discussion in section 2.1, we know that BP is a fixed-point iteration that leads to the minima of a certain function – the Bethe free energy. The question we want to answer is **whether AMP does the same, and what is the function being minimized in this case.**

If we are able to find such function, we could use it to overcome AMP's convergence problems, since they are due to properties inherent to the fixed-point iteration[CKZ14] and could thus be resolved by employing other optimization algorithms.

4.2.1 Naïve mean-field approximation

We start by deriving the equations for the naïve mean-field approximation, which provides a simpler picture that might give us some insight. Recalling that the posterior distribution we are interested in is

$$P(\mathbf{x}|F, \mathbf{y}) = \frac{1}{\mathcal{Z}} e^{-\frac{1}{2\Delta} \|\mathbf{y} - F\mathbf{x}\|_2^2} \prod_{i=1}^N P_0(x_i), \quad (4.41)$$

and that the mean-field approximation gives

$$Q(\mathbf{x}) = \prod_{i=1}^N q_i(x_i), \quad q_i(x_i) \propto \exp \left\langle \log P(\mathbf{y}|F, \mathbf{x}) + \sum_{j=1}^N \log P_0(x_j) \right\rangle_{\mathbf{x}_{\setminus i}}, \quad (4.42)$$

we obtain

$$Q(\mathbf{x}; \mathbf{R}, \Sigma^2) = \prod_{i=1}^N q_i(x_i; R_i, \Sigma_i^2), \quad q_i(x_i; R_i, \Sigma_i^2) = \frac{1}{z_i} P_0(x_i) \exp \left\{ -\frac{1}{2\Sigma_i^2} (x_i - R_i)^2 \right\}, \quad (4.43)$$

Next we compute the associated (variational) free energy

$$\begin{aligned} \mathcal{F}[Q] &= \text{KL}[Q \| P(\mathbf{x}, \mathbf{y})] = \frac{1}{2\Delta} \left\langle \|\mathbf{y} - F\mathbf{x}\|_2^2 \right\rangle_Q + \frac{M}{2} \log \Delta + \text{KL}[Q \| P_0(\mathbf{x})] \\ &= \sum_{\mu=1}^M \frac{1}{2\Delta} \left[\left(y_\mu - \sum_{k=1}^N F_{\mu k} \langle x_k \rangle_Q \right)^2 + \sum_{k=1}^N F_{\mu k}^2 \text{Var}_Q(x_k) \right] + \frac{M}{2} \log \Delta + \sum_{i=1}^N \text{KL}[q_i \| P_0(x_i)], \end{aligned} \quad (4.44)$$

or, writing it as a function of the parameters of the trial distribution, $\mathbf{R}$ and Σ^2

$$\begin{aligned} \mathcal{F}_{\text{MF}}(\mathbf{R}, \Sigma^2) &= \sum_{\mu=1}^M \frac{1}{2\Delta} \left[\left(y_\mu - \sum_{k=1}^N F_{\mu k} a_k \right)^2 + \sum_{k=1}^N F_{\mu k}^2 c_k \right] + \frac{M}{2} \log \Delta \\ &\quad + \sum_{i=1}^N \left\{ -\log z_i(\Sigma_i^2, R_i) - \frac{(a_i - R_i)^2 + c_i}{2\Sigma_i^2} \right\}, \end{aligned} \quad (4.45)$$

where a_i and c_i are shorthands for $f_a(\Sigma_i^2, R)$ and $f_c(\Sigma_i^2, R)$ respectively, using the f_a and f_c functions defined in (2.55). Let us consider a more general *unconstrained* version of the free energy above, in which $\mathbf{a}$ and $\mathbf{c}$ are also variables; computing derivatives with respect to a_i and c_i and equating them to zero yield

$$\frac{\partial \mathcal{F}_{\text{MF}}}{\partial a_i} = 0 \quad \Rightarrow \quad \sum_{\mu=1}^M \frac{F_{\mu i}}{\Delta} \left(y_\mu - \sum_{k=1}^N F_{\mu k} a_k \right) - \frac{a_i - R_i}{\Sigma_i^2} = 0, \quad (4.46)$$

$$\frac{\partial \mathcal{F}_{\text{MF}}}{\partial c_i} = 0 \quad \Rightarrow \quad \sum_{\mu=1}^M \frac{F_{\mu i}^2}{2\Delta} - \frac{1}{2\Sigma_i^2} = 0, \quad (4.47)$$

Figure 4.9: The naïve mean-field iteration (4.48)-(4.50) coupled with the learning rule for Δ , (4.51), provides an algorithm that often leads to good results. Here we run the algorithm for single instances of size $N = 4096$ created using (4.15), with $F_{\mu i} \sim \mathcal{N}(\frac{\gamma}{N}, \frac{1}{N})$ and $\rho = 0.5$, $\Delta = 10^{-8}$ and different values of α . For $\alpha = 0.8$, the algorithm provides good results even for large values of γ , differently from AMP (see Figure 4.8). On the other hand, for $\alpha = 0.65$ it converges to a high MSE solution even for $\gamma = 0$, conditions in which AMP works well.

while derivatives with respect to R_i and Σ_i^2 simply enforce the consistency conditions $a_i = f_a(R_i, \Sigma_i^2)$ and $c_i = f_c(R_i, \Sigma_i^2)$. The stationarity conditions above may be used to devise a fixed-point iteration; in particular, by solving for R_i and Σ_i we obtain

$$R_i^{(t+1)} = a_i^{(t)} + \frac{\Sigma_i^2}{\Delta} \sum_{\mu=1}^M F_{\mu i} \left(y_{\mu} - \sum_{k=1}^N F_{\mu k} a_k^{(t)} \right), \quad (4.48)$$

$$\Sigma_i^2 = \frac{\Delta}{\sum_{\mu} F_{\mu i}^2}, \quad (4.49)$$

$$a_i^{(t+1)} = f_a(\Sigma_i^2, R_i^{(t)}). \quad (4.50)$$

Since fixed-point iterations are guaranteed to converge to a local minima of the naïve mean-field free energy **when iterated sequentially** (see discussion in Box 2.1), the scheme above always converges; however, it converges to a solution with high MSE. Remarkably though, if in addition to this procedure we employ the expectation-maximization algorithm (see Box 1.3) so as to learn the noise parameter[DHD12], by setting at each iteration

$$\frac{\partial \mathcal{F}}{\partial \Delta} = 0 \quad \Rightarrow \quad \Delta = \frac{1}{M} \sum_{\mu=1}^M \left[\left(y_{\mu} - \sum_{k=1}^N F_{\mu k} a_k \right)^2 + \sum_{k=1}^N F_{\mu k}^2 c_k \right], \quad (4.51)$$

then the algorithm will, in many cases, provide solutions with low MSE (see Figure 4.9); moreover, the naïve mean-field free energy actually decreases at each iteration of the algorithm, as it can be seen in Figure 4.10. Still, the region of the α - ρ plane for which it works proves to be smaller than the one for which AMP works.

The improved stability of this algorithm seems to come from the fact it is iterated sequentially. Differently from AMP, it shows to converge for many different types of matrices F , although not

always to low MSE solutions. In Figure 4.9 we investigate the behaviour of the algorithm for matrices with elements sampled from

$$F_{\mu i} \sim \mathcal{N}\left(\frac{\gamma}{N}, \frac{1}{N}\right). \quad (4.52)$$

Figure 4.10: The naïve mean-field free energy is guaranteed to decrease after each iteration of (4.48)-(4.50), provided that they are performed sequentially (i.e. not in parallel). These results are for the same instances used in Figure 4.9.

For $\rho = 0.5$, $\alpha = 0.8$ the algorithm provides good solutions even for large values of γ , while AMP fails already at $\gamma = 2$. On the other hand, for $\rho = 0.5$, $\alpha = 0.65$, it converges to a high MSE solution even for $\gamma = 0$, while AMP converges to a low MSE solution (see Figure 4.8).

When $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$, the central limit theorem gives $\sum_{\mu} F_{\mu i}^2 \approx \frac{M}{N} = \alpha$, so that $\Sigma_i^2 \approx \Delta/\alpha$ and the naïve mean-field iteration becomes

$$a_i^{(t+1)} = \eta_{\Delta} \left(a_i^{(t)} + \alpha^{-1} \sum_{\mu} F_{\mu i} \delta_{\mu}^{(t)} \right), \quad (4.53)$$

for $\delta_{\mu} = y_{\mu} - \sum_k F_{\mu k} a_k^{(t)}$ and $\eta_{\Delta}(\cdot) = f_a(\cdot, \Delta/\alpha)$, which is the same form assumed by iterative thresholding algorithms[MD10]. In Figure 4.11 we compare the empirical performance obtained by this algorithm (in association with noise learning) with that of AMP (2.67)-(2.72) for different values of ρ and α – as expected, AMP provides good results within a larger portion of the α - ρ plane.

4.2.2 Bethe approximation

After [RSR+13], we now consider the following trial distribution, which can be seen as providing independent approximations to the distributions of $\mathbf{x}$ and $\mathbf{z} = F\mathbf{x}$

$$Q(\mathbf{x}, \mathbf{z}) = Q_z(\mathbf{z}; \boldsymbol{\omega}, \mathbf{V}) \delta(\mathbf{z} - F\mathbf{x}) Q_x(\mathbf{x}; \mathbf{R}, \Sigma^2), \quad (4.54)$$

We set $Q_z = \prod_{\mu} q_{\mu}(z_{\mu}; \omega_{\mu}, V_{\mu})$, $Q_x = \prod_i q_i(x_i; R_i, \Sigma_i^2)$ and

$$q_{\mu}(z_{\mu}; \omega_{\mu}, V_{\mu}) = \frac{1}{Z_{\mu}} P(y_{\mu}|z_{\mu}) e^{-\frac{(z_{\mu} - \omega_{\mu})^2}{2V_{\mu}}}, \quad q_i(x_i; R_i, \Sigma_i^2) = \frac{1}{z_i} P_0(x_i) e^{-\frac{(x_i - R_i)^2}{2\Sigma_i^2}}, \quad (4.55)$$

The variational free energy then reads

$$\begin{aligned} \mathcal{F}[Q] &= \text{KL}[Q \| P(\mathbf{x}|F, \mathbf{y})] = \sum_{\mu=1}^M \text{KL}[q_{\mu} \| P(y_{\mu}|z_{\mu})] + \sum_{i=1}^N \text{KL}[q_i \| P_0(x_i)] + \sum_{\mu=1}^M H[q_{\mu}] \\ &= - \sum_{\mu} \left[\log Z_{\mu} + \left\langle \frac{(z_{\mu} - \omega_{\mu})^2}{2V_{\mu}} \right\rangle_{q_{\mu}} \right] - \sum_i \left[\log z_i + \left\langle \frac{(x_i - R_i)^2}{2\Sigma_i^2} \right\rangle_{q_i} \right] \\ &\quad + \frac{1}{2} \sum_{\mu} \left[1 + \log \left(2\pi \frac{\Delta V_{\mu}}{\Delta + V_{\mu}} \right) \right]. \end{aligned} \quad (4.56)$$

Figure 4.11: Empirical probability of success given many runs of the naïve mean-field iteration (4.53) in association with noise learning (left) and of the AMP algorithm (right). For each pair of values (ρ, α) , 100 instances of size $N = 1024$ were generated following (4.15), with $\Delta = 10^{-8}$ and $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$. The grayscale indicate how many instances each of the algorithms was able to solve, i.e. black points indicate no instance was solved while for white points low MSE solutions were obtained for all instances. The top line gives the Donoho-Tanner transition, while the line in the middle gives the Bayes-optimal spinodal transition; the dashed line in the bottom gives the theoretical limit.

By defining $a_i \equiv \langle x_i \rangle_{q_i} = f_a(x_i; R_i, \Sigma_i^2)$, $c_i \equiv \text{Var}_{q_i}(x_i) = f_c(x_i; R_i, \Sigma_i^2)$ and

$$g_\mu \equiv g(y_\mu, \omega_\mu, V_\mu) = \left\langle \frac{z_\mu - \omega_\mu}{V_\mu} \right\rangle_{q_\mu}, \quad (4.57)$$

the expressions between brackets may be written as

$$\left\langle \frac{(x_i - R_i)^2}{2\Sigma_i^2} \right\rangle_{q_i} = \frac{(a_i - R_i)^2 + c_i}{2\Sigma_i^2}, \quad \left\langle \frac{(z_\mu - \omega_\mu)^2}{2V_\mu} \right\rangle_{q_\mu} = \frac{V_\mu}{2} \left(g_\mu^2 + \frac{\partial g_\mu}{\partial \omega_\mu} \right) + \frac{1}{2}. \quad (4.58)$$

For a Gaussian likelihood $P(y|z) \propto e^{-\frac{y-z}{2\Delta}}$ the g function (4.57) becomes $g(y, \omega, V) = \frac{y-\omega}{\Delta+V}$. At this point, [RSR+13] employs two approximations: weak consistency constraints that give $\langle z_\mu \rangle = \sum_i F_{\mu i} a_i$, $\text{Var}(z_\mu) = \sum_i F_{\mu i}^2 c_i$, and the replacement of the entropy $H[q_\mu] = \frac{1}{2} + \frac{1}{2} \log(2\pi \frac{\Delta V_\mu}{\Delta+V_\mu})$ by a convex upper bound $H_{\text{Gauss}}[q_\mu] = \frac{1}{2} \frac{\Delta}{\Delta+V_\mu} + \frac{1}{2} \log(2\pi V_\mu)$, so that the variational free energy, as written in terms of the parameters of the trial distribution, becomes

$$\begin{aligned} \mathcal{F}_{\text{AMP}}(\mathbf{R}, \Sigma^2, \boldsymbol{\omega}, \mathbf{V}) = & \sum_\mu \frac{(y_\mu - \omega_\mu)^2}{2(\Delta + V_\mu)} - \sum_\mu \frac{(\omega_\mu - \sum_i F_{\mu i} a_i)^2}{2V_\mu} + \frac{1}{2} \sum_\mu \log(\Delta + \sum_i F_{\mu i}^2 c_i) \\ & - \sum_i \left\{ \log z_i(\Sigma_i^2, R_i) + \frac{c_i + (a_i - R_i)^2}{2\Sigma_i^2} \right\}. \end{aligned} \quad (4.59)$$

One can show that the extrema of this function give the fixed points of the AMP algorithm, by computing the following derivatives

$$\frac{\partial \mathcal{F}_{\text{AMP}}}{\partial \omega_\mu} = 0 \Rightarrow \frac{y_\mu - \omega_\mu}{\Delta + V_\mu} + \frac{\omega_\mu - \sum_i F_{\mu i} a_i}{V_\mu} = 0, \quad (4.60)$$

$$\frac{\partial \mathcal{F}_{\text{AMP}}}{\partial a_i} = 0 \Rightarrow -\sum_\mu F_{\mu i} \frac{w_\mu - \sum_i F_{\mu i} a_i}{V_\mu} - \frac{a_i - R_i}{\Sigma_i} = 0, \quad (4.61)$$

$$\frac{\partial \mathcal{F}_{\text{AMP}}}{\partial c_i} = 0 \Rightarrow -\frac{1}{2} \frac{F_{\mu i}^2}{\Delta + \sum_i F_{\mu i}^2 c_i} - \frac{1}{2\Sigma_i^2} = 0. \quad (4.62)$$

which, after a rearrangement of the terms, lead to the AMP iteration (2.67)-(2.72). Still, both the ansatz (4.54) and the approximations used by [RSR+13] are somewhat arbitrary, and in fact it can be shown there is no need to use them [KMTZ14] – one may start directly from the Bethe free energy as written in terms of the messages (2.30) and employ the expressions we have obtained for $m_{a \rightarrow i}$ and $\hat{m}_{i \rightarrow a}$ while deriving rBP, (2.49) and (2.52). In order to perform the integrals yielded by this procedure in closed-form, it is only necessary to employ the approximations used to go from rBP to AMP in section 2.2. By doing this, expression (4.59) is re-obtained without the use of heuristic arguments.

4.2.3 Employing general-purpose optimization algorithms

Once we have explicit expressions for the variational free energies, it is not necessary to use fixed-point iterations as we have been doing – there is as a matter of fact a plenitude of other algorithms that, differently from fixed-point iterations, are guaranteed to lead to local minima (see discussion in section 2.3). We must be careful however in that the message-passing algorithms we have seen so far lead in principle to any extremum of the free energy, and not necessarily to minima or maxima.

