Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Physics (37^{th}cycle)

The cavity method for in and out of equilibrium systems on diluted graphs

By

Stefano Crotti

******

Supervisor(s):
Prof. Alfredo Braunstein

Doctoral Examination Committee:
Prof. Silvio Franz, Referee, Université Paris-Sud
Prof. Roberto Mulet, Referee, Universidad de La Habana

Politecnico di Torino
2025

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Stefano Crotti
2025

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

"Once in a lifetime I'll flip the coin
'cause the way that we go is a way that we already know"

Giant Rooks

Acknowledgements

To my supervisor AB for being there for me at any time, for showing me the value of understanding things in simple terms or not understanding them at all, for teaching me how to write fast code.

To the crew, MM, TB, EF, LS, for all the lunch breaks, coffee breaks, mid-afternoon breaks, terrace breaks, summer breaks, mid-semester breaks.

To Replica, the most serious thing I have done in these four years.

To DMP and RM for welcoming me in their crazy Havana, to JPM for the camping trip. To all the people this PhD allowed me to meet all around the world.

To FC for the dinner we spent ignoring the famous professors at our table, talking about our feelings instead.

To LDB for being the person I can turn to, always, and for encouraging me to travel to China.

To SP for reminding me that we can choose to do in life what we like the most.

To my parents, because.

Abstract

This work explores the cavity method from statistical physics as applied to systems of variables interacting on diluted graphs. The cavity approximation, originally conceived for studying disordered systems like spin glasses, has evolved into a versatile framework, providing message-passing algorithms such as belief propagation. Here we consider both static (equilibrium) and dynamic (non-equilibrium) scenarios, exploiting the effectiveness of the cavity approach in its analytical as well as algorithmic variations.

The first chapter introduces the mathematical formulation of the cavity method as originated in physics and then re-invented in the fields of communications and computer science.

In chapter two we study the closest vector problem, a prototypical task in discrete optimization with applications in cryptography and the theory of computational complexity. By means of a mapping to a spin glass model, the cavity method provides semi-analytical results for an infinite-size system. These are then compared with the outcome of approximate optimization on finite instances. Of particular interest is the interplay between phase transitions in the geometry of the space of solutions and the performance of optimization algorithms.

In the third chapter, devoted to non-equilibrium dynamics, we introduce and develop the Matrix Product Belief Propagation (MPBP) algorithm for computing observables of Markov processes on graphs. We consider how reweighting a process, an operation which arises naturally when considering inverse problems and atypical trajectories, can lead to major computational challenges. By leveraging the matrix product state approximation from quantum mechanics, MPBP reduces computational complexity while maintaining high accuracy for dynamical models such as epidemic spreading and Glauber dynamics.

Finally, chapter four contains ongoing work on an extension of MPBP able to describe the steady state of stochastic processes on graphs. This is done by exploiting the Uniform Matrix Product State (UMPS) formalism originally formulated for homogeneous one-dimensional quantum systems.

Co-authored papers

- Alfredo Braunstein, Louise Budzynski, Stefano Crotti, and Federico Ricci-Tersenghi

  Closest-vector problem and the zero-temperature p-spin landscape for lossy compression

  Physical Review E 106, 054101 (2022)

  arXiv:2207.00504

- Stefano Crotti and Alfredo Braunstein

  Matrix Product Belief Propagation for reweighted stochastic dynamics over graphs

  Proceedings of the National Academy of Sciences, 120(47), e2307935120

  arXiv:2303.17403

- Stefano Crotti, Thomas Barthel, and Alfredo Braunstein

  Nonequilibrium steady-state dynamics of Markov processes on graphs

  arXiv:2411.19100

Open source software

The code to reproduce the results in this manuscript is publicly available at the following repositories:

- Closest Vector Problem https://github.com/stecrotti/sourcecoding
- Matrix Product Belief Propagation https://github.com/stecrotti/MatrixProductBP.jl

A few Julia packages, developed and maintained by the author together with supervisor prof. Alfredo Braunstein, provide some core utilities:

- IndexedGraphs.jl https://github.com/stecrotti/IndexedGraphs.jl
- TensorTrains.jl https://github.com/stecrotti/TensorTrains.jl

Finally, a generic implementation of belief propagation can be found at

- BeliefPropagation.jl https://github.com/stecrotti/BeliefPropagation.jl

Contents

Chapter 1

The cavity method

The cavity method is a powerful tool that originated in statistical physics to deal with complex problems in disordered systems. It was first developed to tackle systems like spin glasses, where traditional methods, like mean-field approximations, often failed to capture the full picture. In its original form, the method was devised to describe systems made of a large homogeneous bulk comprised of a virtually infinite number of microscopic degrees of freedom. Later, the same approach was re-discovered independently in the telecommunications and computer science communities, where it assumed a more algorithmic character: it was used to compute properties of a finite number of elementary degrees of freedom, ruled by potentially heterogeneous interactions.

Here we will recall some basic history and mathematical foundations of the method. For an in-depth treatment see, e.g. [1, 2].

1.1 A non-exhaustive history of the cavity method

1.1.1 The Bethe-Peierls Approximation

The method has its roots back in the 1930s when Hans Bethe and Rudolf Peierls introduced what we now call the Bethe-Peierls approximation [3], a significant step in understanding how particles on a lattice interact with their nearest neighbors. In particular, they were interesting in studying a simple model of magnetism where

binary variables $\{\pm 1\}$ representing classical spins, i.e. magnetic dipoles, are attracted by the surrounding ones through some potential. In the simplified mathematical representation, each spin is connected to its few nearest neighbors, thus forming a graph of interactions. According to the axioms of statistical physics, the state of such system at equilibrium is described by a Boltzmann distribution which depends on the internal energy of the system when found in a given configuration of $+1$'s and -1's. To compute interesting physical quantities such as the magnetization, i.e. the proportion of positively oriented spins, one must first marginalize the Boltzmann distribution. To do so in an exact fashion is a computationally prohibitive task, hence the need for approximate methods that would return a good estimate while reducing the computational load.

Bethe and Pierls proposed a strategy which consists in assuming that any two spins sharing a interaction only influence each other through that direct interaction. In general this is true only approximately, as the two spins may very well be interacting via the mediating action of other surrounding spins. However, if there are no loops in the structure of interactions, in other words if the underlying graph is a tree, then the hypothesis of direct interaction turns out to be exact. A consequence of the hypothesis is that if one imagines to remove a single variable from the system, thus creating a *cavity*, its neighbors become statistically independent, allowing for a simplified mathematical treatment. Once the behavior of the system without the spin is understood, the spin is reintroduced, and its interactions with its neighbors are taken into account. This gives a recursive way to estimate properties like local magnetization or correlations.

1.1.2 The independent discovery of an algorithmic version

While the cavity method emerged from the statistical physics community, a parallel development occurred independently in the field of computer science, specifically in the study of probabilistic graphical models. This algorithm, known as belief propagation (BP), was introduced by Judea Pearl in the 1980s as a message-passing algorithm to perform inference on Bayesian networks, graphical models representing probabilistic dependencies between random variables [4].

In belief propagation, nodes in a graph represent random variables, and edges represent pairwise interactions between them. These nodes iteratively pass "messages"

to each other, updating their beliefs about the states of neighboring nodes based on conditional probabilities. The algorithm computes marginal probabilities of each variable by iterating over these local conditional probabilities. As it turns out, the approximation made by BP is equivalent to the cavity approach. The difference, which has more to do with nomenclature and conventions than with essence, is that physicists tend to be interested in infinite-size systems, the so-called thermodynamic limit. The parameters describing the interactions are either homogeneous throughout the system, or described by some probability distribution. On the other hand, in computer science problems, the graph of interactions is typically finite and given and the parameters heterogeneous.

In the 90's, following a revival of Gallager's work on Low Density Parity Check Codes [5], belief propagation was successfully applied to the problem of reliable communication over noisy information channels [6, 7]. The statistical physics community contributed to the analytical study of these topics [8], for a review see [9]. About at the same time, the cavity method and its algorithmic declinations found application in the analysis of constrained satisfaction problems (CSPs) from computer science [10–14]. These problems are particularly interesting in that they play a prominent role in the theory of complexity.

Many other phenomena have successfully been studied via BP or its variants: without being extensive we mention compressed sensing [15], learning in neural networks [16], and matrix inversion [17].

In many of the applications mentioned here, the cavity method features a strong connection with replica theory. Although beyond the scope of this work, it is worth mentioning that this parallelism has been widely acknowledged, to the point that the two often give the same results, and are in some cases considered as the same level of approximation [18, 19].

Finally, it is worth noting that BP can be seen as a special case of the expectation propagation approximation when performed with a fully factorized trial distribution [20].

1.2 Mathematical description

1.2.1 Factor Graphs and graphical models

Factor graphs are a powerful graphical representation that play a central role in probabilistic inference and the belief propagation algorithm. A factor graph $\mathscr{G} = ((V,F),E)$ is comprised of two sets of nodes, variable nodes $V = \{1,2,\ldots,N\}$ and factor nodes $F = \{1,2,\ldots,M\}$, and a set of edges $E \subset \{(ia)|i \in V, a \in F\}$. It is a bipartite graph as there are no edges connecting two variable nodes or two factor nodes. Variable nodes are used to represent the degrees of freedom of interest. Factor node represent functions, or interactions, among the variables they are connected to.

Factor graphs are useful to represent the structure of the local interactions between variables in a multivariate function such as a probability distribution. Consider a set of variables $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ and a global function $f(\mathbf{x})$ that can be factored into a product of local functions $f_a(\mathbf{x}_a)$ over subsets of variables $\mathbf{x}_a = \{x_i : (ia) \in E\}$. This factorization can be written as:

$$f(\mathbf{x}) = \prod_{a \in F} f_a(\mathbf{x}_a), \tag{1.1}$$

where each $f_a(\mathbf{x}_a)$ is a local function (factor) that depends on a subset of variables $\mathbf{x}_a$.

A factor graph provides a visual representation of this factorization, where:

- Each variable x_i is represented as a variable node.
- Each local function f_a is represented as a factor node.
- Edges connect each factor node f_a to the variable nodes x_i on which f_a depends.

As an example, consider a function of three variables x_1, x_2, x_3 that factors as:

$$f(x_1,x_2,x_3) = f_{12}(x_1,x_2) f_{23}(x_2,x_3). \tag{1.2}$$

The corresponding factor graph is depicted in fig 1.1. Factor nodes are usually represented by boxes, variable nodes by circles.

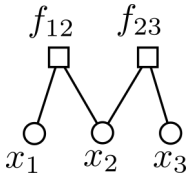


Fig. 1.1 A factor graph for the function $f(x_1,x_2,x_3) = f_{12}(x_1,x_2)f_{23}(x_2,x_3)$.

Factor graphs are particularly useful in situations where computation is performed over a large number of variables, especially when the system exhibits a certain sparsity structure that allows for efficient inference algorithms. In particular, Boltzmann distributions from statistical physics $p(\mathbf{x}) \propto e^{-\beta H(\mathbf{x})}$ are suited for such representations whenever the Hamiltonian H is a sum of local terms $H(\mathbf{x}) = \sum_a H_a(\mathbf{x}_a)$ each involving a small number of variables. After exponentiation one recovers

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{a \in F} \psi_a(\mathbf{x}_a), \tag{1.3}$$

which is of the form (1.1), with $\psi_a \coloneqq e^{-\beta H_a}$. Here β is a real parameter which in physics is the inverse of the temperature. It can be safely set to 1 in cases where there is no notion of temperature.

1.2.2 The belief propagation Algorithm

The belief propagation (BP) algorithm is a message-passing algorithm used to approximately compute marginals of a distribution of the form (1.3) factorized on a factor graph $\mathscr{G} = ((V \cup F), E)$. For each variable node $i \in V$, we denote by $\partial i = \{a \in F : (ia) \in E\}$ the set of factor nodes connected to i, and similarly for each factor node $a \in F$ the set of variable nodes connected to a: $\partial a = \{i \in V : (ia) \in E\}$. We introduce messages $m_{i \to a}$ and $m_{a \to i}$ on each edge $(i,a) \in E$ as the marginal probability laws of x_i in the cavity graph where some interactions are discarded: $m_{i \to a}(x_i)$ is the marginal of x_i when factor a is removed, and $m_{a \to i}(x_i)$ is the marginal of x_i when one removes all the factors in $\partial i \setminus a$. The BP messages obey the following

recursive equations:

$$\begin{aligned} m_{i\to a}(x_i) &= \frac{1}{z_{i\to a}} \prod_{b\in\partial i\setminus a} m_{b\to i}(x_i) \\ m_{a\to i}(x_i) &= \frac{1}{z_{a\to i}} \sum_{\mathbf{x}_{\partial a\setminus i}} \psi_a(\mathbf{x}_a) \prod_{j\in\partial a\setminus i} m_{j\to a}(x_j) \end{aligned} \tag{1.4}$$

where $z_{i\to a}, z_{a\to i}$ are normalization factors. On a tree, i.e. a graph with no loops, the equations above can be solved by sweeping through the graph twice, once inwards from the leaves to the core and once outwards. Also, and perhaps more importantly, messages computed as above can be used to obtain marginals $p_i(x_i) = \sum_{\{x_j\}_{j\neq i}} p(\mathbf{x})$ of the probability distribution under consideration.

On the other hand, on a graph with loops, the BP algorithm is in general not exact. In this case, one initializes the messages to some random values and then iterates (1.4) until a fixed point is found. On a fixed point, messages are used to compute *beliefs*, which are estimates for the true marginals. The extent to which BP on a loopy graph is wrong is often hard to quantify, although there exist rigorous results for some specific instances [21–25]. A brief summary of the limitations of the cavity assumption on graphs with loops is given later in section 1.2.6.

The calculations of observables presented in the following assume messages to be at a fixed point, and it will not in general provide an exact answer unless on a tree. The belief for a generic variable x_i is given by

$$p_i(x_i) = \frac{1}{z_i} \prod_{a\in\partial i} m_{a\to i}(x_i) \tag{1.5}$$

where z_i is the constant necessary to properly normalize to a probability distribution.

The joint distribution for the set of variables incident on factor a, *factor belief*, is given by

$$p_a(\mathbf{x}_a) = \frac{1}{z_a} \psi_a(\mathbf{x}_a) \prod_{j\in\partial a} m_{j\to a}(x_j) \tag{1.6}$$

where again z_a is the appropriate normalization constant.

For later convenience, let us define quantities

$$z_{ai} = \sum_{x_i} m_{a\to i}(x_i) m_{i\to a}(x_i) \tag{1.7}$$

on every edge $(ai) \in E$. It will be useful in the following to express message normalizations $z_{i\to a}, z_{a\to i}$ in terms of z_{ai}:

$$\begin{aligned} b_i(x_i) &= \sum_{\mathbf{x}_{a\setminus i}} b_a(\mathbf{x}_a) \\ &= \sum_{\mathbf{x}_{a\setminus i}} \frac{1}{z_a} \Psi_a(\mathbf{x}_a) \prod_{i\in\partial a} m_{ia}(x_i) \\ &= \frac{1}{z_a} \underbrace{\sum_{\mathbf{x}_{a\setminus i}} \Psi_a(\mathbf{x}_a) \prod_{j\in\partial a\setminus i} m_{ja}(x_j)}_{z_{a\to i} m_{ai}(x_i)} m_{ia}(x_i) \\ &= \frac{z_{a\to i}}{z_a} m_{ai}(x_i) m_{ia}(x_i). \end{aligned} \tag{1.8}$$

Since $b_i(x_i)$ is normalized by construction,

$$1 = \sum_{x_i} b_i(x_i) = \frac{z_{a\to i}}{z_a} z_{ai} \tag{1.9}$$

and therefore

$$z_{a\to i} = \frac{z_a}{z_{ai}}. \tag{1.10}$$

Similarly, in order to express also $z_{i\to a}$ in terms of z_{ai}, one can re-write the expression for b_i separating the a-th incoming message

$$\begin{aligned} b_i(x_i) &= \frac{1}{z_i} \underbrace{\prod_{b\in\partial i\setminus a} m_{bi}(x_i)}_{z_{i\to a} m_{ia}(x_i)} m_{ai}(x_i) \\ &= \frac{z_{i\to a}}{z_i} m_{ia}(x_i) m_{ai}(x_i). \end{aligned} \tag{1.11}$$

Again imposing normalization for b_i, one gets

$$1 = \sum_{x_i} b_i(x_i) = \frac{z_{i\to a}}{z_i} z_{ai} \tag{1.12}$$

giving

$$z_{i\to a} = \frac{z_i}{z_{ai}}. \tag{1.13}$$

Let us mention that for graphs where interactions are pairwise, i.e. factor nodes have at most two neighbors, the BP equations simplify considerably. We will exploit such simplifications in some examples later in this chapter, as well as in chapter 3.

1.2.3 The Bethe Free Energy

BP also provides an approximation to the free energy associated to the distribution (1.3), defined as $F = -\frac{1}{\beta}\log Z$. Motivated by how one can re-write the free energy in cases where the distribution is factorized on a tree, the Bethe Free Energy as a function of node and factor beliefs $F^{Bethe}[\mathbf{b}]$ is defined by [1]

$$-\beta F^{Bethe}[\mathbf{b}] = -\sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \log\left[\frac{b_a(\mathbf{x}_a)}{\Psi_a(\mathbf{x}_a)}\right] - \sum_i (1-|\partial i|)\sum_{x_i} b_i(x_i)\log\left[b_i(x_i)\right] \tag{1.14}$$

where $|\partial i|$ is the degree, i.e. the number of neighbors, of node i.

It is possible to derive an expression $F^{Bethe}[\mathbf{m}]$ for the free energy only in terms of messages. This is found to be a more convenient form when one wants to, for instance, compute the average free energy over distributions of messages obtained via the cavity method. We report here the full derivation since it is sometimes omitted in textbooks. Substituting the definitions for node and factor beliefs in the logs in (1.14),

$$\begin{aligned}\log\left[\frac{b_a(\mathbf{x}_a)}{\Psi_a(\mathbf{x}_a)}\right] &= \log\left[\frac{1}{z_a}\prod_{i\in\partial a} m_{ia}(x_i)\right] \\ &= \sum_{i\in\partial a}\log m_{ia}(x_i) - \log z_a\end{aligned} \tag{1.15}$$

$$\log b_i(x_i) = \sum_{a\in\partial i}\log m_{ai}(x_i) - \log z_i \tag{1.16}$$

The first term in (1.14) is

$$\begin{aligned}-\sum_a\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\log\left[\frac{b_a(\mathbf{x}_a)}{\Psi_a(\mathbf{x}_a)}\right] &= -\sum_a\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\sum_{i\in\partial a}\log m_{ia}(x_i) + \sum_a \underbrace{\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)}_{1}\log z_a \\ &= -\sum_a\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\sum_{i\in\partial a}\log m_{ia}(x_i) + \sum_a \log z_a\end{aligned} \tag{1.17}$$

The second is

$$-\sum_i (1-|\partial i|)\sum_{x_i} b_i(x_i)\log\left[b_i(x_i)\right] =$$

$$\sum_i (1-|\partial i|)\left[-\sum_{x_i} b_i(x_i)\sum_{a\in\partial i}\log m_{ai}(x_i)+\underbrace{\sum_{x_i} b_i(x_i)}_{1}\log z_i\right] = \tag{1.18}$$

$$\sum_i (1-|\partial i|)\left[-\sum_{x_i} b_i(x_i)\sum_{a\in\partial i}\log m_{ai}(x_i)+\log z_i\right].$$

At this point, the expression in (1.14) reads

$$\begin{aligned}-\beta F^{Bethe} =& \sum_a \log z_a + \sum_i (1-|\partial i|)\log z_i - \\ & \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)\sum_{i\in\partial a}\log m_{ia}(x_i) - \sum_i (1-|\partial i|)\sum_{x_i} b_i(x_i)\sum_{a\in\partial i}\log m_{ai}(x_i).\end{aligned} \tag{1.19}$$

Substituting further,

$$\begin{aligned}\sum_i |\partial i|\log z_i &= \sum_i |\partial i|\log\left[\sum_{x_i}\prod_{b\in\partial i} m_{bi}(x_i)\right] \\ &= \sum_i\sum_{a\in\partial i}\log\left[\sum_{x_i} m_{ai}(x_i)\underbrace{\prod_{b\in\partial i\setminus a} m_{bi}(x_i)}_{m_{ia}(x_i)z_{i\to a}}\right] \\ &= \sum_i\sum_{a\in\partial i}\log\left[\underbrace{\sum_{x_i} m_{ai}(x_i)m_{ia}(x_i)}_{z_{ai}} z_{i\to a}\right] \\ &= \sum_{\langle i,a\rangle}\left(\log z_{ai}+\log z_{i\to a}\right).\end{aligned} \tag{1.20}$$

Now

$$-\beta F^{Bethe} = \sum_a \log z_a + \sum_i \log z_i - \sum_{\langle i,a\rangle} \log z_{ai} - \sum_{\langle i,a\rangle} \log z_{i\to a} \\ - \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \log m_{ia}(x_i) - \sum_i (1-|\partial i|) \sum_{x_i} b_i(x_i) \sum_{a\in\partial i} \log m_{ai}(x_i). \tag{1.21}$$

Expanding $\log m_{ia}(x_i)$ in the fourth term,

$$\begin{aligned} \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \log m_{ia}(x_i) = \\ \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \log m_{bi}(x_i) - \sum_a \underbrace{\sum_{\mathbf{x}_a} b_a(\mathbf{x}_a)}_{1} \sum_{i\in\partial a} \log z_{i\to a} = \\ \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \log m_{bi}(x_i) - \sum_{\langle i,a\rangle} \log z_{i\to a} \end{aligned} \tag{1.22}$$

therefore

$$\begin{aligned} -\beta F^{Bethe} = \sum_a \log z_a + \sum_i \log z_i - \sum_{\langle i,a\rangle} \log z_{ai} - \cancel{\sum_{\langle i,a\rangle} \log z_{i\to a}} \\ - \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \log m_{bi}(x_i) + \cancel{\sum_{\langle i,a\rangle} \log z_{i\to a}} \\ - \sum_i (1-|\partial i|) \sum_{x_i} b_i(x_i) \sum_{a\in\partial i} \log m_{ai}(x_i). \end{aligned} \tag{1.23}$$

Now the goal is to get rid of the two last remaining terms. We re-arrange the sums in the fourth term in order to recover the definition of b_i as marginalization of

b_a

$$-\sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \log m_{bi}(x_i) = -\sum_a \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \sum_{x_i} \underbrace{\sum_{\mathbf{x}_{a\setminus i}} b_a(\mathbf{x}_a)}_{b_i(x_i)} \log m_{bi}(x_i)$$

$$= -\sum_a \sum_{i\in\partial a} \sum_{b\in\partial i\setminus a} \sum_{x_i} b_i(x_i) \log m_{bi}(x_i)$$

We can substitute $\sum_a \sum_{i\in\partial a}$ with $\sum_i \sum_{a\in\partial i}$ since they both count each edge exactly once

$$= -\sum_i \sum_{a\in\partial i} \sum_{b\in\partial i\setminus a} \sum_{x_i} b_i(x_i) \log m_{bi}(x_i). \tag{1.24}$$

Now the fourth and fifth terms of (1.23) look like

$$-\sum_i \sum_{a\in\partial i} \sum_{b\in\partial i\setminus a} \sum_{x_i} b_i(x_i) \log m_{bi}(x_i) - \sum_i (1-|\partial i|) \sum_{a\in\partial i} \sum_{x_i} b_i(x_i) \log m_{ai}(x_i) \tag{1.25}$$

which gives zero, because the quantity $\sum_{x_i} b_i(x_i) \log m_{bi}(x_i)$ is counted $(|\partial i|-1)$ times by the summation $\sum_{b\in\partial i\setminus a}$ over all neighbors of i except one (a).

Finally, we get the free energy in terms of messages

$$F^{Bethe}[\mathbf{m}] = -\frac{1}{\beta}\left\{\sum_a \log z_a\left[\{m_{ia}\}_{i\in\partial a}, \Psi_a\right] + \sum_i \log z_i\left[\{m_{ai}\}_{a\in\partial i}\right] - \sum_{(i,a)} \log z_{ai}\left[m_{ai}, m_{ia}\right]\right\}. \tag{1.26}$$

It is worth noting that the expression above contains all terms that are by-products of the computation of messages and beliefs. Hence, the calculation of the free energy can be conveniently embedded in the one for the update of messages.

As a final remark, although not shown here, it is possible to show that extermizing the Bethe Free Energy with respect to the messages while imposing the appropriate normalization and compatibility conditions, recovers the BP equations [2]. This provides an interpretation for the BP algorithm with a variational flavor.

1.2.4 Infinite graphs

Belief Propagation naturally lends itself to generalizations in several directions.

Infinite size

Physicists are often interested in studying systems in the thermodynamic limit of an infinite number of degrees of freedom. This often leads to analytical simplifications, besides being a good approximation in many cases where in practice one is interested in working with very large systems, such as in error correction where codes of large block length are known to perform well [26]. One obvious strategy is to work with large enough instances that approximate the thermodynamic limit. However, the cavity approximation allows for a more efficient approach.

A classic example of infinite-size is the planar ferromagnetic Ising model, described by the distribution

$$p(\sigma) \propto e^{\beta J \sum_{(ij)} \sigma_i \sigma_j} \tag{1.27}$$

where $\{(ij)\}$ are the edges of a square lattice extending indefinitely in space. In this case, one can imagine to apply the BP approximation by computing messages on all the edges. Because of the symmetry of the graph and the homogeneity of the J coupling, it is not hard to be convinced that at a fixed point, all messages should be equal. It is therefore enough to write a fixed point equation[1] for a single message

$$m(\sigma) \propto \left[\sum_{\sigma'} e^{J\sigma\sigma'} m(\sigma') \right]^3 . \tag{1.28}$$

Disorder

Suppose now that the situation is a little more complicated and, while each node in the graph still has the same number of neighbors, the coupling constants are now edge dependent J_{ij} and drawn independently from some probability distribution p_J. The situation where parameters of the problem are sampled from a probability distribution is what people refer to as *disorder*.

One can still avoid instantiating one or many large finite instances of the system. The idea, going sometimes under the name *density evolution* [1], is to consider the probability distribution over the messages induced by the distribution of the J's.

[1] Here we used the simplified expression of the BP equations for pairwise interactions, where usually only one type of messages (here variable $\to$ factor) is used.

Provided that two independence assumptions hold, exactly or approximately: (1) The J's are sampled independently on each edge (2) The cavity hypothesis that fixing the value of a node decouples its neighbors, one can write a BP-like equation for the distribution of messages. This looks like

$$\mathscr{P}\left[m(\sigma)\right] \propto \int Dm_1 \cdots Dm_{k-1} \mathbb{1}\left[m(\sigma) \propto \prod_{j=1}^{k-1} \sum_{\sigma_j} e^{J\sigma\sigma_j} m_j(\sigma_j)\right] \prod_{j=1}^{k-1} \mathscr{P}\left[m_j(\sigma_j)\right] \tag{1.29}$$

where by Dm_j we mean an integral over all possible values taken by a message. This is usually done only in specific cases where messages can be completely specified by a single real, or even integer (see, e.g. [12] or section 2.3)).

Finally, disorder can also appear in the structure of the graph. In graph ensembles such as Erdos-Renyi it's the distribution of the node degrees which induces a distribution over the BP messages. An instance of this can be found later in 2.3.

The approach presented here goes under the name *Replica Symmetric (RS) cavity method* and is explored in more detail in section 2.3. Connections with Replica theory, which lends the name to this flavor of the cavity method, are beyond the scope of this work. More details can be found in [1].

1.2.5 Max-sum, or the zero-temperature limit of BP

Temperature, or equivalently its inverse β, arises naturally in physics contexts. However, it can be a useful concept to introduce artificially in a computational problem to study a probability distribution around its regions of largest probability. In the following we will consider a Boltzmann distribution[2] $p(\mathbf{x}) = \frac{1}{Z} e^{-\beta H(\mathbf{x})}$.

In the $\beta \to \infty$ (zero-temperature) limit, the probability mass concentrates on the configurations that minimize the energy: $\mathbf{x}^* \in \arg\min_{\mathbf{x}} H(\mathbf{x})$. In this case the average

[2]Any given distribution q can be cast into this form by defining $H(\mathbf{x}) := -\frac{1}{\beta} \log q(\mathbf{x})$.

energy equals the minimum one

$$\langle H \rangle = \frac{\sum_{\mathbf{x}} H(\mathbf{x}) e^{-\beta H(\mathbf{x})}}{\sum_{\mathbf{x}} e^{-\beta H(\mathbf{x})}} \tag{1.30}$$

$$\stackrel{\beta \to \infty}{=} \frac{\sum_{\mathbf{x}^* \in \arg\min_{\mathbf{x}} H(\mathbf{x})} H(\mathbf{x}^*) e^{-\beta H(\mathbf{x}^*)}}{\sum_{\mathbf{x}^* \in \arg\min_{\mathbf{x}} H(\mathbf{x})} e^{-\beta H(\mathbf{x}^*)}} \tag{1.31}$$

$$= H(\mathbf{x}^*). \tag{1.32}$$

The entropic component of the free energy vanishes and energy and free energy coincide

$$F = -\frac{1}{\beta} \log \sum_{\mathbf{x}} e^{-\beta H(\mathbf{x})} \tag{1.33}$$

$$\stackrel{\beta \to \infty}{=} -\frac{1}{\beta} \log \max_{\mathbf{x}} e^{-\beta H(\mathbf{x})} \tag{1.34}$$

$$= \min_{\mathbf{x}} H(\mathbf{x}) \tag{1.35}$$

$$= H(\mathbf{x}^*). \tag{1.36}$$

There exists a variant of BP dealing with systems at zero temperature which works directly with logarithms of probabilities instead of probabilities themselves. It is called max-sum (or min-sum) and it is useful to tackle optimization problems defined as the zero-temperature limit of a distribution factorized on a factor graph

$$p(\mathbf{x}) = \frac{1}{Z} e^{-\beta \sum_a H_a(\mathbf{x}_a)}. \tag{1.37}$$

Consider BP messages as defined previously, and the following change of variables

$$\begin{aligned} m_{ia}(x_i) &\propto e^{\beta h_{ia}(x_i)} \\ m_{ai}(x_i) &\propto e^{\beta u_{ai}(x_i)} \end{aligned} \tag{1.38}$$

whose inverse is

$$\begin{aligned} h_{ia}(x_i) &= \frac{1}{\beta}\log m_{ia}(x_i) + const \\ u_{ai}(x_i) &= \frac{1}{\beta}\log m_{ai}(x_i) + const. \end{aligned} \tag{1.39}$$

Substituting this definition in the BP equations (1.4), then taking the limit $\beta \to \infty$ gives the max-sum equations

$$\begin{aligned} h_{i\to a}(x_i) &= \sum_{b\in\partial i\setminus a} u_{b\to i}(x_i) \\ u_{a\to i}(x_i) &= \max_{\mathbf{x}_{\partial a\setminus i}} \left[\psi_a(\mathbf{x}_a) + \sum_{j\in\partial a\setminus i} h_{j\to a}(x_j) \right]. \end{aligned} \tag{1.40}$$

Beliefs are obtained as

$$h_i(x_i) = \frac{1}{\beta}\log b_i(x_i) + const \tag{1.41}$$

$$= \sum_{a\in\partial i} u_{a\to i}(x_i). \tag{1.42}$$

If the ground state (configuration of minimum energy) is unique, then, modulo the approximation introduced by the cavity assumption, decision variables

$$x_i^* = \arg\max_{x_i} h_i(x_i) \tag{1.43}$$

solve the optimization problem $\mathbf{x}^* = \arg\min_{\mathbf{x}} H(\mathbf{x})$. In situations where the ground state is not unique but one is interested in retrieving any one minimizers, some strategies can be employed to bias max-sum towards a specific solution. These techniques, decimation and reinforcement, are explained later in chapter 2.