Let us first look at the naïve mean-field (NMF) free energy (4.45); notice that it is bounded below by $\frac{M}{2} \log(\Delta)$, since the two sums are both non-negative, and thus it must possess local minima. Another indication of this comes from the fact that the NMF iteration is guaranteed to *decrease* the free energy (see Box 2.1), so its fixed points are actually minima; furthermore, since the iteration coupled with noise learning leads to good results, some of these minima must be associated to low MSE estimates.

Now consider the constrained analogue of the Bethe free energy (4.59) with ω and V at their fixed points, where the following relations are satisfied

$$\frac{y_\mu - \omega_\mu^*}{\Delta + V_\mu^*} = -\frac{\omega_\mu^* - \sum_k F_{\mu k} a_k}{V_\mu^*} = \frac{y_\mu - \sum_i F_{\mu i} a_i}{\Delta}, \quad V_\mu^* = \sum_k F_{\mu k}^2 c_k, \quad (4.63)$$

so that (4.59) becomes

$$\begin{aligned} \mathcal{F}_{\text{AMP}}^{\text{constr}}(\mathbf{R}, \Sigma^2) \equiv \mathcal{F}_{\text{AMP}}(\mathbf{R}, \Sigma^2, \omega^*, \mathbf{V}^*) &= \frac{1}{2\Delta} \sum_\mu \left(y_\mu - \sum_k F_{\mu k} a_k \right)^2 + \frac{1}{2} \sum_\mu \log(\Delta + \sum_k F_{\mu k}^2 c_k) \\ &+ \sum_i \left\{ -\log z_i(\Sigma_i^2, R_i) - \frac{c_i + (a_i - R_i)^2}{2\Sigma_i^2} \right\}. \end{aligned} \quad (4.64)$$

As the mean-field free energy, this expression is bounded below by $\frac{M}{2} \log(\Delta)$; thus, at least some of its extrema must be minima. Additionally, since $\log(1+x) \leq x$, it is bounded above by F_{MF} , so it should perform at least as well as the naïve mean-field approach.

Figure 4.12: Mean-squared error of the estimate obtained by directly minimizing the naïve mean-field free energy (left) and the constrained Bethe free energy (right), (4.45) and (4.64) respectively. For each pair of values (α, ρ) , a single instance of size $N = 1024$ was generated following LVM (4.15), with $\Delta = 10^{-8}$ and $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$. Note the similarity between these and Figure 4.11, which reports the results obtained by the respective fixed-point iterations.

In order to minimize expressions (4.45) and (4.64) with respect to $\mathbf{R}$, Σ^2 and Δ , we have used the L-BFGS-B algorithm[ZBLN97]. Figure 4.12 shows that, for zero-mean matrices, the direct optimization succeeds in the same region of the α - ρ plane as the correspondent fixed-point iteration. A comparison with the naïve mean-field iteration in the case of positive-mean matrices (for which AMP does not work) is provided in Figure 4.13: for some values of α and ρ , both algorithms work, with the direct optimization being slower and reaching slightly larger values of the free energy and of the MSE. However, if we fix ρ and decrease α , the mean-field iteration ceases to provide good results while the direct optimization does not, proving it might be an useful alternative to the other approaches we have described so far.

4.3 Swept approximate message-passing

Our purpose in the last section was to find alternatives to the AMP algorithm that work even when the elements of the measurement matrix F are not i.i.d. Gaussian – as we have seen, the estimate provided by the AMP algorithm diverges for most choices of matrices (see subsection 4.1.3). We now have two alternatives that are guaranteed to converge for any choice of F – the naïve mean-field iteration[DHD12; KMTZ14] and the direct optimization of the Bethe free energy[KMTZ14]. However, both these approaches present drawbacks: while the naïve mean-field iteration does not achieve the same performance as the AMP algorithm in the i.i.d. Gaussian case, the direct optimization is usually very slow and, even though there are guarantees of convergence depending on the particular optimization algorithm employed[RFSK15], there are no guarantees that a low MSE estimate will be provided – differently from AMP, that allows for a probabilistic analysis in the $N \rightarrow \infty$ limit that shows it leads to smaller MSEs at each iteration[KMS+12].

Ideally, we would like to have a fast, iterative algorithm that has the same performance as AMP and also provides guarantees or, at least, empirically verifiable convergence for many choices of measurement matrices. Remarkably, the naïve mean-field equations provide such guarantees while being very similar to the AMP equations (despite not being capable of achieving AMP’s performance); in

Figure 4.13: Comparison between the results obtained by employing the naïve mean-field iteration (4.48)-(4.50), in red, and by directly optimizing the Bethe free energy (4.64) using the L-BFGS-B algorithm[ZBLN97], in blue. The quantities in display were averaged over 100 instances of size $N = 1024$, generated using LVM (4.15) with $\rho = 0.4$, $\Delta = 10^{-8}$ and $F_{\mu i} \sim \mathcal{N}(\frac{15}{N}, \frac{1}{N})$ – note that the AMP algorithm would not converge in this case, since the elements of F have non-zero mean. The naïve mean-field iteration yields good results for large values of $\alpha = \frac{M}{N}$, but as α decreases, the algorithm fails to provide low MSE estimates; the direct optimization, on the other hand, is rather slow – it took us 511s in average to solve for a single instance, against 13s by the naïve mean-field iteration – and reaches slightly larger values of the MSE and the free energy, but works for values of α for which the naïve mean-field iteration does not.

fact, both sets of equations, (4.48)-(4.50) and (2.67)-(2.72), may be written as

$$\begin{cases} R_i = a_i + \Sigma_i^2 \sum_{\mu} F_{\mu i} \frac{r_{\mu}}{s_{\mu}}, \\ \Sigma_i^2 = \left[\sum_{\mu} \frac{F_{\mu i}^2}{s_{\mu}} \right]^{-1}, \end{cases} \quad (4.65)$$

but for slightly different definitions of r_{μ} and s_{μ}

$$\begin{cases} r_{\mu} = (y_{\mu} - \sum_k F_{\mu k} a_k) + \frac{r_{\mu}}{s_{\mu}} \sum_k F_{\mu k}^2 c_k, \\ s_{\mu} = \Delta + \sum_k F_{\mu k}^2 c_k, \end{cases} \quad (4.66)$$

with the terms in blue being exclusive to AMP – note that r_{μ} and s_{μ} are written in terms of the ω_{μ} and V_{μ} variables from AMP, $r_{\mu} = y_{\mu} - \omega_{\mu}$ and $s_{\mu} = \Delta + V_{\mu}$.

The main difference between the two algorithms is that AMP is supposed to be iterated in parallel, while the naïve mean-field iteration should be performed sequentially in order to converge. One might wonder if iterating AMP sequentially would not also lead to good results; however, it is not clear how to devise such an update schedule, since AMP consists in updating variables indexed by both $i = 1, \dots, N$ and $\mu = 1, \dots, M$, while the naïve mean-field iteration depends only on variables indexed by i . The two most evident ways of performing this sort of update would be i) updating, at each iteration, all variables indexed in μ and then all variables indexed in i ; or ii) updating all

variables indexed in μ each time a variable indexed in i is updated (see Algorithm 4.1) – however, neither of these schedules will make the algorithm converge.

<pre> Initialize r, s, R, Σ^2, a, c; for $t = 1, \dots, t_{max}$ do for $\mu = 1, \dots, M$ do Update r_μ, s_μ; end for $i = 1, \dots, N$ do Update R_i, Σ_i^2; $a_i \leftarrow f_a(\Sigma_i^2, R_i), c_i \leftarrow f_c(\Sigma_i^2, R_i)$ end end </pre>	<pre> Initialize r, s, R, Σ^2, a, c; for $t = 1, \dots, t_{max}$ do for $i = 1, \dots, N$ do for $\mu = 1, \dots, M$ do Update r_μ, s_μ; end Update R_i, Σ_i^2; $a_i \leftarrow f_a(\Sigma_i^2, R_i), c_i \leftarrow f_c(\Sigma_i^2, R_i)$; end end </pre>
--	---

Algorithm 4.1: Different attempts to make AMP converge by performing sequential updates. Left: at each iteration, we first update all the variables indexed by μ , and then all variables indexed by i . Right: each time a i variable is updated, all the μ are updated as well. Neither of the attempts leads to a convergent algorithm.

As we will show, the correct schedule, i.e. the schedule that leads both to convergence and (in many cases) to low MSE estimates, is the following: one should recompute all the μ terms at the beginning of the iteration, and then as each of the i variables is updated, the terms in r_μ and s_μ that depend on a_i and c_i also should be, **but the r_μ/s_μ term must remain constant all throughout the iteration** (see Algorithm 4.2). Moreover, it is fundamental that the i variables are traversed not in a fixed order, but instead in a random order that is changed from iteration to iteration.

```

Initialize  $r, s, R, \Sigma^2, a, c$ ;
for  $t = 1, \dots, t_{max}$  do
  for  $\mu = 1, \dots, M$  do
    | Update  $r_\mu, s_\mu$ ;
    |  $g_\mu \leftarrow r_\mu/s_\mu$ ;
  end
  for  $i = \sigma_t(\{1, \dots, N\})$  do
    Update  $R_i, \Sigma_i^2$ ;
     $a_i \leftarrow f_a(\Sigma_i^2, R_i), c_i \leftarrow f_c(\Sigma_i^2, R_i)$ ;
    for  $\mu = 1, \dots, M$  do
      |  $D_\mu \leftarrow y_\mu - \sum_k F_{\mu k} a_k, V_\mu \leftarrow \sum_k F_{\mu k}^2 c_k$ ;
      |  $r_\mu \leftarrow D_\mu + g_\mu V_\mu, s_\mu \leftarrow \Delta + V_\mu$ ;
    end
  end
end

```

Algorithm 4.2: In order for the algorithm to converge, one needs to i) recompute all the μ variables every time a i variable is updated, while keeping the r_μ/s_μ term constant throughout the iteration; and ii) perform the updates in a different random order at each iteration – $\sigma_t(\{1, \dots, N\})$ indicates a different permutation of $\{1, \dots, N\}$ must be used for each t . This is the SwAMP algorithm.

The motivation for keeping the r_μ/s_μ term fixed comes from the expansion that gives rise to the Onsager reaction term (2.65), where the $B_{\mu \rightarrow i}$ variable – that eventually becomes the r_μ/s_μ term –

is considered fixed. The same reasoning applies to the standard (parallel) AMP iteration, where the $\frac{r_\mu}{s_\mu} = \frac{y_\mu - \omega_\mu}{\Delta + V_\mu}$ term used in the ω update (2.68) is that of the last iteration, and not the current one.

Figure 4.14: Mean-squared error obtained by the SwAMP algorithm over single instances of size $N = 4096$, created using LVM (4.15) with $\rho = 0.4$, $\Delta = 10^{-8}$ and $F_{\mu i} \sim \mathcal{N}(\frac{\gamma}{N}, \frac{1}{N})$. Figures 4.8, 4.9 and 4.14 demonstrate the performance of different algorithms over instances generated using the same sets of parameters; while AMP fails for $\gamma \gtrsim 1$, the naive mean-field iteration breaks down at $\alpha = 0.65$. The SwAMP algorithm succeeds in both cases.

We denominate Algorithm 4.2 the *swept approximate message-passing* algorithm, or SwAMP. We have tested it with many different types of matrices F and, in all our tests, the algorithm has converged – although not always to low MSE estimates. Furthermore, it has the same performance as the AMP algorithm in cases where AMP works.

The theoretical analysis of this algorithm proves to be much more difficult than that of AMP. While message-passing algorithms iterated in parallel can be analyzed with the use of density evolution techniques, this is not possible in this case due to the sequential update schedule. Thus, an analytical proof that SwAMP always converges remains an open challenge.

We now present a range of numerical results demonstrating the effectiveness of the SwAMP algorithm for problems on which both standard AMP and ℓ_1 minimization via convex optimization fail to provide desirable reconstruction performance. All experiments were conducted on a computer with an i7-3930K processor and run via Matlab. We have provided demonstrations of the SwAMP code on-line ². For calculating ℓ_1 recoveries, we utilize an implementation of the SPGL1[vBF07] algorithm.

Compressed sensing with troublesome matrices

As discussed earlier, using matrices of non-zero mean to sample x is one of the simplest cases for which AMP can fail to converge. However, by using the proposed SwAMP approach, accurate estimates of x can be obtained even when the mean of the matrices is non-negligible. While it may be possible to use mean subtraction, our proposed approach does not require such preprocessing. Additionally, as we will show later, not all problems are amenable to such mean subtraction. To evaluate the effectiveness of SwAMP as compared to the standard parallel-update AMP iteration,

²<https://github.com/eric-tramel/SwAMP-Demo>

we draw the matrix elements from

$$F_{\mu i} \sim \mathcal{N}\left(\frac{\gamma}{N}, \frac{1}{N}\right), \quad (4.67)$$

where the magnitude of the projector mean is controlled by the term γ . For a given signal $\mathbf{x}$ and noise variance Δ , as γ increases from 0, we expect to see AMP failing to converge. This behavior can be observed in the numerical experiments presented in Figure 4.15a. Here, we observe that SwAMP is robust to values of γ over an order of magnitude larger than the standard AMP, converging to a low-MSE solution even for $\gamma \approx 140$ while AMP fails already at $\gamma = 2$. Additionally, for the tested parameters, ℓ_1 minimization fails to provide a meaningful reconstruction for any value of γ .

Figure 4.15: Numerical experiments with troublesome matrices. **(a)** AMP, SwAMP, and ℓ_1 solvers compared for CS signal reconstruction for sensing matrices with positive mean on sparse signals of size $N = 10^4$ and sparsity $\rho = 0.2$ with noise variance $\Delta = 10^{-8}$. The matrices have been created following (4.67) using $M = \alpha N$ measurements with $\alpha = 0.5$. Finally, a comparison between the reconstruction error obtained by SwAMP and ℓ_1 -minimization is given at the bottom for the same experimental settings. **(b)** At the top, convergence behavior of AMP and SwAMP are compared for CS signal reconstruction for correlated sensing matrices on sparse signals of size $N = 10^4$ and sparsity $\rho = 0.2$ with noise variance $\Delta = 10^{-8}$. The projectors have been created according to (4.68) and are rank-deficient for $\eta < \alpha = 0.6$. At the bottom, a comparison between log-scale average reconstruction MSE obtained by SwAMP, BPDN, adaptive Lasso, and ℓ_p regularization is given for signals of size $N = 1024$ for $\Delta = 10^{-8}$, $\rho = 0.2$, and $\alpha = 0.6$.

We also consider an even more troublesome case, namely, matrices with strongly correlated elements. Such problems are of interest as they arise naturally in machine learning and biomedical applications, where the practitioner does not design the projections for optimal signal recovery, but rather the projections represent sampled observations from which the practitioner desires to predict some response variables by finding an interpretable, i.e. sparse, set of regression coefficients. Such observations can be highly correlated and thus represent a significant impediment to the use of parallel AMP for regression tasks.

For these tests, we draw

$$F = \frac{1}{N}PQ, \quad \text{where } P_{\mu k}, Q_{ki} \sim \mathcal{N}(0, 1) \quad (4.68)$$

with $P \in \mathbb{R}^{M \times R}$, $Q \in \mathbb{R}^{R \times N}$ and $R \equiv \eta N$. That is, F is *rank-deficient* for $\eta < \alpha$, where $\alpha = \frac{M}{N}$. In our experiments, we use η to denote the level of independence of the rows of F , with lower values of η representing a more difficult problem. We observe that the elements of F are neither normal nor i.i.d. for these experiments. In Figure 4.15b we see that SwAMP is robust to even these extremely troublesome matrices while AMP fails to converge.

As we cannot guarantee that SwAMP provides Bayes optimal estimation for problems exhibiting strong correlations in F , we provide a comparison to other well-known approaches in Figure 4.15b. Here, we compare the recovery performance of SwAMP to adaptive Lasso[Zou06], basis pursuit denoising (BPDN)[CDS98], as well as the ℓ_p generalization of Lasso[Tib96] for the non-convex case $p < 1$ [Cha07]. In order to address the free-parameter λ of the regularization strength in both ℓ_p and adaptive Lasso, we compute the entire solution path and report only the most accurate solution, providing best-case oracle results for these approaches. Additionally, for adaptive Lasso we choose a weight exponent of 0.1. The tests are conducted over 500 independent realizations of the sparse reconstruction problem for $N = 1024$, $\alpha = 0.6$, and $\rho = 0.2$ with a noise variance $\Delta = 10^{-8}$. We assume that the value of Δ is known *a priori* in all the tested approaches. For the implementation of the ℓ_p regression we utilized the SparseReg Matlab toolbox[ZAD12], while we use the SpaSM Matlab toolbox[SCLE12] for the implementation of adaptive Lasso.

We can see that SwAMP provides more accurate estimates across all η while also possessing a more robust transition between successful and unsuccessful reconstructions in terms of η . Over the tested η , SwAMP averaged 2 seconds of computation time, BPDN averaged 8 seconds, and ℓ_p and adaptive Lasso required on average 40 and 14.88 seconds of compute time respectively. We point out these run times to demonstrate that despite the potential complexity of the SwAMP fixed-point iteration, its computational burden is on par with, or an improvement on, other well known techniques.

These two experiments demonstrate how the proposed SwAMP iteration allows for AMP-like performance while remaining robust to conditions outside the assumptions about the projector.

Box 4.3: Group testing

Group testing, also known as *pooling* in molecular biology, is an approach to designing experiments so as to reduce the number of tests required to identify rare events or faulty items. In the most naïve approach to this problem, the number of tests is equal to the number of items, as each item is tested individually. However, since only a small fraction of the items may be faulty, the number of tests can be significantly reduced via pooling, i.e. testing many items simultaneously and allowing items to be included within multiple different tests. The nature of this linear combination of tests allows for a CS-type approach to faulty item detection, but with a few important caveats. First, the operator is extremely sparse since the number of pools, and the number of items in them, may be limited due to physical testing constraints. Second, the elements of this operator are commonly 0/1. Group testing is therefore a very challenging application for AMP since the properties of the group testing operator do not match AMP's assumptions.

In one recent work[ZKMZ13], the authors use both BP and AMP for group testing and found that while basic AMP would not converge, very good results—optimal ones, in fact—could be obtained by using a BP approach. This came at a large computational cost, however. Here, we have repeated the experiment of [ZKMZ13] using the SwAMP approach instead of AMP and BP. In fact, for SwAMP, a sparse operator is a very advantageous situation in terms of computational efficiency. Since the projector is extremely sparse by construction, we may explicitly ignore operations involving null elements, thus considerably improving the algorithm's speed, as seen in Figure 4.16b. Here, we also see that SwAMP's computational complexity is on the order of $O(N^2)$, as is AMP's. Group testing experiments are shown in Figure 4.16a where we use random 0/1 projections, under the constraint that each projection should sum to 7, to sample sparse 0/1 signals with $K \ll N$ non-zero elements, where N is the signal dimensionality. While AMP diverges when attempting to recover these signals, SwAMP converges to the correct solution in few iterations. Additionally, SwAMP very closely matches the BP transition, thus providing recovery performance better than convex optimization, just as BP does, but with much less computational complexity.

Figure 4.16: (a) Group testing phase transition diagram between successful and unsuccessful signal recovery over M , the number of pools, and K , the number of non-zero signal elements. Successful recovery means the correct identification of all signal elements. The top-left of the diagram represents the easiest problems while the bottom-right the most difficult. The transition lines are drawn along the contour of 50% of recoveries succeeding for many trials. (b) Execution times for both SwAMP and AMP using a sparse matrix with 25% of its elements having non-zero value. The reported times are measured for 500 iterations of the algorithms for each value of N for the parameters $\rho = 0.25$ and $\alpha = 0.75$.

Generalized swept approximate message-passing

The swept strategy we have introduced is also effective for the generalized AMP, or GAMP, algorithm (2.76)-(2.78). GAMP[Ran10] allows us to deal with models that have non-Gaussian likelihoods, i.e. any latent variable model where the observed variables $\mathbf{y}$ are a stochastic function of $\mathbf{z} = F\mathbf{x}$. As AMP, it suffers from convergence problems for most choices of F that depart too much from the ideal one where $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$.

The GAMP equations depend on evaluating the following quantity

$$g(y, \omega, V) = \int dz P(y|z) e^{-\frac{1}{2V}(z-\omega)^2}, \quad (4.69)$$

which for a Gaussian likelihood $P(y|z) \propto e^{-\frac{(y-z)^2}{2\Delta}}$ becomes $g(y, \omega, V) = \frac{y-\omega}{V+\Delta}$. Note that this is *precisely* the quantity that must remain fixed in the SwAMP iteration; it is thus clear that, in order to prescribe a swept version of GAMP, one must evaluate (4.69) and keep it fixed throughout each iteration. The resulting procedure is described in Algorithm 4.3.

```

Initialize  $\omega, V, \mathbf{R}, \Sigma^2, \mathbf{a}, \mathbf{c}$ ;
for  $t = 1, \dots, t_{\max}$  do
  for  $\mu = 1, \dots, M$  do
     $g_\mu \leftarrow g(y_\mu, \omega_\mu, V_\mu)$ ;  $h_\mu \leftarrow \frac{\partial g}{\partial \omega}(y_\mu, \omega_\mu, V_\mu)$ ;
  end
  for  $i = \sigma_t(\{1, \dots, N\})$  do
     $\Sigma_i^2 \leftarrow \left[ \sum_\mu F_{\mu i}^2 h_\mu \right]^{-1}$ ;
     $R_i \leftarrow a_i + \Sigma_i^2 \sum_\mu F_{\mu i} g_\mu$ ;
     $a_i \leftarrow f_a(\Sigma_i^2, R_i)$ ,  $c_i \leftarrow f_c(\Sigma_i^2, R_i)$ ;
    for  $\mu = 1, \dots, M$  do
       $\omega_\mu \leftarrow \sum_k F_{\mu k} a_k - g_\mu V_\mu$ ;  $V_\mu \leftarrow \sum_k F_{\mu k}^2 c_k$ ;
    end
  end
end

```

Algorithm 4.3: Generalized SwAMP algorithm, capable of dealing with models with non-Gaussian likelihoods, i.e. any latent variable model where the observed variables $\mathbf{y}$ are a stochastic function of $\mathbf{z} = F\mathbf{x}$. The $g(y, \omega, V)$ function is defined in (4.69). For $P(y|z) \propto e^{-\frac{1}{2V}(y-z)^2}$, $g = \frac{y-\omega}{\Delta+V}$ and we recover the original SwAMP algorithm.

Box 4.4: 1-bit compressed sensing

It is interesting to consider situations where we do not have access to the full measurements $\mathbf{z} = F\mathbf{x}$ but only to their signs, that is

$$\mathbf{y} = \text{sgn}(F\mathbf{x} + \boldsymbol{\xi}). \quad (4.70)$$

This scenario arises for instance when the measurement is quantized so as to use less storage, procedure known as 1-bit compressed sensing. Assuming the noise variable is normally distributed, $\xi_\mu \sim \mathcal{N}(0, \Delta)$, the likelihood is given by a probit distribution

$$P(y_\mu | z_\mu \equiv \mathbf{F}_\mu \cdot \mathbf{x}) = \frac{1}{2} \text{erfc} \left(-\frac{y_\mu z_\mu}{\sqrt{2\Delta}} \right). \quad (4.71)$$

The model above also works well for binary classification, for which GAMP may also be used. The g function (4.69) may be computed in closed-form in this case, yielding

$$g(y, \omega, V) = \frac{2ye^{-\phi^2}}{\sqrt{2\pi(\Delta + V)} \text{erfc}(\phi)}, \quad \phi = \frac{y\omega}{\sqrt{2(\Delta + V)}}. \quad (4.72)$$

We also ran experiments to demonstrate the performance of the generalized SwAMP (G-SwAMP) algorithm, using the 1-bit compressed sensing model (4.70) and comparing G-SwAMP results with those provided by GAMP and the binary iterative hard thresholding (BIHT) algorithm[LB12].

In Figure 4.17a, we see the results provided by G-SwAMP for $F_{\mu i} \sim \mathcal{N}\left(\frac{20}{N}, \frac{1}{N}\right)$. We observe that G-SwAMP performs admirably even for this non-negligible mean on the projectors. In terms of recovery performance, it does not quite meet the theoretical Bayes optimal performance[XKZ14], however, this is expected as the Bayes optimal performance is calculated for $\gamma = 0$. Additionally, we see that even for this non-zero mean, G-SwAMP outperforms both the BIHT's empirical performance for the same mean, as well as the best-case theoretical ℓ_1 performance for zero mean[XK13]. Finally, in Figure 4.17b, we see that GAMP fails to provide any meaningful signal recovery for γ small, while G-SwAMP continues to converge to low-MSE even for large values of γ .

Figure 4.17: Results for 1-bit CS. (a) Top: Comparison between the Bayes optimal MSE for zero-mean projectors[XKZ14] (dashed lines) and that obtained by SwAMP for projectors with $\gamma = 20$ (markers) for three different levels of signal sparsity. The reported empirical results were obtained by averaging over 200 instances of size $N = 512$. Bottom: Comparison of SwAMP and BIHT for $\rho = 1/8$ for experiment conditions identical to the figure above; theoretical results for zero-mean projectors are also presented for completeness, including theoretical ℓ_1 performance[XK13]. (b) Single instance comparison between GAMP and G-SwAMP for 1-bit CS with $N = 2048$, $\rho = 1/8$, and $\alpha = 3$.

4.3.1 The big picture

The field of compressed sensing has been in steady development ever since Candès, Romberg and Tao provided, in 2004, guarantees to obtain sparse solutions to linear systems. Algorithms based on the minimization of the ℓ_1 norm are still widely used: even though, under ideal conditions, Bayesian algorithms have proved to work better, these conditions are rarely met. We have tried in our work to tackle one of the biggest limitations of the AMP algorithm, namely that it does not converge for most choices of measurement matrices.

In section 4.2, we have analyzed the naïve mean-field approach to compressed sensing, proposed before by [DHD12]; furthermore, we have written the Bethe free energy for this problem. We have shown that the direct optimization of these quantities – using general-purpose algorithms for convex

optimization – should be regarded as a valid alternative approach which, in spite of being slow, leads to good results even when ideal conditions are not met.

After noticing that the naïve mean-field iteration always converges due to the fact it is performed sequentially, we have tried in section 4.3 to mimic this strategy for the AMP algorithm. By looking at the algorithm’s derivation, in particular at the expansion that leads to a correction term that is not present in the mean-field iteration (the Onsager reaction term), we were able to devise a schedule that seems to converge for any choice of F . Even though an analytic proof of this remains an open challenge, we have showed that this algorithm converges and provides good results in many cases for which the other existing approaches do not.

Finally, in Table 4.1, we present a brief summary of the different approaches to compressed sensing, comparing the main advantages and disadvantages of each of them. While certainly not exhaustive, it contains all the approaches we have studied throughout this thesis; the first two of them, in particular, are probably the two most popular among practitioners, while the last two are the ones we have introduced.

algorithm/ approach	type	speed	region where it works (for $F_{\mu i} \sim \mathcal{N}(0, \frac{1}{N})$)	guaranteed to converge for any F
ℓ_1 optim.	optimization	slow	small	yes
AMP	iterative	very fast	large	no
NMF iter.	iterative	fast	medium	yes
BFE optim.	optimization	slow	large	yes
SwAMP	iterative	fast	large	no proof

Table 4.1: Comparison between the approaches presented in the thesis. The first two are arguably the most popular approaches among CS practitioners, while the last two are the ones we have proposed. While we still do not have an approach which is good in all criteria, we were able to solve some of the problems with the previous approaches, in particular the lack of convergence presented by AMP.

Even though the proposed approaches solved the main problem we had before – namely, the lack of convergence presented by AMP – there are still many remaining challenges. It would be interesting for instance to have an iterative algorithm that provably converges in every situation. Furthermore, we would like to be able to relax the hypotheses needed on the F matrix, and to use any knowledge we have on the distribution of its elements[CWF14]. Finally, the use of non-factorized priors could also be of interest in many applications. Addressing these issues would gap the bridge between theory and practice, making Bayesian algorithms virtually unrivaled among algorithms for compressed sensing.

Information Hiding

As demonstrated by a number of incidents in the past decade, the average citizen has little to no privacy in today's world, making it increasingly important to develop mechanisms that allow one to safely transmit sensitive information to others. Cryptography, for instance, allow two individuals who share a secret key to send messages to each other without a third party being able to access the *content* of these messages. However, if the communication channel available to these individuals is censored, then encrypted messages will be easily identified and suppressed.

Thus arises the need of being able to communicate a message without being noticed. Algorithms that seek to accomplish this are known in the engineering literature as *steganographic algorithms* [Fri10; Cox08; CCCM10]. Steganography is a fairly recent, steadily growing field that is concerned with formalizing these concepts and developing increasingly better algorithms that allow one to *hide* information within other forms of media – notably digital media, such as images, videos and text files. A good steganographic algorithm should be able to hide messages while maintaining typical properties of the media intact, so that the censor is not able to distinguish the corrupted media from an ordinary one.

If on one hand these methods should be promoted so as to protect free speech, their use should be restrained if it serves illegal purposes. In this case, one would like to be able to identify whether a given media contains a hidden message; the field concerned with this task is known as *steganalysis*. There is thus a competition of sorts between steganography and steganalysis: while the former tries to design algorithms that are not detectable by steganalysts, the latter seeks to ensure steganographers are not successful in their endeavor.

5.1 Steganographic algorithms

We shall assume without loss of generality that the media where the message will be hidden is an image. We could in principle create an image that contains a given message, or pick an image from a set according to the information we want to convey; we will however consider a particular approach to steganography known as *steganography by cover modification* [Fri10], in which we are given an image and a message, and we want to modify the image so that the message is *embedded* in it, i.e. so that the message may be extracted from it by the recipient.

Let us introduce some notation. The original image $x_0 \in \mathcal{I}^N$ is known as the *cover* image, while the modified one, $x \in \mathcal{I}^N$, is deemed the *stego* image. There are many ways to represent an image; in what follows, we assume the image is rasterized, so that the elements of $\mathcal{I}$ give the possible values assumed by the pixels of the image – for instance, a grayscale image would have $\mathcal{I} = \{0, 1, \dots, 255\}$; furthermore, we assume the matrix describing the image was flattened, that is,

if the image has dimensions $W \times H$, then we concatenate the rows of the matrix that represents it in order to form a one-dimensional vector of size $N = WH$. Finally, the message we want to hide is given by a vector $\mathbf{m} \in \mathcal{M}^M$.

Box 5.1: Digital watermarking

An important application of steganography comes from the case where the message is related to the content, i.e. $\mathbf{m}$ is a function of $\mathbf{x}_0$. That might be useful for instance in identifying copyright infringement. This procedure goes by the name of *digital watermarking* and is widely employed in the multimedia industry.

Following this approach, a steganographic algorithm consists in two functions[Fri10]: one that generates the stego image $\mathbf{x}$ containing $\mathbf{m}$ (the *embedding* function), and another that extracts the message $\mathbf{m}$ from $\mathbf{x}$ (the *extraction* function). It is assumed a secret key κ has been shared between the sender and the recipient, and that both operations depend on this key.

$$\begin{cases} \mathbf{x} = \text{Emb}(\mathbf{x}_0, \mathbf{m}, \kappa), \\ \mathbf{m} = \text{Ext}(\mathbf{x}, \kappa). \end{cases} \quad (5.1)$$

One usually requires that these two functions are designed so as to satisfy $\mathbf{m} = \text{Ext}(\text{Emb}(\mathbf{x}_0, \mathbf{m}, \kappa), \kappa)$ for every triplet $(\mathbf{x}_0, \mathbf{m}, \kappa)$.

A good algorithm should be capable of embedding large messages while making minimal changes to $\mathbf{x}_0$, so that the changes will go undetected by the censor. In order to measure these changes, we introduce a distortion function $D : \mathcal{I} \times \mathcal{I} \rightarrow \mathbb{R}$; for instance, in the case of grayscale images the mean-squared distance can be used, $D(\mathbf{x}_0, \mathbf{x}) = \sum_i (x_{0i} - x_i)^2$. In other words then, we would like to keep D as small as possible while allowing for large values of $\alpha \equiv \frac{M}{N}$.