1.2.6 Limitations of the cavity assumption

As anticipated, the cavity assumption holds exactly only on trees, it is only approximately true otherwise. It was found, at first empirically, that for systems that are not acyclic, but where correlations decay fast enough along loops, calculations based on the

different flavors of the cavity method give almost exact results. A paramount example is given by models living on large, diluted random graphs, such as Erdos-Renyi or Random Regular graphs. These are graphs sampled from a uniform measure over all graphs with a fixed size and a certain degree distribution. In the infinite size limit these locally present a tree-like structure: the subgraph around any node up to finite distance contains no loop with high probability (going to 1 in the limit of size $N \to \infty$).

BP can even be asymptotically exact on such systems, however local tree structure is not a sufficient condition. A modern point of view classifies problems on random graphs according to the structure of correlation functions across the system (see [1, Chapt. 22] and references therein). In particular, given a joint distribution $p(x_1, x_2, \dots, x_N)$ and the corresponding marginals $\{p_{i_1,i_2,\dots,i_l}(x_{i_1}, x_{1_2}, \dots, x_{i_l})\}_{i_1,i_2,\dots,i_l \in \{1,2,\dots,N\}}$, two types of correlations are taken into account. The first is l-point correlations, with l finite with respect to N:

$$\chi^{(l)} = \frac{1}{N^{l-1}} \sum_{i_1,i_2,\dots,i_l \in \{1,2,\dots,N\}} \left\| p_{i_1,i_2,\dots,i_l}(x_{i_1}, x_{1_2}, \dots, x_{i_l}) - p_{i_1}(x_{i_1}) \cdots p_{i_l}(x_{i_l}) \right\|, \tag{1.44}$$

a measure of how close joint distributions are to being factorized[3]. This correlation measure can be related to the stability of the system under small perturbations [1]: whenever $\chi^{(l)}/N$ stays finite as $N \to \infty$, the system is strongly correlated and hardest for BP to correctly describe.

There can be situations where l-point correlations vanish but BP still has trouble converging and finding good solutions. This is due to another type of correlations called *point-to-set* correlations. Let us define $B_l(i)$ the set of vertices at distance less or equal than l from a given origin vertex i, $\overline{B_l(i)}$ its complement, and $\mathbf{x}_{B_l(i)}$ the corresponding set of variables. The point-to-set correlation function at vertex i and distance l is defined as

$$G_l(i) = \left\| p_{i,B_l(i)}(x_i, \mathbf{x}_{B_l(i)}) - p_i(x_i) p_{B_l(i)}(\mathbf{x}_{B_l(i)}) \right\|, \tag{1.45}$$

again as a measure of how "dependent" a given variable is from others far away. In simple terms, non-vanishing point-to-set correlations mean that variables living in far-apart points of the graph can still influence each other.

[3] The sum over all choices of vertices is needed to average over possible inhomogeneities in the system

Defining such two kinds of correlation functions allows for a classification of the possible scenarios. Depending on whether none, both or one of the two vanish at long distance on the graph, three regimes can be identified:

- **Replica Symmetric (RS)**: both types of correlations are "short-ranged", BP and the RS cavity method are asymptotically exact.
- **Dynamical 1-step Replica Symmetry Breaking (d1RSB)**: two-point correlations vanish but point-to-set correlations are long-range. In this regime there exists a stable BP fixed point well describing the distribution $p(\mathbf{x})$, however iterations starting from a random initialization have a vanishing probability of finding it. Nevertheless, and this is clearest in constraint satisfaction problems, samples that are typical of p are far apart in configuration space, such that going from one to the other requires to rearrange a large portion of the system [27]. This regime can be probed dynamically, hence the name, by initializing a local algorithm such as BP in a typical configuration taken from $p(\mathbf{x})$, and verifying that the dynamics gets stuck.
- **Replica Symmetry Breaking (RSB)**: BP and the RS cavity method are not enough to correctly describe the system. In some cases a so-called *static* 1RSB picture is appropriate, in which case BP and the RS cavity method need to be improved upon in order to describe the more complex behavior. The resulting methods take the names survey propagation (SP) and 1RSB cavity method, respectively. We will describe both directly for a specific model in chapter 2, because an example is useful in explaining the idea, and because implementation details depend substantially on the features of the specific model.

Other notable generalizations of BP which will not be used in this manuscript are the cluster variational method [28, 29], later encompassed in generalized BP [2], tree-reweighted BP [30], a hybrid message passing-Monte Carlo strategy [31], the integration of BP with tensor network techniques [32].

Part I

Equilibrium

Chapter 2

The closest vector problem

In this chapter, we present a comprehensive study of the Closest Vector Problem (CVP), a constrained optimization problem that is central to various computational fields. Despite the apparent simplicity of CVP's formulation, defined by linear constraints over the Galois Field $GF(2)$ coupled with a linear optimization function, the problem exhibits a rich and complex structure characterized by non-trivial phase transitions. We use the cavity formalism presented in the previous chapter to study the interplay between these transitions and the performance of message-passing algorithms that look for approximate solutions to the optimization problem.

We first focus on ensembles of systems where each variable participates in at most two constraints. This allows us to derive exact results and demonstrate the correctness of the max-sum algorithm. This result is particularly significant for understanding the behavior of CVP in simpler, yet highly relevant, instances. We then move on to explore more complex instances where variables are involved in more than two constraints. In these scenarios, several critical phase transitions arise. Notably, we identify two distinct clustering transitions: one affecting the solution space of the linear system of constraints, and another impacting the solution of the optimization problem. Between these transitions, the solution space of the linear system remains well-connected, but the energy landscape becomes *glassy*, with optimal solutions forming clustered structures. Surprisingly, message-passing algorithms perform efficiently in this intermediate regime, only experiencing a decline in performance once the solution set of the linear system itself begins to cluster.

2.1 Problem setting

We are going to study a problem whose mathematical formulation can be derived from (at least) four directions, each originating in a different field of science:

- The closest vector problem (CVP), a classic in discrete constrained optimization with interesting consequences for cryptography (discrete mathematics).
- The decoding phase of a certain strategy for lossy compression of binary data from a random source (telecommunications).
- Solving XORSAT formulas with an added bias on each variable to be true or false (computer science).
- Finding the ground state of a p-spin Ising model with ± 1 external fields and ferromagnetic interactions (statistical physics of disordered systems).

We will show how all can be mapped onto the last one. Once the problem is framed in a familiar statistical mechanical context, it can be tackled via the cavity method, in some of its declinations illustrated earlier.

The main motivation for studying this model is that it is an ideal framework to understand, via statistical mechanics computations, the mechanisms underlying the hardness in approximating optimal solutions. As we will see, it shows a wide range of non-trivial properties in the geometry of the solution landscape. At the same time, it conserves some analytic feasibility by means of linear algebra techniques, coming from the fact that its configurations are solutions of a linear system of equations.

2.1.1 The closest vector problem

The Closest Vector Problem (CVP) [33] is a constrained optimization problem in which the objective is to find the vector in a high-dimensional lattice that is closest to a given reference vector (in general, external to the lattice). In this work we will study lattices defined by linear subspaces of $\mathrm{GF}(q)^n$, the n-dimensional vector space over the Galois Field $\mathrm{GF}(q)$.

We will be mostly interested in the case $q = 2$, where the field is based on the set $\{0,1\}$, endowed with XOR ($\oplus$) as the addition operation

$$x \oplus y = \mathbb{1}[x \neq y] \tag{2.1}$$

and AND as multiplication

$$xy = \mathbb{1}[x = 1 \wedge y = 1] \tag{2.2}$$

where $\mathbb{1}$ is the indicator function. With a slight abuse of notation, we will sometimes confuse the field $\mathrm{GF}(2)$ with its base set $\{0,1\}$.

The constrained optimization problem can be formulated as follows. Consider a reference vector $\mathbf{y} \in \mathrm{GF}(2)^n$ and a linear subspace $\mathscr{C} \subset \mathrm{GF}(2)^n$ of dimension $k < n$, often referred to as the *lattice*. $\mathbf{y}$ may or may not belong to $\mathscr{C}$, although in the regimes we will be studying it most often does not. The goal is to find a vector $\hat{\mathbf{x}} \in \mathscr{C}$ that is the closest to $\mathbf{y}$

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x} \in \mathscr{C}} d_H(\mathbf{x}, \mathbf{y}) \tag{2.3}$$

where distances are measured by the Hamming distance

$$d_H(\mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^{n} x_i \oplus y_i. \tag{2.4}$$

The problem of finding the closest vector requires, in principle, to go through all the 2^k elements of the lattice, checking their distance from the reference. Indeed, the CVP has been proven to be computationally NP-hard [34] and hard to approximate, even allowing a potentially slow pre-processing step [35, 36]. CVP is a fundamental problem in theoretical computer science: some versions of it were among the first ones where an equivalence between worst case and average case complexity has been shown. Such an equivalence has been exploited to propose a robust cryptosystem [37] .

2.1.2 Lossy compression of a random binary source

The CVP offers a direct application in lossy compression of a symmetric binary source, a.k.a. source coding. In this context, the goal of compression is to reduce an

input $\mathbf{y} \in \mathrm{GF}(2)^n$ into a compressed form $\mathbf{c} \in \mathrm{GF}(2)^k$, where $k < n$. Decompression then transforms $\mathbf{c}$ back into an estimate $\mathbf{x} \in \mathrm{GF}(2)^n$. Distortion is measured by the Hamming distance $d_H(\mathbf{y}, \mathbf{x})$, which counts the number of differing components between the two vectors. A good compression method minimizes this distortion, and its performance is evaluated based on the average distortion over random binary inputs.

For example, a simple compression method involves truncating the input to the first k components during compression and randomly reconstructing the remaining $n-k$ components during decompression. Naturally, better performance can be achieved. Shannon established the optimal performance for a given input distribution by using a duality with the channel coding problem [38–40]. The minimum achievable average distortion for a binary symmetric source, with a given pair (n,k), is expressed as

$$D = H^{-1}(1-R) \tag{2.5}$$

where $R = \frac{k}{n}$ is the compression rate, and

$$H(p) = -p\log_2(p) - (1-p)\log_2(1-p) \tag{2.6}$$

is the binary entropy function, with H^{-1} as its inverse. Figure 2.1 shows the rate-distortion plane and the corresponding regimes. Throughout this chapter, algorithm quality will be assessed by plotting rate and distortion values on this plane. The closer to the bottom-left corner, the better the performance.

Interestingly, random codes form an asymptotically optimal but computationally inefficient scheme. These codes are constructed by selecting 2^k random vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{2^k}$ from $GF(2)^n$, known as *codewords*. The compression process involves finding the codeword $\mathbf{v}_i$ closest to $\mathbf{y}$, and the binary representation $\mathbf{c}$ of i serves as the compressed vector. As n and k approach infinity, with a fixed rate $R = \frac{k}{n}$, the average distortion $\langle d_H(\mathbf{v}_i, \mathbf{y}) \rangle$ converges to the optimal line [38]. A computationally efficient alternative can be constructed by replacing random vectors with solutions of random linear systems in discrete space, as done in [41], which is the approach we adopt here.

The lossy compression problem for a random uniform binary source is formally equivalent to that of reliable transmission (channel coding) in a binary symmetric

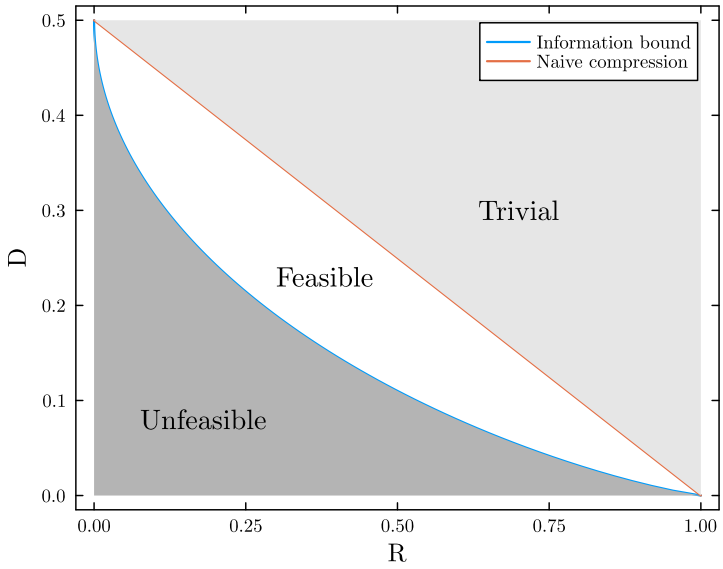


Fig. 2.1 The rate-distortion plane and its regimes. The blue line represents Shannon's bound, while the orange line depicts the trivial strategy of copying the first k bits and randomly reconstructing the rest.

channel, as described in [26]. Although channel coding has been studied using the cavity method [42, 43], the two problems differ in their source vector ensembles. In channel coding, the source vector is a perturbed codeword, whereas in compression, it is a random uniform vector. Thus, compression corresponds to a channel coding problem where decoding errors occur with non-vanishing probability, making it fundamentally more challenging. Moreover, the sets of "good" codes also differ: in channel coding, closely spaced codewords degrade performance because of non-vanishing error rates, while in compression, this proximity is not harmful and may even offer computational advantages. A more detailed comparison with channel coding is given in appendix A.1.

2.1.3 XORSAT formulas

Random codes, introduced in the previous section, correspond to the solution set of XORSAT formulas derived from a random ensemble. XORSAT, short for XOR-satisfiability, is a type of constraint satisfaction problem in Boolean logic, where clauses consist of the XOR operation applied to Boolean variables. This class of problems is an important subset of SAT (Satisfiability) problems, where the objective

is to determine whether there exists an assignment of Boolean variables, true or false, that satisfies the given set of constraints or clauses. XORSAT belongs to the broader class of constraint satisfaction problems (CSPs).

Let H be an $m \times n$ matrix with Boolean entries $H_{ia} \in \{0,1\}$, where $i \in \{1,\dots,n\}$ and $a \in \{1,\dots,m\}$. The solution set of the corresponding homogeneous XORSAT problem is defined as

$$\mathscr{C} = \{\mathbf{x} \in \{0,1\}^n : H\mathbf{x} = \mathbf{0}\} \tag{2.7}$$

where the equality is intended modulo 2. Each row of the linear system

$$\sum_i H_{ai} x_i = 0 \tag{2.8}$$

can be viewed as a *clause*, or *parity-check* in communication theory. The matrix H is typically referred to as the parity-check matrix, and the set of clauses forms a XORSAT *formula*. The system is homogeneous, meaning the right-hand side is the null vector $\mathbf{b} = \mathbf{0}$. This homogeneity does not result in a loss of generality, as any instance with non-zero $\mathbf{b}$ can be transformed into a homogeneous one via gauge transformation [12].

The random XORSAT ensemble (see [12, 44]) is defined by selecting $\mathbf{b}$ uniformly at random from $\{0,1\}^m$ and choosing H from a random matrix ensemble. In this work, we focus on the ensemble where H is sampled uniformly from matrices that have a prescribed distribution of non-zero entries per row and per column. The topology of a XORSAT instance can be represented by a bipartite graph $\mathscr{G} = ((V,F),E)$. The variable nodes V correspond to the binary variables $x_1,\dots,x_n$, while the factor nodes F represent the m constraints encoded by the rows of H. An edge $(i,a) \in E$ is drawn if the variable x_i appears in the a-th constraint, that is, if $H_{ia} = 1$.

A notable characteristic of random CSPs is the emergence of phase transitions as the number of variables n and constraints m grow infinitely large, while maintaining a fixed ratio $\alpha = m/n$, which represents the density of constraints per variable. For instance, the satisfiability threshold α_{sat} separates a satisfiable regime, where $\alpha < \alpha_{\text{sat}}$ and random XORSAT instances typically admit solutions, from an unsatisfiable phase, where $\alpha > \alpha_{\text{sat}}$ and solutions rarely exist. Another transition, known as the clustering transition, occurs in the satisfiable phase at $\alpha_d < \alpha_{\text{sat}}$. Below α_d, the solution set of typical instances is relatively well-connected, and any solution can

be reached from another through a path of nearby solutions. Above α_d, the solution set fragments into an exponential number of distinct clusters of solutions, which are internally well-connected but isolated from each other.

2.1.4 Mapping onto a p-spin Ising model

All three problems mentioned above can be mapped onto that of studying the zero-temperature regime of a discrete p-spin Ising model. Besides the convenience of working in a setting familiar to physicists, this point of view allows to study the same system at finite temperature, which corresponds to solving a "softer" version of the optimization problem. This can prove useful in addressing algorithmic issues.

An instance of the CVP, where the lattice is taken to be the set of solution of a random XORSAT instance as defined above, reads:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}:H\mathbf{x}=\mathbf{0}}{\arg\min}\, d_H(\mathbf{x},\mathbf{y}) \tag{2.9}$$

where $H \in \{0,1\}^{m,n}$ and $y \in \{0,1\}^n$.

By means of a simple change of variables,

$$\begin{aligned} \sigma_i &= (-1)^{x_i} \\ s_i &= (-1)^{y_i} \end{aligned} \tag{2.10}$$

the problem can be re-written as a statistical physics model. We define the probability law:

$$\mu(\underline{\sigma}) = \frac{1}{Z(\beta)} \left(\prod_{a=1}^{m} \mathbb{1}\left[\prod_{i\in\partial a} \sigma_i = 1 \right] \right) e^{\beta \sum_{i=1}^{n} s_i \sigma_i} \tag{2.11}$$

with β a real parameter. The problem (2.9) is then equivalent to finding the configuration σ maximizing the probability law $\mu(\sigma)$, or equivalently minimizing the energy function

$$E(\sigma) = -\sum_{i=1}^{n} \sigma_i s_i \tag{2.12}$$

under the set of constraints $\{\prod_{i\in\partial a}\sigma_i = 1\}_{a=1}^m$. Note that $E(\sigma)$ can be related to the distortion defined earlier in (2.4) as follows: $E(\sigma) = 2d_H(\mathbf{x},\mathbf{y}) - 1$.

It will be convenient to define a softened version of the probability law $\mu(\sigma)$, by replacing the hard constraints on the factors $\mathbb{1}\left[\prod_{i\in\partial a}\sigma_i = 1\right]$ by a soft constraint $e^{\beta J(\prod_{i\in\partial a}\sigma_i - 1)}$, with J a real parameter:

$$\mu_J(\sigma) = \frac{1}{Z(\beta,J)} e^{-\beta E_J(\sigma)} \tag{2.13}$$

where:

$$E_J(\sigma) = -J\sum_{a=1}^{m}\left(\prod_{i\in\partial a}\sigma_i - 1\right) - \sum_{i=1}^{n}\sigma_i s_i \ . \tag{2.14}$$

The first term is bringing an energetic cost $2J$ to each unsatisfied clause, while the second term is the original energy function, which favors configurations close to the source. In statistical physics, this model is known as the p-spin model in presence of heterogeneous external fields $s_1,\dots,s_n$. Sending $J\to\infty$ allows to recover the probability law $\mu(\sigma)$ defined in (2.11).

2.1.5 Random ensemble of instances

We will be interested in the characterization of the 'typical' properties of this constrained optimization problem, where typical is defined with respect to a random ensemble of instances, a property being considered typical if it occurs with a probability going to one in the thermodynamic limit. In particular, we will consider random external fields $s_1,\dots,s_n$ in which each external field s_i is i.i.d. uniformly in $\{-1,1\}$.

For what concerns the randomness in the choice of the parity-check matrix, we will borrow from the literature on communications over noisy channels and work with irregular code ensembles [45, 46]. It is convenient to switch to the equivalent representation of the set of constraints in terms of a factor graph $\mathscr{G} = ((V\cup F),E)$. We will consider random graph ensembles with fixed degree profiles, denoted $\mathbb{G}_n(\Lambda,P)$. Let $\Lambda = \{\lambda_1,\dots,\lambda_{d_{\max}}\}$ be the degree profile of the variable nodes, with $d_{\max}$ the maximal degree, and λ_i the fraction of variable nodes of degree i. Respectively, let $P = \{p_1,\dots,p_{k_{\max}}\}$ be the degree profile of the factor nodes, with

$k_{\max}$ the maximal degree, and p_i the fraction of factor nodes of degree i. The degree profiles are normalized: $\sum_{i=1}^{d_{\max}} \lambda_i = 1$ and $\sum_{i=1}^{k_{\max}} p_i = 1$, and they satisfy the following relation

$$m \sum_{i=1}^{k_{\max}} i p_i = n \sum_{i=1}^{d_{\max}} i \lambda_i = |E| \ .$$

We will be interested in the thermodynamic limit $n, m \to \infty$, with a fixed ratio $\alpha = m/n$, and fixed fractions λ_i, p_i's independent of n. The ratio α is called the density of constraints per variable and is related to the degree profiles as follows: $\alpha = \frac{\sum_{i=1}^{d_{\max}} i\lambda_i}{\sum_{i=1}^{k_{\max}} i p_i}$. In the thermodynamic limit, random graphs extracted from $\mathbb{G}_n(\Lambda, P)$ are locally tree-like: the neighborhood of an uniformly chosen vertex within a finite distance is acyclic, with probability going to 1. Note that in the formalism of lossy compression, the compression rate $R = \frac{n-m}{m}$ can be expressed in terms of the degree profiles:

$$R = 1 - \alpha = 1 - \frac{\sum_{i=1}^{d_{\max}} i\lambda_i}{\sum_{i=1}^{k_{\max}} i p_i}. \tag{2.15}$$

As mentioned earlier, CVP is a computationally hard problem. We therefore now turn to employing the cavity method as a heuristic to provide approximate results, as well as to study theoretical aspects of the problem.

2.2 Belief Propagation

In this section we apply the belief propagation (BP) and max-sum (MS) algorithms presented in section 1.2 for the approximation of the distribution under consideration. When possible, we will work in the slightly more generic version (2.13) with soft couplings. The hard constraint case $J \to \infty$ follows directly. We will see how the specific structure of the problem allows for simplifications: namely, BP messages can be encoded in a single real number, MS messages in an integer.

2.2.1 Belief propagation equations

The BP equations (1.4) specialized for the model under consideration read

$$\begin{aligned} m_{i\to a}(\sigma_i) &= \frac{1}{z_{i\to a}} e^{\beta\sigma_i s_i} \prod_{b\in\partial i\setminus a} m_{b\to i}(\sigma_i) \\ m_{a\to i}(\sigma_i) &= \frac{1}{z_{a\to i}} \sum_{\sigma_{\partial a\setminus i}} e^{\beta J\prod_{j\in\partial a}\sigma_j} \prod_{j\in\partial a\setminus i} m_{j\to a}(\sigma_j) \end{aligned} \tag{2.16}$$

where $z_{i\to a}, z_{a\to i}$ are normalization factors. The most attentive reader will have noticed that these differ from the general form we have given in (1.4) because of the external field term $e^{\beta\sigma_i s_i}$ in the first equation. This is because factors that depend on a single variable can conveniently be incorporated in the variable-to-factor equation. It is of course possible, and completely equivalent, to treat them as regular factors.

Variable beliefs are given by

$$b_i(\sigma_i) = \frac{1}{z_i} e^{\beta\sigma_i s_i} \prod_{b\in\partial i} m_{b\to i}(\sigma_i) \tag{2.17}$$

and factor beliefs by

$$b_a(\sigma_a) = \frac{1}{z_a} \sum_{\sigma_{\partial a}} e^{\beta J\prod_{j\in\partial a}\sigma_j} \prod_{j\in\partial a} m_{j\to a}(\sigma_j). \tag{2.18}$$

Since messages can take only two values, constrained to sum to 1 for normalization, one can parametrize them using only one real number. Consider the following parametrization

$$\begin{aligned} m_{a\to i}(\sigma_i) &= \frac{e^{\beta u_{ai}(\sigma_i)}}{e^{\beta u_{ai}(+1)} + e^{\beta u_{ai}(-1)}} \\ m_{i\to a}(\sigma_i) &\propto \frac{e^{\beta h_{ia}(\sigma_i)}}{e^{\beta h_{ia}(+1)} + e^{\beta h_{ia}(-1)}}. \end{aligned} \tag{2.19}$$

The distributions above can be fully specified by a single parameter defined (with a slight abuse of notation) as

$$u_{ai}(\sigma_i) = u_{ai}\sigma_i, \quad h_{ia}(\sigma_i) = h_{ia}\sigma_i \tag{2.20}$$

in terms of the real numbers u_{ai}, h_{ia}.

We can re-write BP updates (2.16) in terms of the new messages

$$\begin{aligned} e^{\beta h_{ia}\sigma_i} &\propto e^{\beta\sigma_i s_i}\frac{e^{\beta\sigma_i\sum_{b\in\partial i\setminus a}u_{bi}}}{\prod_{b\in\partial i\setminus a}2\cosh(\beta u_{bi})} \\ e^{\beta u_{ai}\sigma_i} &\propto \sum_{\underline{\sigma}_{a\setminus i}} e^{\beta J\sigma_i\prod_{j\in\partial a\setminus i}\sigma_j}\prod_{j\in\partial a\setminus i}\frac{e^{\beta h_{ja}\sigma_j}}{2\cosh\left(\beta h_{ja}\right)}. \end{aligned} \tag{2.21}$$

The second equation becomes, using the identities $e^x=\cosh(x)+\sinh(x)$, $\cosh(-x)=\cosh(x)$, $\sinh(-x)=-\sinh(x)$

$$\begin{aligned} e^{\beta u_{ai}\sigma_i} &\propto \sum_{\underline{\sigma}_{a\setminus i}}\left[\cosh\left(\beta J\sigma_i\prod_{j\in\partial a\setminus i}\sigma_j\right)+\sinh\left(\beta J\sigma_i\prod_{j\in\partial a\setminus i}\sigma_j\right)\right]\prod_{j\in\partial a\setminus i}\frac{e^{\beta h_{ja}\sigma_j}}{2\cosh\left(\beta h_{ja}\right)} \\ &\propto \sum_{\underline{\sigma}_{a\setminus i}}\left[\cosh(\beta J)+\sigma_i\prod_{j\in\partial a\setminus i}\sigma_j\sinh(\beta J)\right]\prod_{j\in\partial a\setminus i}\frac{e^{h_{ja}\sigma_j}}{2\cosh\left(\beta h_{ja}\right)} \\ &\propto \sum_{\underline{\sigma}_{a\setminus i}}\left[1+\sigma_i\prod_{j\in\partial a\setminus i}\sigma_j\tanh(\beta J)\right]\prod_{j\in\partial a\setminus i}\frac{e^{\beta h_{ja}\sigma_j}}{2\cosh\left(\beta h_{ja}\right)} \\ &\propto 1+\sigma_i\tanh(\beta J)\prod_{j\in\partial a\setminus i}\sum_{\sigma_j}\frac{\sigma_j e^{\beta h_{ja}\sigma_j}}{2\cosh\left(\beta h_{ja}\right)} \\ &\propto 1+\sigma_i\tanh(\beta J)\prod_{j\in\partial a\setminus i}\tanh\left(\beta h_{ja}\right) \\ &\propto e^{\tanh^{-1}\left[\tanh(\beta J)\prod_{j\in\partial a\setminus i}\tanh(\beta h_j)\right]\sigma_i} \end{aligned} \tag{2.22}$$

from which we read off

$$u_{ai}=\frac{1}{\beta}\tanh^{-1}\left[\tanh(\beta J)\prod_{i=1}^{k}\tanh(h_i)\right]. \tag{2.23}$$

Finally, the two BP updates are

$$\begin{aligned} h_{ia} &= \sum_{b\in\partial i\setminus a}u_{bi}+s_i \\ u_{ai} &= \frac{1}{\beta}\tanh^{-1}\left[\tanh(\beta J)\prod_{j\in\partial a\setminus i}\tanh(\beta h_{ja})\right]. \end{aligned} \tag{2.24}$$

The normalization of variable beliefs are given by

$$\begin{aligned} z_i &= \sum_{\sigma_i} e^{\beta \sigma_i s_i} \prod_{b \in \partial i} \frac{e^{\beta u_{bi} \sigma_i}}{2\cosh(\beta u_{bi})} \\ &= \frac{2\cosh\left[\beta\left(s_i + \sum_{b\in\partial i} u_{bi}\right)\right]}{\prod_{b\in\partial i} 2\cosh(\beta u_{bi})}. \end{aligned} \tag{2.25}$$

The normalization of factor beliefs can be obtained via a calculation analogous to (2.22)

$$\begin{aligned} z_a &= \sum_{\sigma_{\partial a}} e^{\beta J \prod_{j\in\partial a}\sigma_j} \prod_{j\in\partial a} \frac{e^{\beta h_{ja}\sigma_j}}{2\cosh(\beta h_{ja})} \\ &= \cosh(\beta J)\left[1 + \tanh(\beta J) \prod_{j\in\partial a} \tanh\left(\beta h_{ja}\right)\right]. \end{aligned} \tag{2.26}$$

In the $J \to \infty$ limit, one has $\tanh(\beta J) = 1$. The factor $\cosh(\beta J)$ in front can be disregarded as it is a constant with no physical meaning [1].

A reasoning analogous to the one done for the first update equation gives an expression for the beliefs as

$$b_i(\sigma_i) \propto e^{\beta h_i \sigma_i} \tag{2.27}$$

with

$$h_i = \sum_{b\in\partial i} u_{bi} + s_i. \tag{2.28}$$

Let us mention that this version of the BP updates, while involving fewer parameters, can be numerically unstable with respect to one implementing (2.16) directly. The reason is that tanh is numerically indistinguishable from ± 1 for large (absolute) values of its arguments. For instance, in Julia v10.5,

```
julia> atanh(tanh(20))
Inf
```

[1] Notice that one can cleanly get rid of it by restoring the -1 in the energy term $J\prod_{i\in\partial a}\sigma_i - 1$, resulting in a $e^{-\beta J}$ factor multiplying z_a. This gives an overall contribution of $\cosh(\beta J)e^{-\beta J} = \frac{1+e^{-2\beta J}}{2}$ which is well-behaved for any J, β positive.

2.2.2 Max-sum equations

As illustrated in section 1.2.5, the max-sum algorithm can be used to describe the minimum energy states by taking the $\beta \to \infty$ limit of the BP equations (2.21). The first equation is

$$h_{ia} = \sum_{b \in \partial i \setminus a} u_{bi} + s_i, \tag{2.29}$$

just as in the finite-temperature case. Similarly for the beliefs

$$h_i = \sum_{b \in \partial i} u_{bi} + s_i, \tag{2.30}$$

The factor-to-variable equation is more involved:

$$\begin{aligned} u_{ai}\sigma_i &= \frac{1}{\beta} \log \left\{ e^{\beta \max\limits_{\underline{\sigma}_{a\setminus i}} \left[J\sigma_i \prod_{j\in\partial a\setminus i} \sigma_j + \sum_{j\in\partial a\setminus i} h_{ja}\sigma_j \right]} \right\} \\ &\xrightarrow{\beta\to\infty} \max_{\underline{\sigma}_{a\setminus i}} \left[J\sigma_i \prod_{j\in\partial a\setminus i} \sigma_j + \sum_{j\in\partial a\setminus i} h_{ja}\sigma_j \right]. \end{aligned} \tag{2.31}$$

After some algebra (see appendix A.3), one arrives to the final result

$$u_{ai} = \operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja} \right) \left(\min\left\{ \min_{j\in\partial a\setminus i} \left|h_{ja}\right|, J \right\} \right). \tag{2.32}$$

In summary, the max-sum equations read

$$\begin{aligned} u_{ai} &= \operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja} \right) \left(\min\left\{ \min_{j\in\partial a\setminus i} \left|h_{ja}\right|, J \right\} \right) \\ h_{ia} &= \sum_{b\in\partial i\setminus a} u_{bi} + s_i \\ h_i &= \sum_{b\in\partial i} u_{bi} + s_i. \end{aligned} \tag{2.33}$$

In the $J \to \infty$ limit one simply discards J.

In the case of a tree graph where max-sum is exact, the belief h_i corresponds to the difference in energy $\Delta E_i = E_i^{\min}(+) - E_i^{\min}(-)$, where $E_i^{\min}(\sigma)$ is the ground

state energy when σ_i is fixed to the value σ. Since the energy function $E(\sigma)$ takes only integer values, one can deduce that the max-sum messages satisfying equations (2.33) and max-sum beliefs (corresponding to differences in energy) also take integer values. This observation will be useful later when applying the replica symmetric cavity method (section 2.3).