5.1.1 A naïve algorithm

Suppose the images we are working with are grayscale, $\mathcal{I} = \{0, 1, \dots, 255\}$, and that the messages are binary, $\mathcal{M} = \{0, 1\}$. Instead of working with the full image, we may choose to work only with its least-significant bits (LSBs) – that is, from all the eight bits of the binary representation of each pixel, we are only allowed to change the last one. For instance, a pixel assuming a value of 147 (10010011 in binary) would either stay unchanged or become 146 (10010010 in binary).

A naïve algorithm[Fri10] consists in picking a M -permutation of $\{1, \dots, N\}$, $\sigma = (i_1, \dots, i_M)$, to then replace the LSB of each pixel x_{0i_j} by the corresponding bit of the message, m_j . With a slight abuse of notation, we will denote by $\mathbf{x}$ not the full image but the representation of the image we are working with (the LSBs in this case); the embedding function then reads

$$x_i = \begin{cases} m_j, & \text{if } i = \sigma_j^{(\kappa)}, \\ x_{0i}, & \text{otherwise,} \end{cases} \quad (5.2)$$

where $\sigma^{(\kappa)}$ is a M -permutation of $\{1, \dots, N\}$ indexed by the secret key $\kappa \in \mathbb{Z}$ – for instance, κ can be used as the seed of the random number generator that generates the permutation. The extraction function, on the other hand, is simply

$$m_j = x_{\sigma_j^{(\kappa)}}, \quad j = 1, \dots, M. \quad (5.3)$$

Figure 5.1: Scheme for the naive algorithm defined by (5.2) and (5.3). The cover image $x_0 \in \mathcal{I}^N$ is 8-bit grayscale, i.e. $\mathcal{I} = \{0, 1, \dots, 255\}$, while the message $\mathbf{m} \in \mathcal{M}^M$ is binary, $\mathcal{M} = \{0, 1\}$. After flattening x_0 into a 1D array, we turn to its least-significant bit (LSB) representation, given by the rightmost digit in the binary expansion of each pixel intensity. We then generate a M -permutation of $\{1, \dots, N\}$ and focus on the positions of the array indicated by the permutation (blue boxes); each bit in this sequence is replaced by the correspondent bit of the message. Finally, the modifications are carried back to original image. Note that, since only bits in the LSB plane have been flipped, the modifications cannot be perceived by just looking at the image; however, someone that has access to the pixel intensities and knows the permutation that has been used may extract the message.

Note that this algorithm is rather inefficient, as in the worst case scenario one must alter M bits of x_0 so as to convey M bits of information. We could also analyze its performance in the typical case; to do that, we compute the average of the Hamming distance between x and x_0 , given by

$$\begin{aligned}
 D_H^{\text{avg}} &= N^{-1} \mathbb{E}_{x_0, \mathbf{m}} D(x_0, x(x_0, \mathbf{m})) \\
 &= N^{-1} \mathbb{E}_{x_0, \mathbf{m}} \sum_i \mathbb{I}(x_{0i} \neq x_i) \\
 &= \alpha \mathbb{E}_{x_0, \mathbf{m}} \mathbb{I}(x_{0i} \neq m_i).
 \end{aligned} \tag{5.4}$$

In order to proceed, we need to assign probability distributions to the messages and to the images; as a starting point, we assume that both x_{0i} and m_i are i.i.d. Bernoulli random variables with $p = 1/2$, yielding

$$D_H^{\text{avg}} = \frac{\alpha}{2}, \tag{5.5}$$

that is, for every bit of x_0 that is altered, $\alpha/D_H^{\text{avg}} = 2$ bits of information are conveyed in average.

Efficiency and detectability

When analyzing steganographic algorithms, we will be mainly interested in two of their properties: how *efficient* they are; and how *detectable* is their output, i.e. how easy it is to detect there is a

Figure 5.2: Left: cover image, 8-bit grayscale Lena of dimensions 128×128 , so that $N = 16384$. Middle: LSB plane of the cover image. Right: LSB plane of the stego image, after embedding a message of 10% the size of the image, $M = 1639$, with each bit of the message randomly sampled from a Bernoulli distribution with $p = 1/2$.

message embedded in x . Efficiency can be measured by ratio between the number of bits conveyed and the number of bits of x_0 that have been modified

$$\varepsilon = \alpha / D_H^{\text{avg}}. \quad (5.6)$$

As we have seen, $\varepsilon = 2$ for the naïve algorithm we have discussed. Detectability is more difficult to define, as it depends on the approach employed by the steganalyst. Potential approaches are broadly divided into two groups[Fri10] – targeted and blind – depending on whether or not the method used to embed the message is known.

For this naïve algorithm in particular[WP00], there is a very simple targeted attack that reveals a message was embedded. If one looks at the histogram of pixel intensities (Figure 5.3), it is clear that the stego image evens out neighbouring pairs; that is because, since only the LSB may be modified, a pixel intensity of 147 may only be changed to 146, and vice-versa. A simple way of circumventing this attack is: instead of modifying only the LSB, we pick either $+1$ or -1 uniformly at random, and sum that quantity to the pixel intensity – that way, if we want to change the LSB of 147 from 1 to 0, it would either become 146 or 148. While the mean-squared distance between the cover and stego images would be the same as the obtained by the previous algorithm, the effect on the histogram would not be present anymore. We thus see the previous LSB algorithm’s output is easily detectable, while the ± 1 approach just described is more robust.

In general however we will try to measure undetectability with respect to blind approaches. Blind steganalysis may be regarded as a problem of anomaly detection: after devising a statistical model for natural images, the steganalyst is able to identify images that have been adulterated by asserting whether or not they agree with the model – of course, such a model may be very hard to devise. Analogously, the task may be considered one of binary classification: one seeks a classifier $C_\theta : \mathcal{I}^N \rightarrow \{0, 1\}$ that evaluates to 1 whenever the image $x \in \mathcal{I}^N$ in hands contains a hidden message and to 0 otherwise – θ in this case is the vector of parameters used in the classification, and it may be learned by looking at labeled examples of adulterated and non-adulterated images (i.e. by training the classifier).

Figure 5.3: Top to bottom: histogram of pixel intensities in the (a) cover image, 5-bit grayscale Lena of dimensions 512×512 ; (b) stego image generated using the LSB algorithm; and (c) stego image generated using the ± 1 algorithm. The message embedded has the same size as the image, i.e. $\alpha = 1$, and its bits were sampled from a Bernoulli distribution with $p = 1/2$. The LSB algorithm evens out pairs of pixel intensities, yielding a histogram that is unusual for natural images and thus making the embedded message detectable.

Measuring the distortion

We will assume that if the stego image x is *close enough* to the cover image x_0 in some properly defined metric, then the stego image is undetectable[PFB10] – since x_0 is a natural image, it should agree with the model devised by the steganalyst, so that by generating a new image that has the same properties as x_0 we would also conform to this model. It is important though that the properties we keep are the same as those taken into account by the steganalyst’s model. For instance, the model can rely on the correlations between neighbouring pixels of an image; then if the correlations in x are very similar to those in x_0 , we would expect x to agree with the model.

Let us define a factor graph $\mathcal{G}(V_N \cup V_F, E)$, where the variable nodes $i \in V_N$ represent the bits of a binary image and the factor nodes $a \in V_F$ are linked to sets of variables which correlations are taken into account by the steganalyst (e.g. pair correlations). If every set of variables indexed by $\partial a = \{i_1^a, i_2^a, \dots\}$ has the same correlation in both x and x_0 , then the images are indistinguishable to the steganalyst. We thus introduce the following distortion function

$$D(x, x_0) = \sum_a \mathbb{I} \left(\bigoplus_{i \in \partial a} x_i \neq \bigoplus_{i \in \partial a} x_{0i} \right), \quad (5.7)$$

where $\oplus$ indicates the sum modulo 2. We will say that x is δ -undetectable whenever $D(x, x_0) < \delta$, with δ being a constant related to the capacity of the steganalyst. Of course, D is usually unknown to the steganographer. These notions shall be made precise in the next subsection.

Throughout the rest of this chapter, we shall only work with binary representations; we may assume, for instance, that we are working with the LSB plane of a grayscale image, and that in order to carry the modifications back to the original image, the ± 1 scheme we have mentioned is used. We are especially concerned in designing efficient algorithms, that have ε as large as possible; in order to find out what is the optimal value of ε , we need to look into theory on data compression that goes back as far as 1959[Sha59].

5.1.2 State-of-the-art: minimizing embedding distortion

Let us denote by $\mathcal{C}(\mathbf{m}, \kappa)$ the set of images $\mathbf{x} \in \{0, 1\}^N$ from which the message $\mathbf{m} \in \{0, 1\}^M$ may be extracted using a secret key $\kappa \in \mathcal{Z}$, that is

$$\mathcal{C}(\mathbf{m}, \kappa) = \{\mathbf{x} \mid \text{Ext}(\mathbf{x}, \kappa) = \mathbf{m}\}, \quad (5.8)$$

is the set of images that *contain* $\mathbf{m}$. If the steganographer is aware of the properties taken into account by the steganalyst's model, then he can design a distortion function D that measures how much these properties have changed from the cover to the stego image. He can then try to generate the stego image by specifically looking for the member of $\mathcal{C}(\mathbf{m}, \kappa)$ that minimizes the amount of changes necessary, effectively solving the following optimization problem

$$\begin{cases} \min_{\mathbf{x}} & D(\mathbf{x}, \mathbf{x}_0) \\ \text{s.t.} & \mathbf{x} \in \mathcal{C}(\mathbf{m}, \kappa). \end{cases} \quad (5.9)$$

This approach is known in the literature as steganography by *minimizing embedding distortion*, and it yields the best practical algorithms currently available, both in terms of efficiency and detectability [FF07b; Fri10]. In this context, the design of the distortion function D is fundamental [PFB10]. Ideally, the steganographer would like to keep unchanged all the properties the steganalyst might be looking for. Of course, this is impossible in practice, as there are many properties that might be used; moreover, trying to keep many properties unaltered will typically lead to larger distortion values and make the problem (5.9) harder to solve. Thus, the design should take into account only a few key properties.

Figure 5.4: Scheme for problem (5.9): given a vector $\mathbf{x}_0 \in \{0, 1\}^N$, a set of vectors $\mathcal{C} \subseteq \{0, 1\}^N$ and a metric d on $\{0, 1\}^N$, we want to determine what is the member of $\mathcal{C}$ that is closer to $\mathbf{x}_0$ in that metric.

In order to solve (5.9), we could in principle compute $D(\mathbf{x}, \mathbf{x}_0)$ for all $\mathbf{x} \in \mathcal{C}(\mathbf{m}, \kappa)$; however, the number of elements in $\mathcal{C}$ is typically very large, making this procedure unfeasible. As an example, consider the case where the set of 2^N images has been partitioned into 2^M disjoint subsets with 2^{M-N} elements each, and a different message $\mathbf{m} \in 2^M$ is associated to each subset, so that all the images in a given subset contain the same message $\mathbf{m}$. The number of elements in $\mathcal{C}$ is then exponential in N , $|\mathcal{C}| = 2^{N-M} = 2^{(1-\alpha)N}$.

In what follows, we would like to devise a *practical* way of solving (5.9) – i.e. our algorithm should run in polynomial time in N – if possible for any given choice of D and $\mathcal{C}$. After devising

Figure 5.5: Cartoon of the space of images: each image contains a certain message, as represented by the color of the image in the sketch – in this particular example, there are seven different colors representing seven different messages that can be transmitted. The steganographer has in hand an image x_0 (cross-hatched hexagon), which contains the green message; however, he wants to transmit the red message, so he needs to look for the closest image in red.

such a scheme, we would like to know what are the guarantees on its performance (if any), and what is the typical performance achieved in terms of efficiency and detectability. Finally, we would like to determine the optimal performance that can be achieved, and to compare it to our scheme's performance.

Lossy compression

We start by looking at the simplest form the distortion function can assume, namely that of the Hamming distance

$$D(\mathbf{x}, \mathbf{x}_0) = D_H(\mathbf{x}, \mathbf{x}_0) = N^{-1} \sum_i \mathbb{I}(x_i \neq x_{0i}). \quad (5.10)$$

By employing such a function we maximize the algorithm's efficiency $\varepsilon = \alpha/D_{\text{avg}}^H$. In order to determine the minimum achievable Hamming distance (analogously, the maximum achievable efficiency), we notice that problem (5.9) is essentially a problem of *lossy compression*, since we map a set containing $|\mathcal{I}|^N$ elements to one containing $|\mathcal{C}| \ll |\mathcal{I}|^N$.

In his 1948 work that inaugurated information theory, Shannon established limits not only for channel coding – the transmission of data over noisy channels – but also for *source coding* – the *compression* of data. According to the source coding theorem, a random variable X may be compressed without loss of information if and only if the compression rate R , i.e. the ratio between the number of bits after and before compression, is larger than the entropy of the random variable $H(X)$; the compression is then said to be *lossless*. We are interested in what happens when $R < H(X)$, that is, when there *is* loss of information and the compression is *lossy*. Shannon himself approached this problem almost ten years after his original work [Sha59]; he showed that, in order to obtain an average distortion no larger than D between the vector $\mathbf{x}$ to be compressed and the vector $\mathbf{y}$ obtained after decompression, the compression rate must be at least

$$\begin{aligned}
R_{\min}(D) = \min_{Q(\mathbf{y}|\mathbf{x})} & \sum_{\mathbf{x}, \mathbf{y}} Q(\mathbf{y}|\mathbf{x}) P(\mathbf{x}) \log \frac{Q(\mathbf{y}|\mathbf{x})}{\sum_{\mathbf{x}'} Q(\mathbf{y}|\mathbf{x}') P(\mathbf{x}')} \\
\text{s.t.} & \sum_{\mathbf{x}, \mathbf{y}} Q(\mathbf{y}|\mathbf{x}) P(\mathbf{x}) d(\mathbf{x}, \mathbf{y}) < D
\end{aligned} \tag{5.11}$$

This is the so-called rate-distortion function. The calculation above may be performed analytically for some choices of $P(\mathbf{x})$ and $d(\mathbf{x}, \mathbf{y})$; in particular, if $P(\mathbf{x})$ is a Bernoulli distribution of parameter p and $d(\mathbf{x}, \mathbf{y})$ is the Hamming distance between $\mathbf{x}$ and $\mathbf{y}$, we obtain

$$R_{\min}(D) = \begin{cases} H_2(p) - H_2(D), & \text{if } 0 \leq D \leq \min\{p, 1-p\}, \\ 0, & \text{otherwise,} \end{cases} \tag{5.12}$$

where $H_2(p) = -p \log p - (1-p) \log(1-p)$ is the binary entropy function. Note that this result is independent of *how* the compression is done – in other words, of how the set $\mathcal{C}$ is chosen.

Using this result, we may say in our case that, if the elements of $\mathbf{x}_0$ are Bernoulli random variables of parameter p and the distortion function chosen is the Hamming distance, then the minimum achievable distortion is $R_{\min}^{-1}(N^{-1} \log_2 |\mathcal{C}|)$, with $R_{\min}$ given by (5.12). In particular, if we assume the set of 2^N images has been partitioned into 2^M disjoint subsets of same size, then $\log_2 |\mathcal{C}| = N - M$ and $D_{\text{opt}}^H = R_{\min}^{-1}(1 - \alpha) = H_2^{-1}(\alpha)$ – or, in terms of the efficiency (5.6), $\epsilon_{\max} = \alpha / H_2^{-1}(\alpha)$. In Figure 5.6 we plot the rate-distortion function (5.12) in terms of these different variables.

Figure 5.6: Rate-distortion curve (5.12) for a binary symmetric source ($p = 1/2$), in terms of different variables. From left to right: rate of compression $R = (N - M)/N$ vs. distortion D ; distortion D vs. relative message size $\alpha = M/N$; efficiency ϵ vs. relative message size α . Note that $\alpha = 1 - R$ and $\epsilon = \alpha / H^{-1}(\alpha)$. The hatched region is unfeasible, i.e. no algorithm is able to achieve such values of rate and distortion.