Replacing the hard constraints by soft constraints is equivalent to introducing a cut-off on the values of the factor-to-variable $u_{a\to i} \in [-J, J]$. The variable-to-factor messages then take values $h_{i\to a} \in [-1 - J(d_{\max} - 1), 1 + J(d_{\max} + 1)]$ where d_{max} is the maximum node degree (number of neighbors) in the graph.

Decision variables

Once the beliefs have been computed, the max-sum estimate for the minimum energy configuration is given by

$$\sigma_i^* = \arg\max_{\sigma_i} b_i(\sigma_i) = \text{sign}(h_i). \tag{2.34}$$

It may very well happen that, on a well converged max-sum fixed-point, variables are undecided, i.e. $h_i = 0$ for some i. This corresponds to having multiple configurations minimizing the energy, across which variable σ_i takes different values. Since one is interested in finding a single minimizer, several strategies can be employed to break this symmetry and have the algorithm collapse on a single solution. The details are given in sections 2.2.4 and 2.2.5.

2.2.3 Energy and free energy

On a fixed point, beliefs can be used to compute the average of the energy (2.14)

$$\langle E \rangle = -J \sum_{a=1}^{m} \left(\langle \prod_{i \in \partial a} \sigma_i \rangle_{b_a} - 1 \right) - \sum_{i=1}^{n} \langle \sigma_i \rangle_{b_i} s_i. \tag{2.35}$$

We showed in the introductory chapter how, at zero temperature, average energy, minimum energy and free energy all coincide. In this regime, once messages and beliefs have been computed with max-sum, it is most convenient to compute the

energy through the Bethe free energy (1.26). Written in terms of messages in the u,h parametrization, it reads

$$F(\mathbf{u},\mathbf{h}) = \sum_{a=1}^{m} F_a(\{h_{i\to a}\}_{i\in\partial a}) + \sum_{i=1}^{n} F_i(\{u_{a\to i}\}_{a\in\partial i}) + \sum_{(i,a)\in E} F_{ia}(h_{i\to a}, u_{a\to i}) \quad (2.36)$$

with

$$\begin{aligned} F_a &= -\frac{1}{\beta}\log z_a \\ &= -\frac{1}{\beta}\log\left(\sum_{\sigma_a:\prod_{i\in\partial a}=1}\prod_{i\in\partial a}\frac{e^{\beta\sigma_i h_{ia}}}{2\cosh(\beta h_{ia})}\right) \\ &= -\frac{1}{\beta}\log\left(\sum_{\sigma_a:\prod_{i\in\partial a}=1}\prod_{i\in\partial a}e^{\beta\sigma_i h_{ia}}\right) + \frac{1}{\beta}\log\prod_{i\in\partial a}2\cosh(\beta h_{ia}) \\ &\xrightarrow{\beta\to\infty} -\sum_{i\in\partial a}|h_{ia}| + \Theta\left(-\prod_{i\in\partial a}h_{ia}\right)2\min_{i\in\partial a}|h_{ia}| + \sum_{i\in\partial a}|h_{ia}| \\ &= \Theta\left(-\prod_{i\in\partial a}h_{ia}\right)2\min_{i\in\partial a}|h_{ia}| \end{aligned} \quad (2.37)$$

and

$$\begin{aligned} F_i &= -\frac{1}{\beta}\log z_i \\ &= -\frac{1}{\beta}\log\left(\sum_{\sigma_i}\frac{e^{\beta\sigma_i(s_i+\sum_{a\in\partial i}u_{ai})}}{\prod_{a\in\partial i}2\cosh(\beta u_{ai})}\right) \\ &= -\frac{1}{\beta}\log\left(\sum_{\sigma_i}e^{\beta\sigma_i(s_i+\sum_{a\in\partial i}u_{ai})}\right) + \frac{1}{\beta}\sum_{a\in\partial i}\log(2\cosh(\beta u_{ai})) \\ &\xrightarrow{\beta\to\infty} -\left|s_i+\sum_{a\in\partial i}u_{ai}\right| + \sum_{a\in\partial i}|u_{ai}| \end{aligned} \quad (2.38)$$

and

$$\begin{aligned} F_{ia} &= -\frac{1}{\beta}\log\left(\sum_{\sigma_i}\frac{e^{\beta\sigma_i(h_{ia}+u_{ai})}}{\cosh(\beta h_{ia})\cosh(\beta u_{ai})}\right) \\ &\xrightarrow{\beta\to\infty} -|h_{ia}+u_{ai}| + |h_{ia}| + |u_{ai}| \end{aligned} \quad (2.39)$$

In summary,

$$\begin{aligned} F_a(\{h_{i\to a}\}_{i\in\partial a}) =& 2\min_{i\in\partial a}(|h_{i\to a}|)\Theta\left(\prod_{i\in\partial a} h_{i\to a}\right) \\ F_i(\{u_{a\to i}\}_{a\in\partial i}) =& -\left|s_i+\sum_{a\in\partial i} u_{a\to i}\right| + \sum_{a\in\partial i}|u_{a\to i}| \\ F_{ia}(h_{i\to a},u_{a\to i}) =& -|u_{a\to i}+h_{i\to a}| \\ &+|u_{a\to i}|+|h_{i\to a}|. \end{aligned} \tag{2.40}$$

With soft constraints, i.e. J finite, the factor contribution $E_a(\{h_{i\to a}\}_{i\in\partial a})$ to the Bethe minimal energy (2.36) is replaced by

$$F_a(\{h_{i\to a}\}_{i\in\partial a}) = 2\min\left(J, \min_{i\in\partial a}|h_{i\to a}|\right)\Theta\left(\prod_{i\in\partial a} h_{i\to a}\right). \tag{2.41}$$

Adding the terms together, something cancels out and we spot variable beliefs h_i

$$\begin{aligned} F =& \sum_a \left[\Theta\left(-\prod_{i\in\partial a} h_{ia}\right) 2\min_{i\in\partial a}|h_{ia}|\right] + \sum_i \left[-\underbrace{\left|s_i+\sum_{a\in\partial i} u_{ai}\right|}_{h_i} + \cancel{\sum_{a\in\partial i}|u_{ai}|}\right] + \\ &-\sum_i\sum_{a\in\partial i}\left[-|\underbrace{h_{ia}+u_{ai}}_{h_i}| + |h_{ia}| + \cancel{|u_{ai}|}\right] \\ =& \sum_a \left[\Theta\left(-\prod_{i\in\partial a} h_{ia}\right) 2\min_{i\in\partial a}|h_{ia}|\right] - \sum_i\sum_{a\in\partial i}|h_{ia}| + \sum_i(|\partial i|-1)\,|h_i|\,. \end{aligned} \tag{2.42}$$

Grouping together the second and third terms we see that the free energy is composed of a factor and a variable term

$$F = \sum_{a\in F}\left[\Theta\left(-\prod_{i\in\partial a} h_{ia}\right) 2\min_{i\in\partial a}|h_{ia}|\right] + \sum_{i\in V}\left[(|\partial i|-1)\,|h_i| - \sum_{a\in\partial i}|h_{ia}|\right]. \tag{2.43}$$

This expression will turn out useful later in the context of the replica symmetric cavity method (section 2.3).

2.2.4 Decimation

The output of the BP algorithm is just an estimate of single-site marginals, and to find a solution to the optimization problem, one needs to convert these marginals into a specific spin configuration. However, note that picking σ^* where $\sigma_i^* = \arg\max b_i(\sigma_i)$ does not lead to a good result in general, as it disregards existing correlations between variables (e.g. in case of problems with hard constraints this strategy can lead to inconsistencies, since σ^* in general does not satisfy the constraints). To overcome this issue one typically resorts to decimation: sequentially, one variable is fixed according to its belief by means of a strong external field, then BP (or MS) is run again until convergence, and so on.

Here we employ a similar strategy which however takes advantage of the set of XORSAT constraints being a linear system of equations. We first use Gaussian elimination to build a basis for this linear system, thereby identifying a subset of independent variables. The decimation procedure is then applied only to these independent variables. Once all independent variables are fixed, the remaining variables are determined by the linear constraints, thus ensuring that we obtain a solution to the linear system. At each time step, the algorithm solves iteratively the BP equations (2.16) until convergence and computes the marginal probabilities of each variable. Then, the algorithm picks the most biased variable, i.e. $i^* = \arg\max_i(\mu_i(+) - \mu_i(-))$ among the independent variables that are not yet decimated, samples $\sigma_i \in \{-1, 1\}$ according to its marginal μ_i and switches on a strong external field in the corresponding direction, in such a way that σ_i is now fixed in the direction of its belief. In the max-sum limit of $\beta \to \infty$, the system should be fully concentrated on the configurations of minimal energy. If the minimum is not unique, then decimation is still needed to break the symmetry between equivalent ground states. Alternatively, one can add a small symmetry-breaking random external field so that the ground state becomes unique. This is the strategy we adopted for max-sum to obtain the results later in this chapter. Survey propagation, presented later in 2.4.1, is also complemented with a decimation procedure.

2.2.5 Reinforcement

An alternative to decimation is reinforcement, which consists in updating the external field on a variable according to its belief, thus guiding the system to full polarization.

Reinforcement is also sometimes called soft decimation, as it sets at each iteration a soft external field of all variables instead of a strong field on only one variable. The reinforcement procedure can also be employed to help convergence of MS equations, the small external fields accumulate during time (before convergence) to drive the system to a model with strong external fields for which convergence is easier to achieve. At each iteration t of the max-sum algorithm (2.33), the external field on each variable is updated according to its belief

$$s_i^{(t+1)} = s_i^{(t)} + \gamma(t) h_i^{(t)} \tag{2.44}$$

with $h_i^{(t)}$ the max-sum belief computed at time t, and $\gamma(t)$ a used-defined reinforcement schedule. In practice, to obtain the results in this chapter, we used the same value at each iteration $\gamma(t) \equiv \gamma \sim N_{iter}^{-1}$ where N_{iter} is the number of iterations.

2.3 The replica symmetric cavity method

We now look at the problem of estimating the optimal distortion that can be achieved for an average instance of CVP sampled from a given random ensemble, in the limit $n \to \infty$. As introduced in section 1.2.4, the idea of the cavity method is to use the infrastructure of BP (or MS) messages to construct updates for the probability distributions induced on messages by some disorder in the model. This allows to work directly in the thermodynamic limit while dealing with disorder. In our case, the disorder comes from two factors. The first is the randomness of the source vector $\mathbf{s}$, which we recall is made of independent components, each taken from a $Bernoulli(0.5)$. The second is the randomness in the graph, whose variable and factor nodes have different degrees according to some given probability distribution.

We look at $p(u)$ and $q(h)$, which are the distributions for a generic max-sum message message from factor to variable u_{ai} and from variable to factor h_{ia}, respectively. One could do the same with BP messages, however we showed how MS messages for this problem only take integer values. This is a massive advantage, as their distribution can be specified by a limited set of real parameters.

2.3.1 Fixed degrees

We will begin by considering an edge connecting a factor of degree $k+1$ with a variable of degree $d+1$. Later, we will perform an average over the degree profile according to the desired graph ensemble.

Assuming messages are distributed according to the distributions $p(u), q(h)$, the laws for the outgoing messages, at fixed degrees, are given by

$$\begin{aligned} p_k(u) &= \sum_{h_1,\dots,h_k} \delta(u, \hat{f}^{\mathrm{MS}}(h_1,\dots,h_k)) \prod_{i=1}^{k} q(h_i) \\ q_d(h) &= \sum_s p_s(s) \sum_{u_1,\dots,u_d} \delta(h, f^{\mathrm{MS}}(u_1,\dots,u_d;s)) \prod_{a=1}^{d} p(u_a) \end{aligned} \tag{2.45}$$

where $f^{\mathrm{ms}}(u_1,\dots,u_d;s)$ and $\hat{f}^{\mathrm{ms}}(h_1,\dots,h_k)$ are shorthand notation for the r.h.s. of the MS equations (2.33) and $\delta(x,y) = \mathbb{1}[x=y]$ is the Kroenecker delta function. Here s denotes one component of the source vector, distributed as p_s. In addition to these two, we can also compute (almost for free) the distribution of a belief, which an expression similar to $q(h)$ but with $d+1$ incoming u's.

For practical computational reasons, we will assume $|h| \leq J$ although in principle there is no bound. This way we can build $q(h)$ on a computer as an array with indices fixed at `-J:J`. Notice that the u's instead are bounded by $\pm J$ by construction (see (2.33)).

Factor to variable

Substituting the expression for the MS update in the first equation in (2.59) gives

$$p_k(u) = \sum_{h_1,\dots,h_k} \delta\left(u, \min\left\{J, \min_{j=1,\dots,k} |h_j|\right\} \operatorname{sign}\left(\prod_{j=1}^{k} h_j\right)\right) \prod_{j=1}^{k} q(h_j). \tag{2.46}$$

First it is useful to distinguish the case $u=0$. In this case, we call n the number of incoming h's which are equal to zero (there must be at least one for the delta to be

satisfied) and separate the sum over the zero h's and the non-zero ones:

$$\begin{aligned}
p_k(0) &= \sum_{h_1,\dots,h_k} \delta\left(0, \min\left\{J, \min_{j=1,\dots,k} |h_j|\right\}\right) \prod_{j=1}^{k} q\left(h_j\right) \\
&= \sum_{n=1}^{k} \binom{k}{n} q(0)^n \prod_{j=n+1}^{k} \sum_{h_j \neq 0} q(h_j) \\
&= \sum_{n=1}^{k} \binom{k}{n} q(0)^n \left[1-q(0)\right]^{k-n} \\
&= \underbrace{\sum_{n=0}^{k} \binom{k}{n} q(0)^n \left[1-q(0)\right]^{k-n}}_{[q(0)+1-q(0)]^k} - \left[1-q(0)\right]^k \\
&= 1 - \left[1-q(0)\right]^k .
\end{aligned} \tag{2.47}$$

As for the case $u \neq 0$, we can limit ourselves to studying what happens for $u > 0$ as $p(u) = p(-u)$ by symmetry, since the distribution of the source bit is symmetric. It is useful to separate the sum over signs and absolute values of the h's

$$\begin{aligned}
p_k(u) &= \sum_{|h_1|,\dots,|h_k|} \sum_{\operatorname{sign}(h_1),\dots,\operatorname{sign}(h_k)} \delta\left(u, \min\left\{J, \min_{j=1,\dots,k} |h_j|\right\} \prod_{j=1}^{k} \operatorname{sign}\left(h_j\right)\right) \prod_{j=1}^{k} q\left(|h_j| \operatorname{sign}(h_j)\right) \\
&= \sum_{|h_1|,\dots,|h_k|} 2^{k-1} \delta\left(u, \min\left\{J, \min_{j=1,\dots,k} |h_j|\right\}\right) \prod_{j=1}^{k} q\left(h_j\right) \\
&= \frac{1}{2}\left\{ \sum_{|h_1|,\dots,|h_k| \geq u} \prod_{j=1}^{k} 2q\left(h_j\right) - \sum_{|h_1|,\dots,|h_k| > u} \prod_{j=1}^{k} 2q\left(h_j\right) \right\} \\
&= \frac{1}{2}\left\{ \left[2\sum_{h \geq u} q(h)\right]^k - \left[2\sum_{h > u} q(h)\right]^k \right\}.
\end{aligned} \tag{2.48}$$

In the second line we used again the symmetry on the signs to write $q\left(|h_j| \operatorname{sign}(h_j)\right) = \frac{1}{2} q\left(|h_j|\right) = q(h_j)$ and to say that $\prod_{j=1}^{k} \operatorname{sign}\left(h_j\right)$ is positive in half of the possible configurations of $h_1, \dots, h_k$, namely 2^{k-1} times. In the third line, we used $\left\{|h_1|, \dots, |h_k| : \min\left\{J, \min_{j=1,\dots,k} |h_j|\right\} = u\right\} = \left\{|h_1|, \dots, |h_k| \geq u\right\} \setminus \left\{|h_1|, \dots, |h_k| > u\right\}$.

Finally,

$$p_k(u) = \begin{cases} 1-[1-q(0)]^k & u=0 \\ 2^{k-1}\left\{\left[\sum_{h\geq u} q(h)\right]^k - \left[\sum_{h>u} q(h)\right]^k\right\} & u>0 \end{cases} \tag{2.49}$$

Variable to factor

Substituting the expression for the MS update in the second equation in (2.59) gives

$$q_d(h) = \sum_{u_1,\ldots,u_d} \sum_{s\in\{\pm 1\}} \delta\left(h, s+\sum_{a=1}^{d} u_a\right) p(s) \prod_{a=1}^{d} p(u_a). \tag{2.50}$$

First notice that $p_s(+1) = p_s(-1) = \frac{1}{2}$, so that

$$q_d(h) = \sum_{u_1,\ldots,u_d} \sum_{s} \frac{1}{2}\delta\left(h, s+\sum_{a=1}^{d} u_a\right) \prod_{a=1}^{d} p(u_a). \tag{2.51}$$

This is a convolution power: define $q_b(h) = \sum_{u_1,\ldots,u_b} \sum_s \frac{1}{2}\delta\left(h, s+\sum_{a=1}^{b} u_a\right) \prod_{a=1}^{b} p(u_a)$ and build a recursive relation

$$\begin{aligned} q_b(h) &= \sum_{u_b} p(u_b) \sum_{u_1,\ldots,u_{b-1}} \sum_{s} \frac{1}{2}\delta\left(h-u_b, s+\sum_{a=1}^{b-1} u_a\right) \prod_{a=1}^{b-1} p(u_a) \\ &= \sum_{u_b} p(u_b) q_{b-1}(h-u_b) \end{aligned} \tag{2.52}$$

with $b \leq d$ and with initial condition

$$q_0(h) = \frac{1}{2}\sum_{s} \delta(h,s) = \frac{1}{2}\left(\delta(h,+1) + \delta(h,-1)\right). \tag{2.53}$$

Denoting by $\circledast$ the convolution operation

$$(f \circledast g)(z) = \sum_{x+y=z} f(x)g(y), \tag{2.54}$$

we get

$$q_d(h) = q_0 \circledast \underbrace{p \circledast \cdots \circledast p}_{d \text{ times}}. \tag{2.55}$$

Thanks to the argument above, the distribution of a belief is analogously given by

$$b_d(h) = q_0 \circledast \underbrace{p \circledast \cdots \circledast p}_{d+1 \text{ times}}. \tag{2.56}$$

2.3.2 Arbitrary degree profile

For an arbitrary degree profile P, Λ, given an edge (a,i) taken uniformly at random, the probability that node i has degree d ($d-1$ neighbors plus a) is $\tilde{\lambda}_d$, the probability of a having $k-1$ neighbors besides i is $\tilde{p}_k$, where

$$\tilde{\lambda}_d = \frac{d\lambda_d}{\sum_{i=1}^{d_{\max}} i\lambda_i}, \quad \tilde{p}_k = \frac{kp_k}{\sum_{i=1}^{k_{\max}} ip_i}. \tag{2.57}$$

These are called *residual* degree distributions in graph theory.

Using these coefficients to weight the incoming probabilities to take into account the degree profile, gives:

$$\begin{aligned} p(u) &\propto \sum_k \tilde{p}_k p_k(u) \\ q(h) &\propto \sum_d \tilde{\lambda}_d q_d(h). \end{aligned} \tag{2.58}$$

Finally, through (2.45), these result in two self-consistency equations for the distributions $p(u), q(h)$

$$\begin{aligned} p(u) &\propto \sum_k \tilde{p}_k \sum_{h_1,\dots,h_k} \delta(u, \hat{f}^{\text{ms}}(h_1,\dots,h_k)) \prod_{i=1}^{k} q(h_i) \\ q(h) &\propto \sum_d \tilde{\lambda}_d \sum_s p_s(s) \sum_{u_1,\dots,u_d} \delta(h, f^{\text{ms}}(u_1,\dots,u_d;s)) \prod_{a=1}^{d} p(u_a). \end{aligned} \tag{2.59}$$

These are the RS cavity equations. Just as in the single-instance case, on a fixed point of these one can compute expectations of observables.

2.3.3 Averages of observables

In the limit $J \to \infty, \beta \to \infty$, the overlap of a typical configuration with the source $O = \sum_{i=1}^{n} s_i \sigma_i$ coincides with minus the free energy density of the system. Recalling

the expression we found in (2.43) for the free energy in terms of max-sum messages, one gets for the overlap

$$\begin{aligned} O = -\frac{\langle F \rangle}{n} =& -\frac{1}{n}\sum_{a=1}^{m}\left\langle \Theta\left(-\prod_{i\in\partial a} h_{ia}\right) 2\min_{i\in\partial a}|h_{ia}| \right\rangle_{\{q(h_{ia})\}_{i\in\partial a}} - \frac{1}{n}\sum_{i=1}^{n}\left\langle (|\partial i|-1)\,|h_i| \right\rangle_{\{q(h_i)\}} \\ &+ \frac{1}{n}\sum_{i=1}^{n}\left\langle \sum_{a\in\partial i}|h_{ia}| \right\rangle_{\{q(h_{ia})\}_{i\in\partial a}} \\ =& -\alpha\sum_k P_k \sum_{h_1,\ldots,h_k}\Theta\left(-\prod_{i=1}^{k}h_i\right) 2\min_{i=1,\ldots,k}|h_i| \prod_{i=1}^{k} q(h_i) \\ &+ \sum_d \Lambda_d \left\{ d\sum_h |h| q_{d-1}(h) - (d-1)\sum_h |h| q_d(h) \right\} \end{aligned} \tag{2.60}$$

where we have spotted the density of clauses or one minus the compression rate $\alpha = \frac{m}{n} = 1 - R$.

The first term can be simplified in a way analogous to (2.48), giving

$$\begin{aligned} \sum_{h_1,\ldots,h_k}\Theta\left(-\prod_{i=1}^{k}h_i\right) 2\min_{i=1,\ldots,k}|h_i|\prod_{i=1}^{k}q(h_i) =& 2^k \sum_{h_1,\ldots,h_k>0} \min_{i=1,\ldots,k}|h_i| \prod_{i=1}^{k} q(h_i) \\ =& 2^k \sum_{h>0} h \sum_{n=1}^{k}\binom{k}{n}(q(h))^n \left(\sum_{h'>h} q(h')\right)^{k-n} \\ =& 2^k \sum_{h>0} h \left\{ \left(\sum_{h'\geq h} q(h')\right)^k - \left(\sum_{h'>h} q(h')\right)^k \right\}. \end{aligned} \tag{2.61}$$

Finally, the overlap is given by

$$\begin{aligned} O =& -\alpha\sum_k P_k 2^k \sum_{h>0} h \left\{ \left(\sum_{h'\geq h} q(h')\right)^k - \left(\sum_{h'>h} q(h')\right)^k \right\} \\ &+ \sum_d \Lambda_d \left\{ d\sum_h |h| q_{d-1}(h) - (d-1)\sum_h |h| q_d(h) \right\}. \end{aligned} \tag{2.62}$$

From the overlap, one can compute the distortion as

$$D = \frac{1}{2}(1-O). \tag{2.63}$$

2.4 Survey propagation and the 1RSB cavity method

As we will see, in some regimes BP and its replica symmetric (RS) variants are not sophisticated enough methods to describe the system at hand. In such cases one can posit a 1-step replica symmetry breaking (1RSB) scenario. Just as at the RS level one has BP (and MS) as an algorithm for single instances and the RS cavity method for ensembles of graphs in the thermodynamic limit, the situation is analogous in the 1RSB formalism. The corresponding algorithm is called survey propagation while the ensemble version takes the name 1RSB cavity method.

2.4.1 Survey propagation

Survey propagation was first introduced as an algorithm to find solutions to the particular problem of K-satisfiability [10, 13]. Since then it has been framed as the algorithmic counterpart of the 1RSB cavity method and it can be derived for a generic model[2] from first principles [1].

The idea is to consider an auxiliary model where variables are the Max-Sum messages $\mathbf{u},\mathbf{h}$ described by

$$\mathbb{P}_y(\mathbf{u},\mathbf{h}) = \frac{1}{\Xi(y)}\mathbb{1}\Big[(\mathbf{u},\mathbf{h}) \text{ fixed-point of max-sum}\Big] e^{-yF(\mathbf{u},\mathbf{h})} \tag{2.64}$$

where sets of messages are given non-zero probability only if consistent with the Max-Sum updates, with a weight depending on the free energy F as defined in (2.36). The so-called Parisi paramter y acts as an inverse temperature for the upper-level system. Finding a good value for y is in general a delicate matter for which rigorous theory is still lacking; see section 2.6.4.

In practice, we have defined a new statistical mechanical model whose variables are messages, given non-zero probability only if they are a solution to the max-sum equations. On this new outer model, we "simply" apply BP.

[2]There are, however, considerable restrictions to those models where the BP/MS equations take a particularly simple form

Auxiliary factor graph

One can ask what is the new factor graph corresponding to the higher-level model. For a given pair (a,i) of neighboring factor and variable in the original graph, the corresponding region in the auxiliary graph is shown in fig. 2.2. Messages, which lived on the edges of the original model, now are the variables.

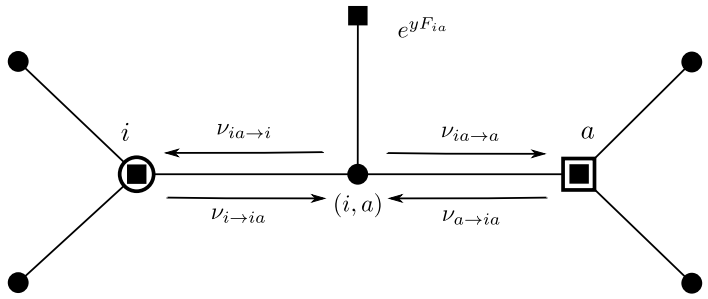


Fig. 2.2 Snapshot of the auxiliary factor graph around a node (i,a)

We can rewrite (2.64) with both the indicator function and the free energy split intro products

$$\mathbb{P}_y(\mathbf{u},\mathbf{h}) = \frac{1}{\Xi(y)} \prod_a \Psi_a\left(\{u_{ai},h_{ia}\}_{i\in\partial a}\right) \prod_i \Psi_i\left(\{u_{ai},h_{ia}\}_{a\in\partial i}\right) \prod_{(i,a)} \Psi_{ia}\left(u_{ai},h_{ia}\right) \tag{2.65}$$

where the factors of the new model are of three types

$$\begin{aligned} \Psi_a &= e^{-yF_a} \prod_{i\in\partial a} \mathbb{1}\left[u_{ai} = \hat{f}^{MS}(\{h_{ja}\}_{j\in\partial a\setminus i})\right] \\ \Psi_i &= e^{-yF_i} \prod_{a\in\partial i} \mathbb{1}\left[h_{ia} = f^{MS}(\{u_{bi}\}_{b\in\partial i\setminus a})\right] \\ \Psi_{ia} &= e^{yF_{ia}} \end{aligned} \tag{2.66}$$

where f^{MS} is shorthand notation for the Max-Sum equations (2.33).

Notice that this can be seen as a Boltzmann distribution

$$\mathbb{P}_y(\mathbf{u},\mathbf{h}) = \frac{1}{\Xi(y)} e^{-yF(\mathbf{u},\mathbf{h})}, \quad \mathbf{u},\mathbf{h} \text{ satisfy MS} \tag{2.67}$$

where the role of the energy is played by the free energy of the underlying model and the role of the (inverse) temperature is played by the real number y. Additionally,

we constrain the new variables, i.e. the old messages, to be satisfying the max-sum equations.

On the auxiliary model, we write the BP equations. In principle there are four types of messages:

- $\nu_{ia\to i}(u_{ai}, h_{ia})$
- $\nu_{i\to ia}(u_{ai}, h_{ia})$
- $\nu_{ia\to a}(u_{ai}, h_{ia})$
- $\nu_{a\to ia}(u_{ai}, h_{ia})$

plus the ones from Ψ_{ia} factors. However, some simplifications are possible. The messages coming from Ψ_{ia} factors can be incorporated in other equations, analogously to what happened for the external fields in the BP equations (2.16). The updates for the first and third kinds of messages are simple enough to also be incorporated in the others: first, we observe that $\nu_{ia\to i}$ is just the product of two incoming messages

$$\nu_{ia\to a}(u_{ai}, h_{ia}) = \frac{1}{z_{ia\to a}} \nu_{i\to ia}(u_{ai}, h_{ia}) e^{yF_{ia}} \tag{2.68}$$

and similarly

$$\nu_{ia\to i}(u_{ai}, h_{ia}) = \frac{1}{z_{ia\to i}} \nu_{a\to ia}(u_{ai}, h_{ia}) e^{yF_{ia}}. \tag{2.69}$$

Furthermore, the dependence on one of the two sets of (old) messages is unnecessary. To show it, we write the BP update for $\nu_{ia\to a}$ and $\nu_{ia\to i}$:

$$\begin{aligned}
\nu_{ia\to a}(u_{ai}, h_{ia}) &= \frac{1}{z_{ia\to a}} e^{yF_{ia}} \nu_{i\to ia}(u_{ai}, h_{ia}) \\
&= \frac{1}{z_{ia\to a}} e^{yF_{ia}} \sum_{\{h_{ib}, u_{bi}\}_{b\in\partial i\setminus a}} \Psi_i(\{u_{ci}, h_{ic}\}_{c\in\partial i}) \prod_{b\in\partial i\setminus a} \nu_{ib\to i}(u_{bi}, h_{ib}) \\
&= \frac{1}{z_{ia\to a}} e^{yF_{ia}} \sum_{\{h_{ib}, u_{bi}\}_{b\in\partial i\setminus a}} e^{-yF_i} \prod_{c\in\partial i} \mathbb{1}\left[h_{ic} = f^{MS}(\{u_{di}\}_{d\in\partial i\setminus c})\right] \prod_{b\in\partial i\setminus a} \nu_{ib\to i}(u_{bi}, h_{ib})
\end{aligned} \tag{2.70}$$

The exponents add up, giving

$$\begin{aligned}
C_{ia} \coloneqq F_i - F_{ia} &= -\frac{1}{\beta}\log\frac{z_i}{z_{ia}} \\
&\overset{(1.13)}{=} -\frac{1}{\beta}\log z_{i\to a} \\
&= -\frac{1}{\beta}\log\left[\sum_{\sigma_i}\prod_{b\in\partial i\setminus a}\frac{e^{\beta u_{bi}\sigma_i}}{2\cosh(\beta u_{bi})}e^{\beta s_i\sigma_i}\right] \\
&\xrightarrow{\beta\to\infty} -\frac{1}{\beta}\log\left[\prod_{b\in\partial i\setminus a}e^{-\beta|u_{bi}|}\max_{\sigma_i}e^{\beta\sigma_i\left(\sum_{b\in\partial i\setminus a}u_{bi}+s_i\right)}\right] \\
&= -\frac{1}{\beta}\log\left[e^{-\beta\sum_{b\in\partial i\setminus a}|u_{bi}|}\right] - \frac{1}{\beta}\log\left[e^{\beta\left|\sum_{b\in\partial i\setminus a}u_{bi}+s_i\right|}\right] \\
&= \sum_{b\in\partial i\setminus a}|u_{bi}| - \left|\sum_{b\in\partial i\setminus a}u_{bi}+s_i\right|
\end{aligned} \tag{2.71}$$

Notice that C_{ia}, which in principle could depend on both the h and u messages, only depends on u_{ai}.