Structure of linear codes

In order to devise practical algorithms, we need to specify how the set $\mathcal{C}$ is built given a message $\mathbf{m}$ and a key κ . We could in principle partition the set of all possible images $\{0, 1\}^N$ into 2^M subsets so as to map each message to a subset; this construction however is not practical, as it would require us to store in memory a hash table of size exponential in N .

In coding theory [Mac03; RU08], the set $\mathcal{C}$ is known as the *codebook*, while its elements are known as *codewords*. Many codebook constructions have been studied in the literature; one that will serve us particularly well consists in adapting linear codes that are traditionally used for channel coding. In brief, a binary linear code of length N and rank K is defined by a *parity-check* matrix $\mathbb{H} \in \{0, 1\}^{(N-K) \times N}$ which kernel gives the codebook, i.e. $\mathcal{C} = \{\mathbf{x} \mid \mathbb{H}\mathbf{x} = 0\}$. Alternatively, it may be described by a *generator* matrix $\mathbb{G} \in \{0, 1\}^{N \times K}$ which defines the codebook by means of $\mathcal{C} = \{\mathbf{x} \mid \mathbf{x} = \mathbb{G}\mathbf{z} \wedge \mathbf{z} \in \{0, 1\}^K\}$. Since every codeword $\mathbf{x} \in \mathcal{C}$ must satisfy both $\mathbb{H}\mathbf{x} = 0$ and $\mathbf{x} = \mathbb{G}\mathbf{z}$ for a given $\mathbf{z}$, it follows that $\mathbb{H}\mathbb{G} = 0$. All the operations are performed modulo 2.

Box 5.2: Channel vs. source coding

In channel coding, the sender picks a codeword $\mathbf{x} \in \mathcal{C}$, which is then corrupted by noise and arrives as $\mathbf{y}$ to the receiver – this process is usually described by a probability distribution $P(\mathbf{y}|\mathbf{x})$. The receiver may then apply the parity-check matrix to $\mathbf{y}$, yielding $\mathbb{H}\mathbf{y} = \mathbf{s}$, and the *syndrome vector* $\mathbf{s}$ is zero if and only if $\mathbf{y}$ is also a codeword – typically however this will not be the case.

The task of recovering $\mathbf{x}$ after receiving $\mathbf{y}$ is known as decoding. Decoding may be done in different ways, and is essentially a problem of statistical inference; in particular, optimal decoding is achieved by determining

$$\hat{\mathbf{x}}_{\text{MAP}} = \arg \max_{\mathbf{x}} P(\mathbf{x}|\mathbf{y}) \propto P(\mathbf{y}|\mathbf{x}) P_0(\mathbf{x}), \quad (5.13)$$

which is known to be computationally hard in most cases. For the binary symmetric channel, $P(\mathbf{y}|\mathbf{x}) = \prod_i [p \mathbb{I}(x_i \neq y_i) + (1-p) \mathbb{I}(x_i = y_i)] \propto \exp \left\{ \log \frac{1-p}{p} \sum_i \mathbb{I}(x_i = y_i) \right\}$, so if $\mathbf{x}$ is picked uniformly at random among the codewords, maximizing $P(\mathbf{x}|\mathbf{y})$ is analogous to solving

$$\begin{cases} \min & D_H(\mathbf{x}, \mathbf{y}) = \sum_i \mathbb{I}(x_i \neq y_i), \\ \text{s.t.} & \mathbb{H}\mathbf{x} = 0, \end{cases} \quad (5.14)$$

a problem very similar to (5.9). There is however a fundamental difference between them: in this case, we know there is a specific codeword $\mathbf{x}^*$ which is related to $\mathbf{y}$ according to $y_i = x_i^* \oplus \eta_i$, for $\eta_i \sim \text{B}(p)$ – in other words, $\mathbf{x}^*$ is a *planted configuration*. From this, it follows for instance that $N^{-1}D(\mathbf{x}^*, \mathbf{y})$ approaches p as $N \rightarrow \infty$. We may thus affirm channel coding is the planted analogue of source coding, which makes it a significantly easier problem (see discussion on section 3.2).

We thus proceed by employing linear codes to define the codebooks; the compression problem becomes

$$\begin{cases} \min_{\mathbf{x}} & D_H(\mathbf{x}, \mathbf{x}_0), \\ \text{s.t.} & \mathbf{x} \in \mathcal{C} = \{\mathbf{x} \mid \mathbb{H}\mathbf{x} = 0\}. \end{cases} \quad (5.15)$$

Using that $\mathbb{H}\mathbf{x} = 0 \Leftrightarrow \mathbf{x} = \mathbb{G}\mathbf{z}$, we may rewrite the problem above as an unconstrained optimization problem

$$\min_{\mathbf{z}} D_H(\mathbb{G}\mathbf{z}, \mathbf{x}_0), \quad (5.16)$$

and for $\mathbb{H}\mathbb{G} = 0$, both the constrained and the unconstrained problems have the same solution.

Figure 5.7: While noisy channel coding (left) is a problem of inference, lossy source coding (right) is a problem of optimization. The main difference lies on the distribution of the x_0 variables (white circles): while in channel coding they are related to the noise introduced by the channel, in source coding they are arbitrary.

We can easily generalize this procedure so as to have a codebook for each message $\mathbf{m} \in 2^M$; instead of solving for $\mathbb{H}\mathbf{x} = 0$, we solve for $\mathbb{H}\mathbf{x} = \mathbf{m}$. By doing this, the set of images $\{0, 1\}^N$ is partitioned into 2^M disjoint subsets $\{\mathcal{C}(\mathbf{m})\}$, each of them containing the 2^{N-M} solutions of the system for a fixed value of $\mathbf{m}$.

$$\begin{cases} \min_{\mathbf{x}} & D_H(\mathbf{x}, \mathbf{x}_0), \\ \text{s.t.} & \mathbf{x} \in \mathcal{C}(\mathbf{m}) = \{\mathbf{x} \mid \mathbb{H}\mathbf{x} = \mathbf{m}\}. \end{cases} \quad (5.17)$$

This approach goes by the name of *syndrome coding*[Fri10; Cra98], since instead of looking for a codeword we are now looking for a vector with syndrome $\mathbf{m}$.

If we pick any $\mathbf{x}_m^* \in \mathcal{C}(\mathbf{m})$, then all other members of $\mathcal{C}(\mathbf{m})$ can be determined from $\{\mathbf{x}_m^* \oplus \mathbf{x} \mid \mathbf{x} \in \mathcal{C}(0)\}$ – in other words, once we find one solution to the *nonhomogeneous* system, all the other solutions are given by combining it to the solutions of the *homogeneous* system; $\mathcal{C}(\mathbf{m})$ is said to be a coset of $\mathcal{C}(0)$ with respect to $\mathbf{x}_m^*$. Since $\mathcal{C}(0) = \{\mathbb{G}\mathbf{z} \mid \mathbf{z} \in \{0, 1\}^{(N-M)}\}$, problem (5.17) may also be written as

$$\min_{\mathbf{z}} D_H(\mathbf{x}_m^* \oplus \mathbb{G}\mathbf{z}, \mathbf{x}_0), \quad (5.18)$$

and again, if $\mathbb{H}\mathbb{G} = 0$, both problems (5.17) and (5.18) have the same solution.

Practical compression algorithms using belief propagation

As discussed in section 1.3, optimization problems such as (5.17) and (5.18) give rise to probability distributions, in this case given by

$$P_{\text{PC}}(\mathbf{x} \mid \mathbf{x}_0, \mathbf{m}, \mathbb{H}) = \frac{1}{\mathcal{Z}(\mathbf{x}_0, \mathbf{m}, \mathbb{H})} e^{-\beta D_H(\mathbf{x}, \mathbf{x}_0)} \mathbb{I}[\mathbb{H}\mathbf{x} = \mathbf{m}], \quad (5.19)$$

and

$$P_{\text{GM}}(\mathbf{z} \mid \mathbf{x}_0, \mathbf{x}_m^*, \mathbb{G}) = \frac{1}{\mathcal{Z}(\mathbf{x}_0, \mathbf{x}_m^*, \mathbb{G})} e^{-\beta D_H(\mathbf{x}_m^* \oplus \mathbb{G}\mathbf{z}, \mathbf{x}_0)}, \quad (5.20)$$

respectively, and we are interested in the $\beta \rightarrow \infty$ limit in both cases. We now map the $\{0, 1\}$ variables into $\{+1, -1\}$, so that $x \oplus y \rightarrow x \cdot y$ and $D_H(\mathbf{x}, \mathbf{y}) = (1 - N^{-1} \sum_i x_i y_i)/2$. By appropriately rescaling the terms, we may rewrite (5.19) as

$$P_{\text{PC}}(\mathbf{x}|\mathbf{x}_0, \mathbf{m}) \propto \prod_{i=1}^N \exp \{ \beta x_{0i} x_i \} \prod_{a=1}^M \mathbb{I} \left[\prod_{i \in \partial a} x_i = m_a \right]. \quad (5.21)$$

Analogously, (5.20) becomes

$$P_{\text{GM}}(\mathbf{z}|\mathbf{x}_0, \mathbf{x}_m^*) \propto \prod_{i=1}^N \exp \{ \beta x_{0i} x_{mi}^* \prod_{\mu \in \partial i} z_\mu \}, \quad (5.22)$$

Note that by setting $m_a = 1 \forall a$ in (5.21) or $x_{mi}^* = 1 \forall i$ in (5.22), we recover the compression problem. The factor graphs associated to these distributions are given in Figure 5.8.

Figure 5.8: The factor graph in the left is associated to model (5.21), and the one in the right to model (5.22). We refer to the scheme on the left as the one using *parity-check* matrices, while the one on the right uses *generator* matrices.

Let us now write the belief propagation equations for model (5.22), which will allow us to obtain approximate solutions to optimization problem (5.18). The equations for model (5.21) are similar and will be written later. Following (1.9) and (2.25), we obtain

$$\begin{cases} h_{\mu \rightarrow i} = \sum_{j \in \partial \mu \setminus i} u_{j \rightarrow \mu} \\ u_{i \rightarrow \mu} = \frac{1}{\beta} \tanh^{-1} \left[\tanh(\beta x_{0i} x_{mi}^*) \prod_{\nu \in \partial i \setminus \mu} \tanh(\beta h_{\nu \rightarrow i}) \right]. \end{cases} \quad (5.23)$$

In particular, in the $\beta \rightarrow \infty$ limit the equations for the u variables become

$$u_{i \rightarrow \mu} = \text{sgn} \left(x_{0i} x_{mi}^* \prod_{\nu \in \partial i \setminus \mu} h_{\nu \rightarrow i} \right) \min \left(\{x_{0i} x_{mi}^*\} \cup \{|h_{\nu \rightarrow i}|\}_{\nu \in \partial i \setminus \mu} \right). \quad (5.24)$$

In order to use these equations to solve optimization problem (5.18), the decimation procedure described in Box 1.6 must be employed. In short, the BP equations should be iterated until convergence is reached, at which point we pick a variable μ uniformly at random and fix it to either plus or minus one, depending on the current approximation for that variable's marginal distribution

$$P(z_\mu) \propto \exp \left\{ \beta \left(\sum_{i \in \partial \mu} u_{i \rightarrow \mu} \right) z_\mu \right\}, \quad (5.25)$$

i.e. if $P(z_\mu = +1) > P(z_\mu = -1)$, we set z_μ to $+1$, and otherwise we set it to -1

$$z_\mu^* = \text{sgn} \left(\sum_{i \in \partial \mu} u_{i \rightarrow \mu} \right).$$

We then iterate the BP equations once again, but now with the μ variable fixed – essentially, all the messages $h_{\mu \rightarrow \cdot}$ are set to either $+\infty$ or $-\infty$, depending on whether z_μ was fixed to $+1$ or -1 . This procedure is repeated until all the variables are fixed.

The performance of the decimation scheme will ultimately depend on how we choose the $\mathbb{H}$ and $\mathbb{G}$ matrices, i.e. on what is the topology of the factor graph¹. As discussed earlier, the BP equations only provide good approximations when the underlying factor graph is sparse; furthermore, correlations between variables in the graph should decay as their distance in the graph increases. In order to fulfill the first requirement, we will use matrices $\mathbb{H}$ and $\mathbb{G}$ that are *sparse*; ensembles of sparse $\mathbb{H}$ matrices are typically known as *low-density parity check* (LDPC), and of sparse $\mathbb{G}$ matrices as *low-density generator matrices* (LDGM).

It has been showed[MY04] that ensembles of LDPC matrices do not typically achieve rate-distortion, i.e. the average distortion obtained when using these ensembles does not approach $R_{\min}^{-1}$ as $N \rightarrow \infty$; that is because, in order for this to happen, the factor nodes' degrees should be large, and the $\mathbb{H}$ matrix relatively dense. Thus, compression using LDPC has been neglected by all but a few studies[BKZ09]. On the other hand, employing LDGM ensembles to solve the dual problem (5.18) seems to yield good results, as we will see next.

Box 5.3: Ensembles of LDPC and LDGM

Ensembles of LDPC and LDGM are typically defined by the degree distribution of the factor and variable nodes in the corresponding factor graph. Some examples are

- k -regular/Poisson ensemble – every factor node is linked to k variables nodes picked uniformly at random, so that the degree of the variable nodes follows a Poisson distribution.
- regular (c, p) ensemble – all variable nodes have degree c , and all factor nodes have degree p . The rate of the code is $R = p/c$.
- irregular ensemble – variable and factor nodes have arbitrary degree distributions.

We now analyze the rate-distortion performance of the BP-guided decimation over LDGM ensembles, considering at first regular (c, p) ensembles of varying rates $R = p/c$. At this point we are specifically interested in the performance for *compression*, so we set $x_{mi}^* = 1 \forall i$. In principle, the BP equations in the $\beta \rightarrow \infty$ limit should be used, since we are looking for the minima of the distortion function – that is, we should iterate (5.23) while replacing the u equations by (5.24). However, the BP algorithm often does not converge for large β , and using smaller values of β ends up leading to better results in practice (see Figure 5.9). The reason for this will be made clear in section 5.3.

In Figure 5.9, we have used regular $(c, 3)$ ensembles, that is, each factor node in the graph (see Figure 5.8a) is connected to 3 variable nodes, and each variable node is connected to $c = p/R$ factor nodes – for instance, in order to achieve $R = 1/2$, we pick $c = 6$. We extend the analysis for other regular (c, p) ensembles in Figure 5.10a, now using only $\beta = 1$; note that the average distortion increases with p .

As we can see, the naïve use of LDGM ensembles does not necessarily lead to good results; however, one is able to approach the rate-distortion curve by employing some particular variations

¹ $\mathbb{H}$ and $\mathbb{G}$ actually serve as adjacency matrices to the corresponding factor graphs.

Figure 5.9: Performance obtained in the solution of optimization problem (5.18) by using the BP equations (5.23) in conjunction with a decimation scheme (see Box 1.6). Results reported are averages over 100 instances of size $N = 10000$ – each instance consists of a graph sampled from the regular $(c, 3)$ ensemble and a vector x_0 to be compressed, $x_{0i} \sim B(1/2)$. Left: average distortion obtained using different values of β in the BP equations; even though $\beta = \infty$ should be used for optimal results to be obtained, in practice the BP equations do not converge and $\beta = 1$ ends up providing lower distortion values. The black line gives the optimal rate-distortion that can be obtained for a binary symmetric source, (5.12). Right: average distortion as a function of β , for different rates R ; note that the optimal distortion happens at a specific value of β which is rate-dependent.

that have been introduced in the past few years. There are basically three strategies that can be used

- decimating by using an improved algorithm inspired on survey propagation [WM05; MW06; WMM10], instead of BP;
- employing irregular LDGM ensembles that are dual to LDPC ensembles optimized for channel coding [FF07a];
- using spatially-coupled LDGM ensembles [AMUV12; AMV15].

The most straightforward of them is arguably the second one, since the other two depend on relatively complicated algorithms or code constructions. In Figure 5.10b we compare the performance of the irregular LDGM ensemble used by [FF07a] with that of regular ensembles; as we can see in Figure 5.10c, the distortion approaches the optimal as $N \rightarrow \infty$, and we can say that rate-distortion is achieved by this ensemble.