So,

$$\nu_{ia\to a}(u_{ai},h_{ia}) = \frac{1}{z_{ia\to a}}\sum_{\{h_{ib},u_{bi}\}_{b\in\partial i\setminus a}}\prod_{c\in\partial i}\mathbb{1}\left[h_{ic}=f^{MS}(\{u_{di}\}_{d\in\partial i\setminus c})\right]e^{-yC_{ia}\left(\{u_{bi}\}_{b\in\partial i\setminus a}\right)}\prod_{b\in\partial i\setminus a}\nu_{ib\to i}(u_{bi},h_{ib}). \tag{2.72}$$

Similarly,

$$\begin{aligned}
\nu_{ia\to i}(u_{ai},h_{ia}) &= \frac{1}{z_{ia\to i}}e^{yF_{ia}}\sum_{\{u_{aj},h_{ja}\}_{j\in\partial a\setminus i}}\Psi_a\left(\{u_{ak},h_{ka}\}_{k\in\partial a}\right)\prod_{j\in\partial a\setminus i}\nu_{ja\to a}(u_{aj},h_{ja}) \\
&= \frac{1}{z_{ia\to i}}e^{yF_{ia}}\sum_{\{u_{aj},h_{ja}\}_{j\in\partial a\setminus i}}e^{-yF_a}\prod_{k\in\partial a}\mathbb{1}\left[u_{ak}=f^{MS}\left(\{h_{la}\}_{l\in\partial a\setminus k}\right)\right]\prod_{j\in\partial a\setminus i}\nu_{ja\to a}(u_{aj},h_{ja}).
\end{aligned} \tag{2.73}$$

Here the exponents become

$$\begin{aligned} C_{ai} \coloneqq F_a - F_{ia} &= -\frac{1}{\beta}\log\frac{z_a}{z_{ia}} \\ &\overset{(1.10)}{=} -\frac{1}{\beta}\log z_{a\to i} \\ &= -\frac{1}{\beta}\log\left[\sum_{\sigma_i}\sum_{\underline{\sigma}_{a\setminus i}:\mathrm{SAT}}\prod_{j\in\partial a\setminus i}\frac{e^{\beta h_{ja}\sigma_j}}{2\cosh(\beta h_{ja})}\right] \\ &\xrightarrow{\beta\to\infty} -\frac{1}{\beta}\log\left[\prod_{j\in\partial a\setminus i}e^{-\beta|h_{ja}|}\max_{\sigma_i,\underline{\sigma}_{a\setminus i}:\mathrm{SAT}}e^{\beta\sum_{j\in\partial a\setminus i}h_{ja}\sigma_j}\right] \\ &= -\frac{1}{\beta}\log\left[e^{-\beta\sum_{b\in\partial i\setminus a}|h_{ja}|}\right] - \frac{1}{\beta}\log\left[e^{\beta\sum_{b\in\partial i\setminus a}|h_{ja}|}\right] \\ &= 0, \end{aligned} \tag{2.74}$$

giving

$$\nu_{ia\to i}(u_{ai},h_{ia}) = \frac{1}{z_{ia\to i}}\sum_{\{u_{aj},h_{ja}\}_{j\in\partial a\setminus i}}\prod_{k\in\partial a}\mathbb{1}\left[u_{ak} = f^{MS}\left(\{h_{la}\}_{l\in\partial a\setminus k}\right)\right]\prod_{j\in\partial a\setminus i}\nu_{ja\to a}(u_{aj},h_{ja}). \tag{2.75}$$

Notice that $\nu_{ia\to a}$ does not depend on u_{ai}, as well as $\nu_{ia\to i}$ does not depend on h_{ia}. We can therefore make the ansatz that a consistent solution to these equations can be found of the form (taking the opportunity to move to a less cumbersome notation)

$$\begin{aligned} Q_{ia}(h_{ia}) &\coloneqq \nu_{ia\to a}(h_{ia}) \\ P_{ai}(u_{ai}) &\coloneqq \nu_{ia\to i}(u_{ai}) \end{aligned} \tag{2.76}$$

Now the sums on the h_{ib} in (2.72) only act on the respective $|\partial i|-1$ indicator functions giving 1. The same happens for the sums on the u_{aj}'s in (2.75). The equation for Q admits further simplification using the constraint imposed in the delta

$$\begin{aligned} Q_{ia}(h_{ia}) &= \frac{1}{Z_{ia}}\sum_{\{u_{bi}\}_{b\in\partial i\setminus a}}\delta\left(h_{ia} - \sum_{b\in\partial i\setminus a}u_{bi} - s_i\right)e^{-y\left(\sum_{b\in\partial i\setminus a}|u_{bi}| - \left|\sum_{b\in\partial i\setminus a}u_{bi}+s_i\right|\right)}\prod_{b\in\partial i\setminus a}P_{bi}(u_{bi}) \\ &= \frac{1}{Z_{ia}}e^{y|h_{ia}|}\sum_{\{u_{bi}\}_{b\in\partial i\setminus a}}\delta\left(h_{ia} - \sum_{b\in\partial i\setminus a}u_{bi} - s_i\right)e^{-y\sum_{b\in\partial i\setminus a}|u_{bi}|}\prod_{b\in\partial i\setminus a}P_{bi}(u_{bi}) \end{aligned} \tag{2.77}$$

Survey

When working on single graph instances, we would like to obtain marginals for each variable i and use them as decision variables to find a solution. This can be done by adding n extra factor nodes $\{h_i\}_i$ to the auxiliary graph, each attached to one (original) variable i. Messages $i \to h_i$ will contain the information we seek. Messages $h_i \to i$ flowing out from these nodes towards the rest of the graph will be uniform, hence they will not be seen by the rest of the system, keeping the whole probability distribution untouched. We can also imagine to incorporate in factor i the constraint to be fulfilled by MS beliefs, namely $h_i = s_i + \sum_{b\in\partial i} u_{bi}$. The situation is depicted in fig. 2.3.

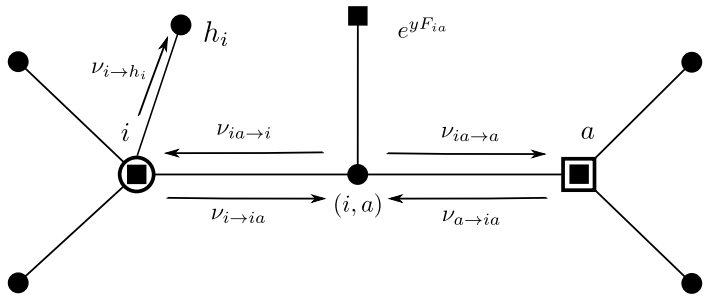


Fig. 2.3 Snapshot of the auxiliary factor graph around a node (i,a) with extra node h_i

Following the rules of BP updates for factors,

$$
\begin{aligned}
B_i(h_i) :=& \nu_{i\to h_i}(h_i) \\
=& \frac{1}{Z_i} \sum_{\{u_{bi},h_{ib}\}_{b\in\partial i}} e^{-yF_i} \left(\prod_{c\in\partial i} \mathbb{1}\left[h_{ic} = f^{MS}(\{u_{di}\}_{d\in\partial i\setminus c}) \right] \right) \mathbb{1}\left[h_i = \sum_{b\in\partial i} u_{bi} + s_i \right] \prod_{b\in\partial i} P_{bi}(u_{bi}) \\
=& \frac{1}{Z_i} \sum_{\{u_{bi},h_{ib}\}_{b\in\partial i}} e^{-y(-|h_i| + \sum_{b\in\partial i}|u_{bi}|)} \left(\prod_{c\in\partial i} \mathbb{1}\left[h_{ic} = f^{MS}(\{u_{di}\}_{d\in\partial i\setminus c}) \right] \right) \\
& \times \mathbb{1}\left[h_i = \sum_{b\in\partial i} u_{bi} + s_i \right] \prod_{b\in\partial i} P_{bi}(u_{bi}) \\
=& \frac{1}{Z_i} e^{y|h_i|} \sum_{\{u_{bi}\}_{b\in\partial i}} e^{-y\sum_{b\in\partial i}|u_{bi}|} \delta\left(h_i - s_i - \sum_{b\in\partial i} u_{bi} \right) \prod_{b\in\partial i} P_{bi}(u_{bi})
\end{aligned}
\tag{2.78}
$$

which is unsurprisingly very similar to the update for Q_{ai}.

Finally, the survey propagation equations read

$$\begin{aligned}
P_{ai}(u_{ai}) &= \frac{1}{Z_{ai}} \sum_{\{h_{ja}\}_{j\in\partial a\setminus i}} \delta\left(u_{ai} - \min_{j\in\partial a\setminus i} |h_{ja}| \prod_{j\in\partial a\setminus i} \operatorname{sign}(h_{ja})\right) \prod_{j\in\partial a\setminus i} Q_{ja}(h_{ja}) \\
Q_{ia}(h_{ia}) &= \frac{1}{Z_{ia}} e^{y|h_{ia}|} \sum_{\{u_{bi}\}_{b\in\partial i\setminus a}} \delta\left(h_{ia} - \sum_{b\in\partial i\setminus a} u_{bi} - s_i\right) \prod_{b\in\partial i\setminus a} P_{bi}(u_{bi}) e^{-y|u_{bi}|} \\
B_i(h_i) &= \frac{1}{Z_i} e^{y|h_i|} \sum_{\{u_{bi}\}_{b\in\partial i}} \delta\left(h_i - s_i - \sum_{b\in\partial i} u_{bi}\right) \prod_{b\in\partial i} P_{bi}(u_{bi}) e^{-y|u_{bi}|}.
\end{aligned} \tag{2.79}$$

The computational complexity of the updates in (2.79) is manifestly exponential in the factor and node degrees. In appendix A.4, we show how such cost can be brought down to polynomial.

2.4.2 Average overlap

We wish to compute the free energy of the lower-level system averaged over the distributions resulting from a fixed point of SP variable and factor beliefs, in order to obtain a probability distribution to average over. Variable beliefs are given by the product of incoming messages

$$\begin{aligned}
B_{ia}(h_{ia}, u_{ai}) &= \frac{1}{Z_{ia}} e^{yF_{ia}} \nu_{i\to ia} \nu_{a\to ia} \\
&= \frac{1}{Z_{ia}} e^{yF_{ia}} \nu_{ia\to a} e^{-yF_{ia}} \nu_{ia\to i} e^{-yF_{ia}} \\
&= \frac{1}{Z_{ia}} e^{-yF_{ia}(u_{ai}, h_{ia})} Q_{ia}(h_{ia}) P_{ai}(u_{ai}) \\
&= \frac{1}{Z_{ia}} e^{y(|h_{ia}+u_{ai}|-|h_{ia}|-|u_{ai}|)} Q_{ia}(h_{ia}) P_{ai}(u_{ai}).
\end{aligned} \tag{2.80}$$

Factor beliefs are of two types, one for ex-variables i and one for ex-factors a

$$\begin{aligned}
B_i(\{u_{ai}\}_{a\in\partial i}) &= \frac{1}{Z_i} \Psi_i \prod_{a\in\partial i} P_{ai}(u_{ai}) = \frac{1}{Z_i} e^{-yF_i(\{u_{ai}\}_{a\in\partial i})} \prod_{a\in\partial i} P_{ai}(u_{ai}) \\
B_a(\{h_{ia}\}_{i\in\partial a}) &= \frac{1}{Z_a} \Psi_a \prod_{i\in\partial a} Q_{ia}(h_{ia}) = \frac{1}{Z_a} e^{-yF_a(\{h_{ia}\}_{i\in\partial a})} \prod_{a\in\partial i} P_{ai}(u_{ai})
\end{aligned} \tag{2.81}$$

where we dropped the indicator functions in Ψ_i and Ψ_a since they were functions of the "other" set of messages ($\{h_{ia}\}$ for Ψ_i and $\{u_{ai}\}$ for Ψ_a).

To compute the average overlap, we need to compute the average of F, the free energy of the original model, over the probability distribution $\underline{B}$ we just computed

$$\begin{aligned}\langle F\rangle_{\underline{B}} &= \left\langle \sum_a F_a(\{h_{ia}\}) + \sum_i F_i(\{u_{ai}\}) - \sum_{(ia)} F_{ia}(h_{ia},u_{ai}) \right\rangle_{\underline{B}} \\ &= \sum_a \langle F(\{h_{ia}\})\rangle_{B_a} + \sum_i \langle F_i(\{u_{ai}\})\rangle_{B_i} - \sum_{(ia)} \langle F_{ia}(h_{ia},u_{ai})\rangle_{B_{ia}}\end{aligned} \tag{2.82}$$

and

$$O = -\frac{1}{n}\langle F\rangle_{\underline{B}}. \tag{2.83}$$

Some further simplifications are reported in appendix A.5.

2.4.3 1RSB free energy and complexity

The 1RSB formalism gives access to two more quantities, playing the role of free energy and entropy for the auxiliary model whose variables are BP messages. The 1RSB free energy is related to the logarithm of the partition function $\Xi(y)$ of distribution (2.64)

$$F_e^{\text{1rsb}}(y) = -\frac{1}{y}\log \Xi(y). \tag{2.84}$$

$\Xi(y)$ can be obtained by means of a Bethe factorization analogous to the one in (1.26).

The role of the entropy is played by the *complexity* $\Sigma(y)$, which is related to the number of solutions at a given "inverse temperature" y. It can be computed as

$$\Sigma(y) = y\left[-O(y) - F_e^{\text{1rsb}}(y)\right] \tag{2.85}$$

where $O(y)$ is the average overlap, corresponding to the free energy of the underlying model and to minus the energy of the outer one. A complexity value of zero indicates that there is only one solution to the max-sum equations. Negative values of the complexity are a good diagnostic for wrong assumptions: it might be the case, for instance, that a 1RSB picture is not appropriate.

2.4.4 The 1RSB cavity method and population dynamics

Analogously to what we have seen at a replica symmetric level, the recipe to deal with probabilistic ensembles of models in the thermodynamic limit is to write self-consistency equations for the distributions of SP messages. These are:

$$\begin{aligned}\mathscr{P}(P) &= \int DQ_1 \cdots DQ_k \mathbb{1}\left[P = \hat{f}^{SP}(Q_1,\ldots,Q_k)\right] \prod_{j=1}^{k} \mathscr{Q}(Q_j) \\ \mathscr{Q}(Q) &= \int DP_1 \cdots DP_d \mathbb{1}\left[Q = f^{SP}(P_1,\ldots,P_d)\right] \prod_{b=1}^{d} \mathscr{P}(P_b)\end{aligned} \tag{2.86}$$

where $\hat{f}^{SP}$ and f^{SP} are shorthand notation for (2.79). These equations always admit a trivial fixed-point

$$\begin{aligned}\mathscr{P}(P) &= \sum_h p(h)\delta[P, \delta(\cdot - h)] \\ \mathscr{Q}(Q) &= \sum_h q(u)\delta[Q, \delta(\cdot - u)] \ ,\end{aligned} \tag{2.87}$$

with $p(u)$ and $q(h)$ solution of the RS equation (2.59). In the RS phase, this trivial fixed-point is the unique solution, while in the 1RSB phase, the trivial solution becomes unstable and the above equations admit a non-trivial solution.

It is clear that, since SP messages are complicated distributions themselves, managing distributions over those is unfeasible. The typical strategy is then to keep a finite population of SP messages (actually, one population of P messages and one population of Q messages) playing the role of a finite-sample approximation to the distribution of distributions. This finite set of messages is updated by an iterative procedure:

1. Sample a factor node degree k from the residual distribution (2.57)
2. Sample k messages from the Q population
3. Compute the outgoing P message according to the first SP equation in (2.79)
4. Replace a message at random in the P population with the newly-computed one
5. Repeat

and analogously for Q-messages. The procedure is iterated until some convergence criterion is met, for instance the average first moment of the population reaches a stationary point. Beliefs, which are then used to estimate observables, can either be computed on the converged population, or during the iterations together with the messages.

Notice that in practice, for efficiency reasons, we used a slightly different procedure. One can sample the degree $k+1$ from the degree profile instead of the residual distribution, then sample $k+1$ Q messages. Each combination (there are $k+1$ of them) of all-but-one of these, passed through the SP update, gives an outgoing P message. The $k+1$ resulting P messages are then substituted in block in their the population. Since in our implementation of SP computing one outgoing message is as expensive as computing all the outgoing messages from a node, this strategy helped with faster convergence.

The description of the 1RSB cavity method concludes the methodological part of this chapter. In the following we begin employing the tools just described to study instances of CVP.