Back to steganography

The differences between compression and steganography are subtle but crucial. At a first glance, the main difference is in the distribution of $\mathbf{m}$: by setting $P(\mathbf{m}) = \prod_{a=1}^M \delta(m_a - 1)$, our approach to steganography turns into a classical problem of compression. There is however another peculiarity to steganography: the distortion functions that we are interested in are often much more complicated than the Hamming distance, since we want them to correlate well with the steganalyst's model[PFB10].

Figure 5.10: Rate-distortion performance of the BP-guided decimation, for different choices of factor graph and vector size. In (a), we use regular (c, p) ensembles with varying p ; in (b), we compare these ensembles with irregular ensembles optimized for channel coding [FF07a]; and in (c), we analyze the rate-distortion performance as a function of the length N of the vector to be compressed, while using the same irregular ensemble as in (b). Unless otherwise mentioned $N = 10000$, and in all cases $\beta = 1$. Notice in (c) that the empirical distortion approaches the optimal (dashed line) as N grows – for this reason, it is said that this irregular ensemble *achieves rate-distortion*.

In general however, it is difficult to approach more complicated distortion functions – the optimization problem becomes harder to solve, and there may be no guarantees whatsoever on the solutions obtained. A possibility is to use functions that are slightly more complicated but still tractable, such as the following *weighted Hamming distance* that has been considered in the steganography literature

$$D(\mathbf{x}, \mathbf{x}_0) = N^{-1} \sum_i w_i \mathbb{I}[x_i \neq x_{0i}] \quad (5.26)$$

where the weights $\{w_i\}$ are assigned according to how important it is for each pixel to remain unchanged. For instance, one may consider a more complicated distortion function $\tilde{D}$, and set the weights as $w_i = \tilde{D}(\mathbf{x}_0^{(i)}, \mathbf{x}_0)$, where $\mathbf{x}_0^{(i)}$ is the same as the original image $\mathbf{x}_0$ in all positions except i , i.e.

$$x_{0j}^{(i)} = \begin{cases} x_j, & \text{if } j = i, \\ x_{0j}, & \text{otherwise.} \end{cases}$$

This is the so-called *additive approximation*.

In [FF10], Filler and Fridrich introduced a framework to deal with more complicated distortion functions by sampling from

$$P(\mathbf{x}) = \frac{1}{\mathcal{Z}(\mathbf{x}_0)} e^{-\beta D(\mathbf{x}, \mathbf{x}_0)}, \quad (5.27)$$

using Markov chain Monte Carlo algorithms – more specifically the Gibbs sampler, reason why they name it the *Gibbs construction*. Besides being able to estimate the optimal rate-distortion given an arbitrary distortion function, they propose a practical scheme to sample an image $\mathbf{x}$ from (5.27) while embedding a message in it.

In the next section, we propose a way of dealing with complicated distortion functions that, in spite of having similarities to the Gibbs construction, goes in a different direction.

5.2 Exploiting LDPC-based quantizers

Yet another distinction between compression and steganography is that, in compression, one must necessarily look for a codeword – i.e. a vector in $\mathcal{C}$ – while in steganography we are not in principle restricted to the vectors in $\mathcal{C}(\mathbf{m})$: we may allow for the possibility of embedding a message $\mathbf{m}'$ that differs from $\mathbf{m}$ in just a few bits, so that

$$f = 1 - D_H(\mathbf{m}, \mathbf{m}'), \quad (5.28)$$

remains close to 1. We denote the quantity f above the *fidelity* in the transmission of the message. Note that it is not possible to achieve $f < 1$ using the generator matrix approach (5.18); on the other hand, in the dual approach using parity-check matrices (5.17), the hard constraint $\mathbb{I}[\prod_{i \in \partial a} x_i = m_a]$ can be relaxed into a soft one $\exp\{\tilde{\beta} m_a \prod_{i \in \partial a} x_i\}$

$$P_{\text{PC}}(\mathbf{x}|\mathbf{x}_0, \mathbf{m}) \approx \frac{1}{\mathcal{Z}(\mathbf{x}_0, \mathbf{m})} \prod_{i=1}^N \exp\{\tilde{\beta} B x_{0i} x_i\} \prod_{a=1}^M \exp\{\tilde{\beta} m_a \prod_{i \in \partial a} x_i\}. \quad (5.29)$$

In the $\tilde{\beta} \rightarrow \infty$ limit, the two terms compete with each other – the first seeks to minimize the Hamming distortion, and the second to maximize the fidelity (5.28). The B parameter should be set according to the relative importance of minimizing one and maximizing the other; for instance, using B small indicates that maximizing the fidelity is more important to us than minimizing the distortion. The BP equations for the model above are

$$\begin{cases} h_{i \rightarrow a} = B x_{0i} + \sum_{b \in \partial i \setminus a} u_{b \rightarrow i} \\ u_{a \rightarrow i} = \frac{1}{\tilde{\beta}} \tanh^{-1} \left[\tanh(\tilde{\beta} m_a) \prod_{j \in \partial a \setminus i} \tanh(\tilde{\beta} h_{j \rightarrow a}) \right], \end{cases} \quad (5.30)$$

and as before, in the $\tilde{\beta} \rightarrow \infty$ limit the equations for the u variables should be replaced by

$$u_{a \rightarrow i} = \text{sgn} \left(m_a \prod_{j \in \partial a \setminus i} h_{j \rightarrow a} \right) \min \left(\{m_a\} \cup \{|h_{j \rightarrow a}|\}_{j \in \partial a \setminus i} \right). \quad (5.31)$$

These equations should be iterated until convergence is reached, at which point the marginals of x_i are obtained from

$$P(x_i) \propto \exp \left\{ \tilde{\beta} \left(B x_{0i} + \sum_{a \in \partial i} u_{a \rightarrow i} \right) x_i \right\}. \quad (5.32)$$

An approximate solution to the optimization problem may be obtained by means of the same decimation procedure we used in the last section, i.e. by fixing the x_i variables one at a time so as to maximize $P(x_i)$.

Our goal in this section is to show that, in spite of LDPC ensembles not being able to achieve good rate-distortion performance in the solution of (5.17), they might still be useful, in particular for steganography. We will try to make a case for the relaxed approach above, by arguing that

- it does not necessarily results in significant losses in the rate-distortion performance (despite not approaching the rate-distortion as $N \rightarrow \infty$);
- it allows one to reduce the embedding distortion significantly by transmitting messages with $f < 1$;
- it is a more natural approach to complicated distortion functions.

Each of these is discussed separately below.

Rate-distortion performance

We first investigate the rate-distortion performance of this approach for different choices of parameters – we assume $m_a = 1 \forall a$, for which the compression problem is recovered. Since we are working with the relaxed model (5.29), we now have to measure not only the distortion D but also the fidelity f ; if $f < 1$, then the algorithm's output is useless for compression, since the approximate solution obtained is not a codeword.

In order to evaluate the performance, we generate many instances of the problem, run the BP-guided decimation algorithm and measure the distortion and fidelity of its output; we then compute the average of the results. Each instance consists of an image $x_0 \in \{+1, -1\}^N$ to be compressed and a matrix $\mathbb{H}$ which defines the factor graph; unless otherwise mentioned, $x_{0i} \sim B_{\pm 1}(1/2)$ and $\mathbb{H}$ is sampled from a p -regular/Poisson LDPC ensemble with $p = 8$.

The performance obtained for different choices of parameters is given in Figure 5.11. For smaller compression rates R , the algorithm typically is not able to achieve $f = 1$, and is thus unsuitable for compression. For large R however, a visual comparison with Figure 5.9 reveals the performance of this approach is comparable to the previous one using LDGM ensembles.

Figure 5.11: Performance of the BP-guided decimation in determining the maximum of the relaxed distribution (5.29), while using sparse factor graphs defined by p -regular/Poisson LDPC ensembles and different sets of parameters. The values reported for the distortion D and the fidelity f are averages over 500 instances of size $N = 1024$, obtained using different values of (a) β in the BP equations; (b) the B parameter tuning the relative importance of the distortion and the fidelity; and (c) the factor node degree p .

The B parameter seems to fulfill its role, as increasing it results in smaller values of both the distortion and the fidelity (Figure 5.11b). The relation between the performance achieved and the

value of β used, on the other hand, is somehow unclear, but using the $\beta \rightarrow \infty$ limit seems again to be unhelpful – for small R , it leads to a much larger distortion and insignificant gains in the fidelity (Figure 5.11a). Finally, reducing the factor node degree p leads to a larger fidelity but also a larger distortion (Figure 5.11c).

We conclude the analysis by looking at the dependency of the results obtained on the length N of the image (Figure 5.12a) and on the distribution of the message (Figure 5.12b). Note that the average distortion remains the same as N grows, so that it cannot be said that this scheme achieves rate-distortion.

Figure 5.12: Performance of the BP-guided decimation using (a) images x_0 of different lengths N ; and (b) different distributions for m_a . In (a), each circle gives the result obtained for a single instance, and the line gives the average over all instances; in (b), only the average is given. Unless otherwise mentioned, the 8-regular/Poisson ensemble is used, and instances have size $N = 1024$; in all cases, averages are computed over 500 instances, and β is set to 1.

Embedding messages with $f < 1$

We now move on to analyze the use of relaxed model (5.29) in the context of steganography, by giving a distribution to the messages instead of adopting $m_a = 1 \forall a$; more specifically, we set $P(m_a) = B_{\pm 1}(p_m)$, with $p_m = 1/2$ – notice however that the performance seems to be independent of the value of p_m (Figure 5.12b). We fix most of the parameters and investigate the performance with respect to $\alpha = M/N = 1 - R$ (the relative message size) and B only; the factor graph is sampled from the 8-regular/Poisson LDPC ensemble, the images from $P(x_{0i}) = B_{\pm 1}(1/2)$ and β is set to 1.

In Figure 5.13, we plot the values of D and f for many pairs (α, B) ; as the contour lines show, there is a large region of the plane where it is possible to obtain less than 10% of distortion while losing as little as 1% of the message.

In order to compare the performance obtained with the optimal one given by the rate-distortion curve, we may numerically invert D with respect to B , and plot the fidelity as a function of α and D (Figure 5.14). We thus see it is possible to achieve values of α and D below the rate-distortion curve while losing less than 5% of the message.

Figure 5.13: Average distortion (left) and fidelity (right) obtained using different relative message sizes $\alpha = M/N$ and values of the B parameter. Results reported are averages over 400 instances of size $N = 1024$, each consisting of an image $\mathbf{x}_0 \in \{+1, -1\}^N$, $x_{0i} \sim B_{\pm 1}(1/2)$, a message $\mathbf{m} \in \{+1, -1\}^M$, $m_a \sim B_{\pm 1}(1/2)$ and a factor graph sampled from the 8-regular/Poisson LDPC ensemble. The decimation was performed using the BP equations (5.30) with $\beta = 1$.

Figure 5.14: Average fidelity as a function of the relative message size α and the distortion D . For a fixed value of α , the trade-off between fidelity and distortion is governed by the B parameter (see Figure 5.12). As it can be seen, it is possible to achieve (α, D) values below the rate-distortion curve (thick line) while losing as little as 5% of the message bits.

Complicated distortion functions

As discussed earlier, another difference between steganography and compression is in the distortion functions used, which must reflect the properties taken into account by the steganalyst's model. For instance, the model might consider correlations between a pixel and its neighbours, i.e. the steganalyst measures these correlations in many natural images and builds a statistical model of how they typically behave. We may then write the following distortion function

$$D_C(\mathbf{x}, \mathbf{x}_0) = \sum_{(i,j) \in \mathbb{Z}^2} x_{0i} x_{0j} x_i x_j, \quad (5.33)$$

and by flipping bits of $\mathbf{x}_0$ while keeping D small, we generate an image $\mathbf{x}$ that is in accordance with the steganalyst's model – note that we are working with $x_i \in \{+1, -1\}$, so that by minimizing (5.33) we are effectively attempting to keep the product between pixels $(i, j) \in \mathbb{Z}^2$ the same in the two images.

In the generator matrix approach (5.22), x_i would be replaced by $x_{mi}^* \prod_{\mu \in \partial i} z_\mu$, yielding

$$P_{GM}(\mathbf{z} | \mathbf{x}_0, \mathbf{x}_m^*) \propto \prod_{(i,j) \in \mathbb{Z}^2} \exp \{ \beta x_{0i} x_{0j} x_{mi}^* x_{mj}^* \prod_{\mu \in \partial i} z_\mu \prod_{\nu \in \partial j} z_\nu \}, \quad (5.34)$$

which is rather difficult to treat, as the quantity of spins in each interaction has doubled in average. Luckily, in the parity-check matrix approach (5.21) changes are not as drastic – the first term remains unchanged, while the second becomes an Ising-like two-spin interaction

$$P_{PC}(\mathbf{x} | \mathbf{x}_0, \mathbf{m}) \propto \prod_{a=1}^M \exp \{ \beta m_a \prod_{i \in \partial a} x_i \} \prod_{(i,j) \in \mathbb{Z}^2} \exp \{ \beta B x_{0i} x_{0j} x_i x_j \}. \quad (5.35)$$

In what follows, we will consider a slightly more complicated version of this distortion function, a linear combination between D_C and the Hamming distance D_H , $D = D_H + \lambda D_C$. The distribution then assumes the following form

$$P_{PC}(\mathbf{x} | \mathbf{x}_0, \mathbf{m}) \propto \prod_{a=1}^M \exp \{ \beta m_a \prod_{i \in \partial a} x_i \} \prod_{(i,j) \in \mathbb{Z}^2} \exp \{ \beta \lambda x_{0i} x_{0j} x_i x_j \} \prod_{i=1}^N \exp \{ \beta B x_{0i} x_i \}. \quad (5.36)$$

The factor graph representing distribution (5.36) is given in Figure 5.15; notice that a new type of factor node is now present, connecting pairs of neighbouring variables.

Figure 5.15: Factor graph representing distribution (5.36). The factor nodes in dark gray seek to enforce correlations between neighboring pixels to assume the desired values.

The BP equations may be easily adapted for this model; we just need to add contributions from the new factor nodes. The messages incoming at these nodes are denoted by $h^{(2)}$, and those outgoing by $u^{(2)}$

$$\begin{cases} h_{i \rightarrow a}^{(1)} = Bx_{0i} + \sum_{j \in \partial i} u_{j \rightarrow i}^{(2)} + \sum_{b \in \partial i \setminus a} u_{b \rightarrow i}^{(1)}, \\ h_{i \rightarrow j}^{(2)} = Bx_{0i} + \sum_{a \in \partial i} u_{a \rightarrow i}^{(1)} + \sum_{k \in \partial i \setminus j} u_{k \rightarrow i}^{(2)}, \\ u_{a \rightarrow i}^{(1)} = \frac{1}{\beta} \tanh^{-1} \left[\tanh(\beta m_a) \prod_{j \in \partial a \setminus i} \tanh(\beta h_{j \rightarrow a}^{(1)}) \right], \\ u_{j \rightarrow i}^{(2)} = \frac{1}{\beta} \tanh^{-1} \left[\tanh(\beta \lambda x_{0i} x_{0j}) \tanh(\beta h_{i \rightarrow j}^{(2)}) \right]. \end{cases} \quad (5.37)$$

The marginals may now be computed from

$$P(x_i) \propto \exp \left\{ \beta \left(Bx_{0i} + \sum_{j \in \partial i} u_{j \rightarrow i}^{(2)} + \sum_{a \in \partial i} u_{a \rightarrow i}^{(1)} \right) x_i \right\}, \quad (5.38)$$

Results obtained by employing the decimation scheme together with the equations above are given in Figure 5.16. Notice that, while setting $\lambda > 0$ leads to smaller values of D_C (center), it also induces a sudden increase in the Hamming distortion D_H (left), and decrease in the fidelity f (right).

Figure 5.16: Performance of the decimation scheme over a more complicated model (5.36) that seeks to enforce correlations between neighbouring pixels of the image. As we increase λ , both the Hamming (left) and the correlation (center) distances decrease in average; the message is however transmitted with more and more errors (right). The factor graphs were sampled from 8-regular/Poisson ensembles, the components of x_0 from Bernoulli distributions with $p = 1/2$, and m was chosen to be all-one; B is set to 10^{-3} and β to 1.

5.3 Insights from statistical physics

The approaches we presented so far to both compression and steganography give rise to distributions of the form $P(x) \propto \exp \{ -\beta \mathcal{H}(x) \}$, for appropriately defined Hamiltonians $\mathcal{H}(x)$. In particular,

the generator matrix approach (5.22) allows us to write

$$\mathcal{H}_{\text{GM}}(z) = - \sum_{i=1}^N x_{0i} x_{mi}^* \prod_{\mu \in \partial i} z_{\mu}, \quad (5.39)$$

for a dynamic variable z and quenched variables x_0 and x_m^* . For the dual approach using parity-check matrices (5.29) we have

$$\mathcal{H}_{\text{PC}}(x) = - \sum_{a=1}^M m_a \prod_{i \in \partial a} x_i - B \sum_{i=1}^N x_{0i} x_i. \quad (5.40)$$

and the dynamic variable is now x , while the quenched ones are x_0 and m . Note that in both cases the factor graph is also random, as the matrices $\mathbb{H}$ and $\mathbb{G}$ that define them are sampled from certain ensembles. The compression problem discussed in subsection 5.1.2 is recovered if we set $x_{mi}^* = 1 \forall i$ in (5.39), or $m_a = 1 \forall a$ in (5.40).

We wanted at first to determine the minima of these Hamiltonians – the ground states of the associated physical systems – for a given instance of the problem, i.e. a given sample of the quenched variables. The algorithms we introduced in the last section serve this purpose. We are now interested in using techniques from statistical physics so as to study typical properties of these minima, given not a single instance but an ensemble of instances.

Model (5.39) has been recently studied in the statistical physics literature, although in the context of compression, where $x_{mi}^* = 1 \forall i$ and thus $\mathcal{H}_{\text{GM}}(z) = \sum_{i=1}^N x_{0i} \prod_{\mu \in \partial i} z_{\mu}$. By using either the replica [Mur04] or the cavity [CM05] method, one is able to compute averages over x_0 and the ensemble of factor graphs, thus determining the rate-distortion performance of that ensemble analytically. This analysis could in principle be extended to our case; note however that the distribution of x_m^* is rather complicated, as it is defined as being a particular solution of $\mathbb{H}x = m$, and $\mathbb{H}$ must be obtained by solving for $\mathbb{H}\mathbb{G} = 0$.

In what follows, we shall focus on model (5.40). In the next subsection, we will use the cavity method in the replica-symmetric approximation (see Chapter 3) in order to analytically determine average ground-state properties; this will allow us to reproduce the results obtained in the last section without the need of averaging over many trials.

5.3.1 Average ground-state energy

As discussed in Chapter 3, the RS cavity equation may be derived by simply looking at the fixed points of the BP equations for a random edge of the factor graph. Thus, from (5.30), we obtain

$$\begin{cases} h \stackrel{d}{=} Bx_0 + \sum_{a=1}^{c-1} u_a, \\ u \stackrel{d}{=} \text{sgn} \left(m \prod_{i=1}^{p-1} h_i \right) \min \left(\{1\} \cup \{|h_i|\}_{i=1, \dots, p-1} \right). \end{cases} \quad (5.41)$$

Recall that c and p are random variables that follow the degree distributions of the variable and factor nodes respectively. Given the equalities in distribution above, we determine $P(h)$ and $P(u)$ by means of a numerical MCMC procedure known as population dynamics (see Box 2.4). The average ground-state energy per spin as a function of the cavity fields and biases h and u is then given by (3.57), which in this case may be written as

$$\begin{aligned} \epsilon_0^{\text{RS}} = & -\alpha \left\{ 1 + 2 \mathbb{E}_{h,p,m} (p-1) \min(1, \{|h_i|\}_{i=1,\dots,p}) \mathbb{I} \left[m \prod_{i=1}^p h_i < 0 \right] \right\} \\ & - \mathbb{E}_{u,c,x_0} \left\{ \left| Bx_0 + \sum_{a=1}^c u_a \right| - \sum_{a=1}^c |u_a| \right\}, \end{aligned} \quad (5.42)$$

for $\alpha = \mathbb{E}_c c / \mathbb{E}_p p$. We may also write the average distortion as a function of h and u , following the cavity method prescriptions

$$D_{\text{avg}} = \mathbb{E}_{x_0,m,\mathcal{G}} \frac{1 - N^{-1} \sum_{i=1}^N \langle x_{0i} x_i \rangle}{2} = \frac{1}{2} \left\{ 1 - \mathbb{E}_{x_0,h} x_0 \operatorname{sgn}(h) \right\}, \quad (5.43)$$

and since $\epsilon_0^{\text{RS}} = \mathbb{E}_{x_0,m,\mathcal{G}} N^{-1} \langle \mathcal{H}_{\text{PC}} \rangle = \alpha(1 - 2f_{\text{avg}}) + B(2D_{\text{avg}} - 1)$, the average fidelity may be obtained from

$$f_{\text{avg}} = \frac{1}{2} - \frac{1}{2\alpha} \left\{ \epsilon_0^{\text{RS}} - B(2D_{\text{avg}} - 1) \right\} = \frac{1}{2} \left\{ 1 - \alpha^{-1} \epsilon_0^{\text{RS}} \Big|_{B=0} \right\}. \quad (5.44)$$

Let us consider for instance a k -regular/Poisson ensemble of factor graphs with $k = 8$, so that $p = 8$ and $c \sim \text{Poisson}(\alpha p)$. Moreover, we set $P(m) = \mathbb{B}_{\pm 1}(1/2)$, $P(x_0) = \mathbb{B}_{\pm 1}(1/2)$, $\alpha = 0.25$ and $B = 0.01$. We obtain approximate solutions to (5.41) by means of the population dynamics algorithm, and then proceed to compute the quantities above, attaining the following numerical values

$$\begin{aligned} \epsilon_0^{\text{RS}} &= -0.260 \pm 0.001, \\ D_{\text{avg}} &= 0.073 \pm 0.003, \\ f_{\text{avg}} &= 1.004 \pm 0.003, \end{aligned} \quad (5.45)$$

by using populations of size $N_{\text{pop}} = 2 \cdot 10^4$ and iterating over $t_{\text{max}} = 5 \cdot 10^4$ steps. In Figure 5.17, we compare these values to those obtained by the decimation scheme proposed in the last section using different sample sizes N .

Figure 5.17: Comparison between predictions of the cavity method (red line) and results obtained in multiple runs of the decimation procedure (black dots).

This procedure is easily generalizable for other forms of the distortion function, so that the same analysis can be performed for instance for model (5.36).

5.3.2 Perspectives

By means of the proposed scheme, one is able to embed a message into an image while incurring a distortion smaller than the usual, due to the fact that we allow for small embedding errors. We have also introduced a practical decimation algorithm based on belief propagation that performs the embedding and, through the cavity method, we were able to anticipate the performance of this algorithm.

Besides allowing us to predict the performance of the decimation algorithm with much less computational effort than by running it multiple times, statistical physics may also provide some insight into other matters. Below, we briefly outline some of the possibilities.

Replica symmetry-breaking

Even though in the example we gave the results anticipated by the cavity method were very close to those obtained by the decimation algorithm, this is not always the case. In particular for larger values of α , the system might undergo a glassy transition (see Chapter 3), after which the results provided by the RS hypothesis are not accurate[MM09; AMV15]. The performance of the BP algorithm is also affected, leading to worse results in the decimation. Determining the values of the parameters for which this transition happens might be useful, as it would help us to understand how the structure of the problem affects the algorithm's performance.

Codebook design

It would be interesting to try to gain some insight on what are the optimal codebook constructions, i.e. what are the constructions which allow us to achieve rate-capacity and why. This could be done in principle by looking at the so-called *weight enumerator function*[RU08; MU07], also known as the Franz-Parisi potential in statistical physics, which gives the distribution of the average distance between codewords, thus giving some intuition on the geometrical structure of the codebook.

Optimal distortion

It should also be possible, using techniques from statistical physics, to evaluate the minimum distortion that can be achieved by any code. While this is known when the Hamming distortion is used, in general for more complicated distortions it is not. In order to compute such minimum distortion, it should be sufficient to take the appropriate limit given a rate-capacity achieving ensemble[CM05; AMUV12].

Conclusion

An interdisciplinary field that is founded on statistics and borrows its methods from information theory, machine learning and statistical physics has been steadily emerging over the past couple of decades. We have tried in the first part of this thesis to survey its basics; as we have shown, the problems approached within this field belong essentially to one of two related, but fundamentally distinct, classes: statistical inference and combinatorial optimization.

We have then described two specific problems using the language provided by this field. The problem of compressed sensing, first introduced around a decade ago, was rewritten as a problem of inference. By doing this, we were able to design algorithms to solve it, to analyze the typical performance of these algorithms, and to compare the results obtained to the optimal achievable. While similar algorithms had already been introduced in the literature, they are known to suffer from a severe lack of convergence whenever ideal conditions are not met, in contrast to those we have proposed which are either provably shown or empirically verified to converge in all situations.

The same tasks were performed for the problem of information hiding; after describing it as a problem of optimization, we proposed algorithms and analyzed their performance – not only empirically but also theoretically, using tools from statistical physics. Differently from other approaches in the literature, the one we propose allows one to hide a message into an image by making fewer modifications than it is usually necessary; that is because we allow a few bits of the message to be lost in the process. The advantages and disadvantages of this approach have been analyzed using the forementioned tools.

Through applying this field's methods to analyze new and different problems, one promotes its growth – new algorithms and analysis techniques are introduced, and through time they become part of the practitioner's toolbox, which is then allowed to analyze a wider range of phenomena. We hope to have been able to contribute to the growth of this exciting emerging field.

Bibliography

- [AM00] S. Aji and R. McEliece, “The generalized distributive law,” *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 325–343, 2000. DOI: 10.1109/18.825794.
- [Ami92] D. J. Amit, *Modeling Brain Function: The World of Attractor Neural Networks*. Cambridge University Press, 1992, 524 pp.
- [AMUV12] V. Aref, N. Macris, R. Urbanke, and M. Vuffray, “Lossy source coding via spatially coupled LDGM ensembles,” *arXiv:1202.4959 [cond-mat]*, 2012. arXiv: 1202.4959.
- [AMV15] V. Aref, N. Macris, and M. Vuffray, “Approaching the rate-distortion limit with spatial coupling, belief propagation, and decimation,” *IEEE Transactions on Information Theory*, vol. 61, no. 7, pp. 3954–3979, 2015. DOI: 10.1109/TIT.2015.2434842.
- [Bar11] D. Barber, *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2011.
- [BDDW08] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, “A simple proof of the restricted isometry property for random matrices,” *Constructive Approximation*, vol. 28, no. 3, pp. 253–263, 2008. DOI: 10.1007/s00365-007-9003-x.
- [Bet35] H. A. Bethe, “Statistical theory of superlattices,” *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 150, no. 871, pp. 552–575, 1935.
- [Bis07] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2007.
- [Bis96] C. M. Bishop, *Neural Networks for Pattern Recognition*, 1 edition. Clarendon Press, 1996, 504 pp.
- [BKZ09] A. Braunstein, F. Kayhan, and R. Zecchina, “Efficient LDPC codes over $GF(q)$ for lossy data compression,” *arXiv:0901.4467 [cs, math]*, 2009. arXiv: 0901.4467.
- [BL07] C. M. Bishop and J. Lasserre, “Generative or discriminative? getting the best of both worlds,” in *Proceedings of the Valencia International Meetings on Bayesian Statistics 8*, 2007.
- [BMP08] A. Braunstein, R. Mulet, and A. Pagnani, “Estimating the size of the solution space of metabolic networks,” *BMC Bioinformatics*, vol. 9, no. 1, p. 240, 2008. DOI: 10.1186/1471-2105-9-240.
- [BP03] J.-P. Bouchaud and M. Potters, *Theory of Financial Risk and Derivative Pricing: From Statistical Physics to Risk Management*. Cambridge University Press, 2003, 408 pp.

- [BSB10] D. Baron, S. Sarvotham, and R. Baraniuk, "Bayesian compressive sensing via belief propagation," *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 269–280, 2010. DOI: 10.1109/TSP.2009.2027773.
- [BV04] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1 edition. Cambridge University Press, 2004, 727 pp.
- [BW87] J. D. Bryngelson and P. G. Wolynes, "Spin glasses and the statistical mechanics of protein folding," *Proceedings of the National Academy of Sciences*, vol. 84, no. 21, pp. 7524–7528, 1987.
- [CCCM10] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: survey and analysis of current methods," *Signal Processing*, vol. 90, no. 3, pp. 727–752, 2010. DOI: 10.1016/j.sigpro.2009.08.010.
- [CDS98] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998. DOI: 10.1137/S1064827596304010.
- [CFL09] C. Castellano, S. Fortunato, and V. Loreto, "Statistical physics of social dynamics," *Reviews of Modern Physics*, vol. 81, no. 2, pp. 591–646, 2009. DOI: 10.1103/RevModPhys.81.591.
- [Cha07] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007. DOI: 10.1109/LSP.2007.898300.
- [CKZ14] F. Caltagirone, F. Krzakala, and L. Zdeborová, "On convergence of approximate message passing," *arXiv:1401.6384 [cond-mat]*, 2014.
- [CM05] S. Ciliberti and M. Mezard, "The theoretical capacity of the parity source coder," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 10, P10003–P10003, 2005. DOI: 10.1088/1742-5468/2005/10/P10003. arXiv: cond-mat / 0506652.
- [Cox08] I. J. Cox, *Digital watermarking and steganography*. Morgan Kaufmann Publishers, 2008.
- [Cra98] R. Crandall. (1998). Some notes on steganography, Steganography mailing list.
- [CRT04] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *arXiv:math/0409186*, 2004. arXiv: math/0409186.
- [CT91] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley-Interscience, 1991, 576 pp.
- [CV11] N. Caticha and R. Vicente, "Agent-based social psychology: from neurocognitive processes to social data," *Advances in Complex Systems*, vol. 14, no. 5, pp. 711–731, 2011.
- [CW08] E. Candes and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008. DOI: 10.1109/MSP.2007.914731.

- [CWF14] B. Cakmak, O. Winther, and B. Fleury, "S-AMP: approximate message passing for general matrix ensembles," in *2014 IEEE Information Theory Workshop (ITW)*, 2014, pp. 192–196. DOI: 10.1109/ITW.2014.6970819.
- [DDD04] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004. DOI: 10.1002/cpa.20042.
- [DE03] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization," *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003. DOI: 10.1073/pnas.0437847100.
- [DE11] M. Duarte and Y. Eldar, "Structured compressed sensing: from theory to applications," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4053–4085, 2011. DOI: 10.1109/TSP.2011.2161982.
- [DET06] D. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions on Information Theory*, vol. 52, no. 1, pp. 6–18, 2006. DOI: 10.1109/TIT.2005.860430.
- [DHD12] A. Dremeau, C. Herzet, and L. Daudet, "Boltzmann machine and mean-field approximation for structured sparse decompositions," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3425–3438, 2012. DOI: 10.1109/TSP.2012.2192436.
- [DMM09] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proceedings of the National Academy of Sciences*, vol. 106, no. 45, pp. 18914–18919, 2009. DOI: 10.1073/pnas.0909892106.
- [DT05] D. L. Donoho and J. Tanner, "Neighborliness of randomly projected simplices in high dimensions," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 27, pp. 9452–9457, 2005. DOI: 10.1073/pnas.0502258102.
- [DT09] D. Donoho and J. Tanner, "Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1906, pp. 4273–4293, 2009. DOI: 10.1098/rsta.2009.0152.
- [DY00] A. Dragulescu and V. M. Yakovenko, "Statistical mechanics of money," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 17, no. 4, pp. 723–729, 2000. DOI: 10.1007/s100510070114.
- [EB01] A. Engel and C. v. d. Broeck, *Statistical mechanics of learning*. Cambridge University Press, 2001.
- [FF07a] T. Filler and J. Fridrich, "Binary quantization using belief propagation with decimation over factor graphs of LDGM codes," in *arXiv:0710.0192 [cs, math]*, 2007. arXiv: 0710.0192.
- [FF07b] J. Fridrich and T. Filler, "Practical methods for minimizing embedding impact in steganography," vol. 6505, 2007. DOI: 10.1117/12.697471.
- [FF10] T. Filler and J. Fridrich, "Gibbs construction in steganography," *Information Forensics and Security, IEEE Transactions on*, vol. 5, no. 4, pp. 705–720, 2010. DOI: 10.1109/TIFS.2010.2077629.

- [FH93] K. H. Fischer and J. Hertz, *Spin glasses*. Cambridge University Press, 1993.
- [FLS13] D. Fisher, M. Lässig, and B. Shraiman, “Evolutionary dynamics and statistical physics,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2013, no. 1, N01001, 2013. DOI: 10.1088/1742-5468/2013/01/N01001.
- [Fri10] J. Fridrich, *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2010.
- [Gal62] R. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962. DOI: 10.1109/TIT.1962.1057683.
- [GGS82] S. Galam, Y. Gefen, and Y. Shapir, “Sociophysics: a new approach of sociological collective behaviour. i. mean-behaviour description of a strike,” *Journal of Mathematical Sociology*, vol. 9, no. 1, pp. 1–13, 1982.
- [GHH14] D. Gamarnik, M. Hemery, and S. Hetterich, “Local algorithms for graphs,” *arXiv:1409.5214 [cond-mat]*, 2014. arXiv: 1409.5214.
- [GS10] S. Ganguli and H. Sompolinsky, “Statistical mechanics of compressed sensing,” *Physical Review Letters*, vol. 104, no. 18, p. 188 701, 2010. DOI: 10.1103/PhysRevLett.104.188701.
- [GT02] F. Guerra and F. L. Toninelli, “The thermodynamic limit in mean field spin glass models,” *Communications in Mathematical Physics*, vol. 230, no. 1, pp. 71–79, 2002. DOI: 10.1007/s00220-002-0699-y.
- [GW06] D. Guo and C.-C. Wang, “Asymptotic mean-square optimality of belief propagation for sparse linear systems,” in *IEEE Information Theory Workshop, 2006. ITW '06 Chengdu*, 2006, pp. 194–198. DOI: 10.1109/ITW2.2006.323786.
- [Hes03] T. Heskes, “Stable fixed points of loopy belief propagation are minima of the bethe free energy,” *Advances in neural information processing systems*, vol. 15, pp. 359–366, 2003.
- [Iba99] Y. Iba, “The nishimori line and bayesian statistics,” *Journal of Physics A: Mathematical and General*, vol. 32, no. 21, p. 3875, 1999. DOI: 10.1088/0305-4470/32/21/302.
- [IM09] A. T. Ihler and D. A. McAllester, “Particle belief propagation,” in *International Conference on Artificial Intelligence and Statistics*, 2009, pp. 256–263.
- [Jay03] E. T. Jaynes, *Probability theory the logic of science*. Cambridge University Press, 2003.
- [Jay57] E. T. Jaynes, “Information theory and statistical mechanics,” *Physical Review*, vol. 106, no. 4, pp. 620–630, 1957. DOI: 10.1103/PhysRev.106.620.
- [Jor04] M. I. Jordan, “Graphical models,” *Statistical Science*, vol. 19, no. 1, pp. 140–155, 2004.
- [KF09] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, 1 edition. The MIT Press, 2009, 1280 pp.
- [KFL06] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Trans. Inf. Theor.*, vol. 47, no. 2, pp. 498–519, 2006. DOI: 10.1109/18.910572.

- [KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983. DOI: 10.1126/science.220.4598.671.
- [KMR+07] F. Krzakala *et al.*, "Gibbs states and the set of solutions of random constraint satisfaction problems," *Proceedings of the National Academy of Sciences*, vol. 104, no. 25, pp. 10 318–10 323, 2007. DOI: 10.1073/pnas.0703685104.
- [KMS+12] F. Krzakala *et al.*, "Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices," *arXiv:1206.3953*, 2012, J. Stat. Mech. (2012) P08009. DOI: 10.1088/1742-5468/2012/08/P08009.
- [KMTZ14] F. Krzakala, A. Manoel, E. W. Tramel, and L. Zdeborova, "Variational free energies for compressed sensing," in *2014 IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2014.
- [KS04] Y. Kabashima and D. Saad, "Statistical mechanics of low-density parity-check codes," *Journal of Physics A: Mathematical and General*, vol. 37, no. 6, R1, 2004. DOI: 10.1088/0305-4470/37/6/R01.
- [KWT09] Y. Kabashima, T. Wadayama, and T. Tanaka, "A typical reconstruction limit for compressed sensing based on lp-norm minimization," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2009, no. 9, p. L09003, 2009. DOI: 10.1088/1742-5468/2009/09/L09003.
- [KZAC15] F. Krzakala, L. Zdeborova, M. C. Angelini, and F. Caltagirone, *Statistical physics of inference and bayesian estimation*, 2015.
- [LB12] J. Laska and R. Baraniuk, "Regime change: bit-depth versus measurement-rate in compressive sensing," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3496–3505, 2012. DOI: 10.1109/TSP.2012.2194710.
- [Mac03] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003, 640 pp.
- [Mal08] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, 2008, 829 pp.
- [MD10] A. Maleki and D. Donoho, "Optimally tuned iterative reconstruction algorithms for compressed sensing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 330–341, 2010. DOI: 10.1109/JSTSP.2009.2039176.
- [Min01] T. P. Minka, "Expectation propagation for approximate bayesian inference," in *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, ser. UAI'01, Morgan Kaufmann Publishers Inc., 2001, pp. 362–369.
- [MM09] M. Mezard and A. Montanari, *Information, Physics, and Computation*. Oxford University Press, 2009.
- [MM11] C. Moore and S. Mertens, *The Nature of Computation*, 1 edition. Oxford University Press, 2011, 985 pp.
- [Mon10] A. Montanari, "Graphical models concepts in compressed sensing," *arXiv e-print 1011.4328*, 2010.

- [MP01] M. Mézard and G. Parisi, "The bethe lattice spin glass revisited," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 20, no. 2, pp. 217–233, 2001. DOI: 10.1007/PL00011099.
- [MPV87] M. Mezard, G. Parisi, and M. A. Virasoro, *Spin Glass Theory and Beyond*. World Scientific, 1987, 461 pp.
- [MS99] R. N. Mantegna and H. E. Stanley, *Introduction to Econophysics: Correlations and Complexity in Finance*. Cambridge University Press, 1999, 164 pp.
- [MU07] A. Montanari and R. Urbanke, "Modern coding theory: the statistical mechanics and computer science point of view," *arXiv:0704.2857 [cond-mat]*, 2007. arXiv: 0704.2857.
- [Mur04] T. Murayama, "Thouless-anderson-palmer approach for lossy compression," *Physical Review E*, vol. 69, no. 3, p. 035 105, 2004. DOI: 10.1103/PhysRevE.69.035105.
- [Mur12] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012, 1104 pp.
- [MV13] A. Manoel and R. Vicente, "Statistical mechanics of reputation systems in autonomous networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2013, P08002 2013. DOI: 10.1088/1742-5468/2013/08/P08002.
- [MW06] E. Martinian and M. J. Wainwright, "Low density codes achieve the rate-distortion bound," *arXiv:cs/0601123*, 2006. arXiv: cs/0601123.
- [MWJ99] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: an empirical study," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'99, Morgan Kaufmann Publishers Inc., 1999, pp. 467–475.
- [MY04] E. Martinian and J. S. Yedidia, "Iterative quantization using codes on graphs," in *Proceedings of the 41st Annual Allerton Conference on Communication, Control, and Computing*, 2004. arXiv: cs/0408008.
- [Nis01] H. Nishimori, *Statistical physics of spin glasses and information processing an introduction*. Oxford Univ. Press, 2001.
- [NS01] H. Nishimori and D. Sherrington, "Absence of replica symmetry breaking in a region of the phase diagram of the ising spin glass," in *AIP Conference Proceedings*, vol. 553, AIP Publishing, 2001, pp. 67–72. DOI: 10.1063/1.1358165.
- [OS01] M. Oppor and D. Saad, *Advanced Mean Field Methods: Theory and Practice*. MIT Press, 2001.
- [OW05] M. Oppor and O. Winther, "Expectation consistent approximate inference," *J. Mach. Learn. Res.*, vol. 6, pp. 2177–2204, 2005.
- [Par83] G. Parisi, "Order parameter for spin-glasses," *Physical Review Letters*, vol. 50, no. 24, pp. 1946–1948, 1983. DOI: 10.1103/PhysRevLett.50.1946.
- [Pea82] J. Pearl, "Reverend bayes on inference engines: a distributed hierarchical approach," in *AAAI*, 1982, pp. 133–136.
- [PFB10] T. Pevný, T. Filler, and P. Bas, "Using high-dimensional image models to perform highly undetectable steganography," in *Information Hiding*, ser. Lecture Notes in Computer Science 6387, R. Böhme, P. W. L. Fong, and R. Safavi-Naini, Eds., Springer Berlin Heidelberg, 2010, pp. 161–177.

- [PS98] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Courier Corporation, 1998, 530 pp.
- [Ran10] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," *arXiv:1010.5141 [cs, math]*, 2010.
- [RFSK15] S. Rangan, A. K. Fletcher, P. Schniter, and U. Kamilov, "Inference for generalized linear models via alternating directions and bethe free energy minimization," *arXiv preprint arXiv:1501.01797*, 2015.
- [RKZF00] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, "Reputation systems," *Commun. ACM*, vol. 43, no. 12, pp. 45–48, 2000. DOI: 10.1145/355112.355122.
- [RMO15] J. Raymond, A. Manoel, and M. Opper, "Expectation propagation," in *Statistical Physics, Optimization, Inference, and Message-Passing Algorithms*, Oxford University Press, 2015.
- [RS09] F. Ricci-Tersenghi and G. Semerjian, "On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2009, no. 9, P09001, 2009. DOI: 10.1088/1742-5468/2009/09/P09001.
- [RSR+13] S. Rangan *et al.*, "Fixed points of generalized approximate message passing with arbitrary matrices," *arXiv e-print 1301.6295*, 2013.
- [RU08] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008, 590 pp.
- [Sal01] S. Salinas, *Introduction to Statistical Physics*. Springer, 2001, 478 pp.
- [SCLE12] K. Sjöstrand, L. H. Clemmensen, R. Larsen, and B. Ersbøll, "Spasm: a matlab toolbox for sparse statistical modeling," *Journal of Statistical Software Accepted for publication*, 2012.
- [Sha48] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.
- [Sha59] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec*, vol. 4, no. 142, p. 1, 1959.
- [SII+10] E. B. Sudderth *et al.*, "Nonparametric belief propagation," *Commun. ACM*, vol. 53, no. 10, pp. 95–103, 2010. DOI: 10.1145/1831407.1831431.
- [SK75] D. Sherrington and S. Kirkpatrick, "Solvable model of a spin-glass," *Physical Review Letters*, vol. 35, no. 26, pp. 1792–1796, 1975. DOI: 10.1103/PhysRevLett.35.1792.
- [Sou94] N. Sourlas, "Spin glasses, error-correcting codes and finite-temperature decoding," *EPL (Europhysics Letters)*, vol. 25, no. 3, p. 159, 1994. DOI: 10.1209/0295-5075/25/3/001.
- [SS06] D. Sivia and J. Skilling, *Data Analysis: A Bayesian Tutorial*, 2 edition. Oxford University Press, 2006, 264 pp.
- [Str03] G. Strang, *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 2003, 584 pp.
- [SWW08] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky, *Loop series and Bethe variational bounds in attractive graphical models*. 2008.

- [TAP77] D. J. Thouless, P. W. Anderson, and R. G. Palmer, "Solution of 'solvable model of a spin glass'," *Philosophical Magazine*, vol. 35, no. 3, pp. 593–601, 1977. DOI: 10.1080/14786437708235992.
- [Tib96] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [vBF07] E. van den Berg and M. P. Friedlander, "SPGL1: a solver for large-scale sparse reconstruction," *Online: <http://www.cs.ubc.ca/labs/scl/spgl1>*, 2007.
- [VSK02] R. Vicente, D. Saad, and Y. Kabashima, "Low-density parity-check codes: a statistical physics perspective," *Advances in Imaging and Electron Physics*, vol. 125, pp. 232–355, 2002.
- [VSR+14] J. Vila *et al.*, "Adaptive damping and mean removal for the generalized approximate message passing algorithm," *arXiv:1412.2005 [cs, math]*, 2014. arXiv: 1412.2005.
- [Was04] L. Wasserman, *All of Statistics: A Concise Course in Statistical Inference*, 1st Corrected ed. 20 edition. Springer, 2004, 442 pp.
- [Wei48] P. R. Weiss, "The application of the bethe-peierls method to ferromagnetism," *Physical Review*, vol. 74, no. 10, pp. 1493–1504, 1948. DOI: 10.1103/PhysRev.74.1493.
- [WF01] Y. Weiss and W. T. Freeman, "Correctness of belief propagation in gaussian graphical models of arbitrary topology," *Neural Computation*, vol. 13, no. 10, pp. 2173–2200, 2001. DOI: 10.1162/089976601750541769.
- [WJ08] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Found. Trends Mach. Learn.*, vol. 1, no. 1, pp. 1–305, 2008. DOI: 10.1561/22000000001.
- [WM05] M. J. Wainwright and E. Maneva, "Lossy source encoding via message-passing and decimation over generalized codewords of LDGM codes," *arXiv:cs/0508068*, 2005. arXiv: cs/0508068.
- [WMM10] M. Wainwright, E. Maneva, and E. Martinian, "Lossy source compression using low-density generator matrix codes: analysis and algorithms," *IEEE Transactions on Information Theory*, vol. 56, no. 3, pp. 1351–1368, 2010. DOI: 10.1109/TIT.2009.2039160.
- [WP00] A. Westfeld and A. Pfitzmann, "Attacks on steganographic systems," in *Information Hiding*, ser. Lecture Notes in Computer Science 1768, A. Pfitzmann, Ed., Springer Berlin Heidelberg, 2000, pp. 61–76.
- [XK13] Y. Xu and Y. Kabashima, "Statistical mechanics approach to 1-bit compressed sensing," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2013, no. 2, P02041, 2013. DOI: 10.1088/1742-5468/2013/02/P02041.
- [XKZ14] Y. Xu, Y. Kabashima, and L. Zdeborová, "Bayesian signal reconstruction for 1-bit compressed sensing," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2014, no. 11, P11015, 2014. DOI: 10.1088/1742-5468/2014/11/P11015.
- [YFW00] J. S. Yedidia, W. Freeman, and Y. Weiss, "Generalized belief propagation," in *Advances in Neural Information Processing Systems 13*, 2000.

- [YFW01] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Bethe free energy, kikuchi approximations and belief propagation algorithms," Mitsubishi Electric Research Laboratories, TR2001-16, 2001.
- [YFW05] J. S. Yedidia, W. Freeman, and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2282–2312, 2005. DOI: 10.1109/TIT.2005.850085.
- [Yui02] A. L. Yuille, "CCCP algorithms to minimize the bethe and kikuchi free energies: convergent alternatives to belief propagation," *Neural Computation*, vol. 14, no. 7, pp. 1691–1722, 2002. DOI: 10.1162/08997660260028674.
- [ZAD12] H. Zhou, A. Armagan, and D. B. Dunson, "Path following and empirical bayes model selection for sparse regression," *arXiv:1201.3528 [stat]*, 2012. arXiv: 1201.3528.
- [Zam10] F. Zamponi, "Mean field theory of spin glasses," *arXiv:1008.4844 [cond-mat]*, 2010. arXiv: 1008.4844.
- [ZBLN97] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-b: fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, 1997. DOI: 10.1145/279232.279236.
- [Zde09] L. Zdeborová, "Statistical physics of hard optimization problems," *Acta Physica Slovaca*, vol. 59, no. 3, pp. 169–303, 2009. DOI: 10.2478/v10155-010-0096-6.
- [ZKMZ13] P. Zhang, F. Krzakala, M. Mezard, and L. Zdeborova, "Non-adaptive pooling strategies for detection of rare faulty items," in *2013 IEEE International Conference on Communications Workshops (ICC)*, 2013, pp. 1409–1414. DOI: 10.1109/ICCW.2013.6649458.
- [Zou06] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006. DOI: 10.1198/016214506000000735.