2.5 A simple case: cycle codes

We start our analysis with a family of linear systems called cycle codes which were already studied in [46]. They correspond to systems of linear equations (in $GF(2)$) in which each variable participates in at most 2 equations. In the graphical representation, a cycle code is a bipartite graph $\mathscr{G} = ((V \cup F), E)$ in which each variable node $i \in V$ has degree ≤ 2. This particular ensemble has a simple structure that allows to provide exact results. In particular, we know that max-sum is exact under some conditions, as already shown in [47]. We also present a greedy optimal algorithm that, by leveraging a mapping onto another optimization problem on graphs, is guaranteed to find the optimal solution in polynomial time.

2.5.1 A greedy but optimal algorithm

We start by describing a greedy optimal (GO for short in the figures) algorithm to solve the optimization problem

$$\min_{\mathbf{x}:H\mathbf{x}=0} E(\mathbf{x}) = \min_{\mathbf{x}:H\mathbf{x}=0} \sum_{i=1}^{n} x_i \oplus y_i, \tag{2.88}$$

which is the $\{0,1\}$-variable version of minimizing the p-spin energy (2.12).

It is an iterative strategy which outputs a codeword at each iteration, with the guarantee that the energy $E(\mathbf{x})$ either decreases with respect to the previous iteration, or is at the minimum.

As a preliminary step, if there are any leaves, i.e. variables of degree 1, we add a constraint connected to all of them. This amounts to adding a new row to matrix H but does not change the space of solutions: it is equal to the sum of all the rows where the leaves were involved, hence it is linearly dependent on the others.

Now that all variables have degree exactly 2, it is convenient to consider a slightly simplified factor graph $\mathscr{G}' = (F, E')$ were vertices are the old set of factors F, and the edges $e' = (a,b) \in E'$ are linking two ex-factors $a, b \in F$ through the variable node $i \in V$ that was attached to them in the original graph $\mathscr{G}$. At this point, a set of edges on $\mathscr{G}'$ corresponds to a set of variables in $\mathscr{G}$. Two observations are due: the first is that cycles on $\mathscr{G}'$ correspond to cycles on $\mathscr{G}$. The second is that, given a cycle on $\mathscr{G}$, a vector with 1's on the variables touched by the cycle and 0's elsewhere is a codeword, because for each clause the number of connected variables that are 1 is even, thereby satisfying all XOR constraints.

For the iterative part of the algorithm, one keeps a current guess $\mathbf{x}$ for the optimal codeword, which is updated at each step by summing it (modulo 2) with another codeword. This amounts to flipping the bits in one codeword that are equal to one in the other (and vice-versa). More specifically, let us first define a weight function on the edges of $\mathscr{G}'$ $w_{(\mathbf{x},\mathbf{y})} : E' \to \{-1,1\}$ which also depends on the current guess $\mathbf{x}$ and on the source vector $\mathbf{y}$. Edge $e'_i \in E'$, which as we saw is in one-to-one correspondence with variable $i \in V$, is given weight

$$\begin{aligned} w_{(\mathbf{x},\mathbf{y})}(e'_i) &= 1 - 2x_i \oplus y_i \\ &= \begin{cases} +1 & \text{if } x_i = y_i \\ -1 & \text{if } x_i \neq y_i. \end{cases} \end{aligned} \tag{2.89}$$

Now, if there exists a cycle $\mathscr{L} \subseteq E'$ where the sum of the weights of the included edges is negative, the corresponding cycle on $\mathscr{G}$ identifies a codeword $\mathbf{x}_{\mathscr{L}}$. It is easy

to see that the new codeword obtained by summing it to the previous guess gives a new codeword $\mathbf{x}' = \mathbf{x} \oplus \mathbf{x}_{\mathscr{L}}$ with a strictly smaller energy. The procedure can be iterated until no negative cost cycle is present in the auxiliary graph, at which point the algorithm stops and the current guess is the solution. Crucially, negative cost cycles can be found in polynomial time, as shown in [48]. [AB: Converges within n steps].

2.5.2 Comparison of cavity predictions and algorithmic performances on single instances

We focus on cycle codes with a factor degree profile $P = \{p_k, p_{k+1}\}$, where k is a positive integer and $p_k, p_{k+1} \in [0,1]$ with $p_k + p_{k+1} = 1$. The variable degree profile is $\Lambda = \{\lambda_2 = 1\}$, meaning each variable node has a fixed degree of 2. The compression rate for this family of instances is expressed as a function of k and p_{k+1}:

$$R(k, p_{k+1}) = 1 - \frac{2}{k + p_{k+1}}. \tag{2.90}$$

By varying k and p_{k+1}, the full range $R \in [0,1]$ can be covered. Figure 2.4 shows the performance of the greedy optimal (GO) algorithm (red circles) and MaxSum with reinforcement (green squares). These results are compared with the replica symmetric prediction (gray dashed line). The results in Figure 2.4 are shown in the rate-distortion plane introduced earlier (see Figure 2.1). The blue line represents the exact rate-distortion bound, given by (2.5), which shows the minimum distortion achievable at a given rate R. The red line corresponds to the distortion achieved with the trivial compression strategy described in Section 2.1.2.

It is worth noting that the MaxSum results show slightly higher distortion than the GO algorithm, due to the fact that MaxSum does not converge for all instances. In contrast, the GO algorithm provides the exact solution in all cases. When MaxSum fails to converge, the strategy was to separate the variables into independent and dependent sets, as discussed in section 2.2.4. After running the MaxSum on (2.33) for some time without convergence, the independent variables were fixed based on their beliefs, and the dependent variables were adjusted to satisfy the linear constraints.

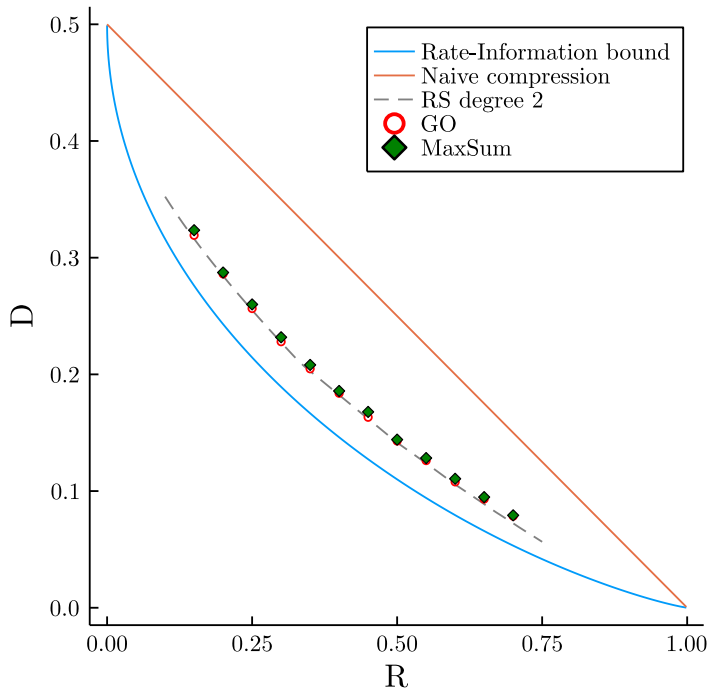


Fig. 2.4 Results for cycle codes: rate-distortion performance for the GO and Max-Sum algorithms on graphs of size $n = 1800$, with degree profile $\Lambda(x) = x^2$ and $P(x) = p_k x^k + p_{k+1} x^{k+1}$. Points represent the average over 20 random graphs and source vectors.

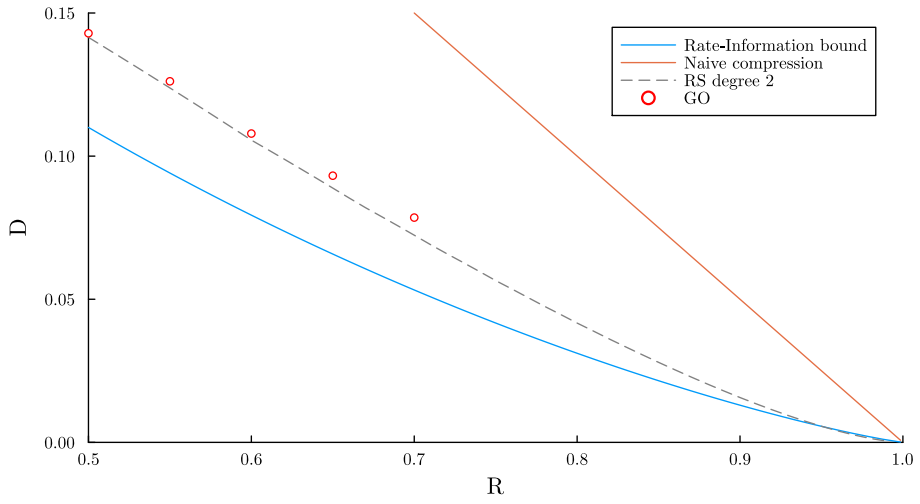


Fig. 2.5 Detail of the rate-distortion diagram at large rate for cycle codes. Two observations suggest that replica symmetry is not the correct framework: points obtained with the GO algorithm start to deviate from the RS prediction, and the RS prediction falls below the Shannon bound for rates close to 1.

For rates up to $R = 0.6$ (i.e., $k \leq 5$), we observe good agreement between the RS cavity prediction and the results of both algorithms. Numerical analysis showed that for $R < 0.6$, the unique solution of the 1RSB equations (2.86) is the trivial RS solution (2.87), confirming that we remain in the Replica Symmetric phase in this regime.

For higher rates ($R > 0.6$), several indicators suggest that the RS solution is no longer accurate. First, there is a discrepancy between its predicted average distortion and the results obtained with GO on large instances (size $n = 16000$). Above $R = 0.95$, the distortion predicted by the RS ansatz even falls below the exact rate-distortion lower bound. While it is possible that an RSB transition occurs as k increases, we were unable to confirm this hypothesis due to convergence issues when solving the 1RSB equations numerically at higher rates. It might also be that in this regime one cannot exchange the thermodynamic and zero-temperature limit, and therefore that the cavity method is not correct.

2.6 Moving to higher degrees

In the previous section, we demonstrated that the constrained optimization problem CVP on cycle codes, where variable nodes have a degree of at most 2, can be solved exactly. In this section, we turn our attention to random graph ensembles with variable nodes of higher degree. We will show that transitioning to these ensembles has two effects: it enables to reach lower minimal energies for a given rate, at the same time it adds some computational complications.

We analyze random graph ensembles with a variable degree profile $\Lambda = \{\lambda_2, \lambda_3\}$, meaning a fraction λ_2 of variable nodes have degree 2, and a fraction $\lambda_3 = 1 - \lambda_2$ have degree 3. The factor degree profile is $P = \{p_3 = 1\}$, indicating all factor nodes have a fixed degree of 3. This represents the simplest non-trivial phase diagram. For this ensemble, the compression rate can be written as a function of the fraction of degree 3 variables:

$$R(\lambda_3) = \frac{1 - \lambda_3}{3} \in [0, 1/3]. \tag{2.91}$$

We will compare this random graph ensemble to cycle codes, characterized by the variable degree profile $P = \{p_3 = 1\}$ and $\Lambda = \{\lambda_1, \lambda_2\}$, where a fraction λ_1 of variable nodes has degree 1, and a fraction $\lambda_2 = 1 - \lambda_1$ has degree 2. In this ensemble, the rate is expressed in terms of the fraction of degree 1 variables:

$$R(\lambda_1) = \frac{1 + \lambda_1}{3} \in [1/3, 2/3]. \tag{2.92}$$

2.6.1 Results from the Cavity Method

In Fig. 2.6, the left panel presents the results of the cavity method, using both the RS formalism (solid lines) and the 1RSB formalism (circles). For the ensemble with degree 1 and 2 variable nodes corresponding to the rate in equation (2.92) (in green), we observe that the RS and 1RSB predictions coincide, as the only 1RSB solution found for this ensemble is the trivial RS solution . The minimal distortion for this

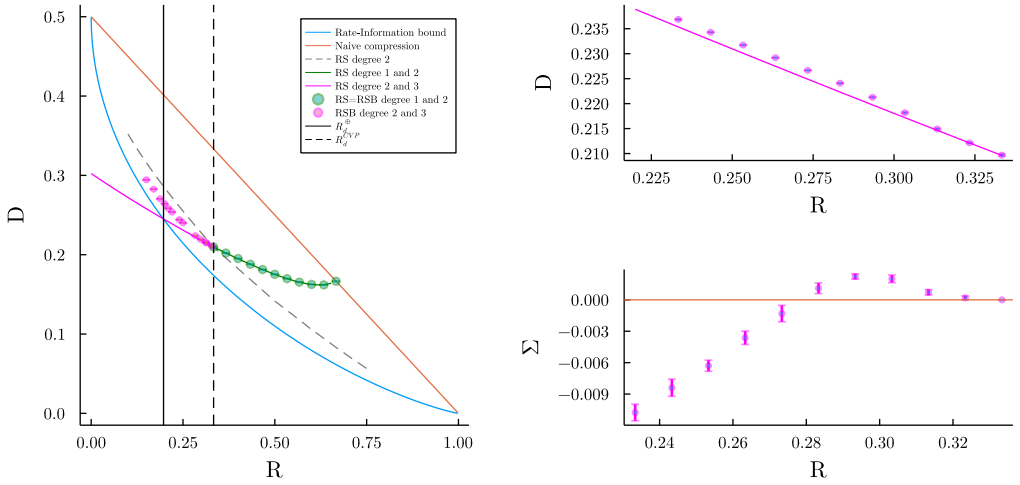


Fig. 2.6 Left panel: zero-temperature cavity prediction of the minimal distortion. In green: for variable nodes of degree 1 and 2 (2.92), in pink: for variable nodes of degree 2 and 3 (2.91). 1RSB points are obtained by optimizing the Parisi parameter y, see 2.6.4. Solid lines represent the RS prediction, circles represent the 1RSB prediction. The gray dashed line is the RS prediction for the ensemble from the previous section (2.90). The two vertical lines correspond to the clustering transitions for CVP (at $R_d^{\mathrm{CVP}} = 1/3$, dashed line) and for the XORSAT problem (at $R_d^{\oplus} \approx 0.197$, solid line). Right panel: zoom-in near $R = 1/3$, showing fixed $y = 1.0$ results for minimal distortion (top) and complexity (bottom), indicating a continuous clustering transition.

graph ensemble is higher than for the ensemble studied in the previous section, with rate (2.90), shown by the gray dashed line.

The more interesting case is the graph ensemble with variable nodes of degree 2 and 3, corresponding to (2.91). The RS prediction (pink line) is evidently unphysical at small rates, as it falls below the rate-information bound. Therefore, a 1RSB formalism is (at least) required to produce a reliable prediction of the minimal distortion in this case: pink circles represent the 1RSB solution, calculated using the optimal Parisi parameter $y = y^{\mathrm{opt}}(R)$ (see discussion in 2.6.4). The system enters a 1RSB phase when a non-zero fraction of degree 3 variables is introduced, as detailed near $\lambda_3 = 0$ in the right panel of Fig. 2.6.

Decreasing the rate below the critical value of $R_d^{\mathrm{CVP}} = 1/3$, marked by the vertical dashed line in fig. 2.6, left panel, a non-trivial 1RSB solution arises. The right panel displays a detail close to the CVP transition (upper-right) and the complexity computed by the 1RSB cavity method (lower-right), showing that such transition is first order. This 1RSB prediction is confirmed through finite-size analysis, presented in section 2.6.3. We were able to reach the physical solution down to $R = 0.15$ (left-most pink circle). Below this rate, no physical solution was found. Additional details on the numerical resolution of the 1RSB equations, particularly for small rates, are provided in section 2.6.4.

Decreasing the rate further to $R_d^{\oplus} \approx 0.197$, one encounters the dynamical-1RSB transition in the XORSAT problem[3] corresponding to the measure (2.11) without the external field, at $\beta = 0$ (solid vertical line in fig. 2.6).

There is a rate range $R \in [R_d^{\oplus}, R_d^{\mathrm{CVP}}]$ where the constrained optimization problem is in a 1RSB phase, despite the underlying XORSAT problem defining the set of constraints being replica symmetric. This outcome is particularly intriguing because it indicates that the structure of the constraint set alone does not fully explain the complexity of the constrained optimization problem. As we will see in the next section, the role played by these critical rate values is even more important when considering the performance of optimization algorithms on single instances.

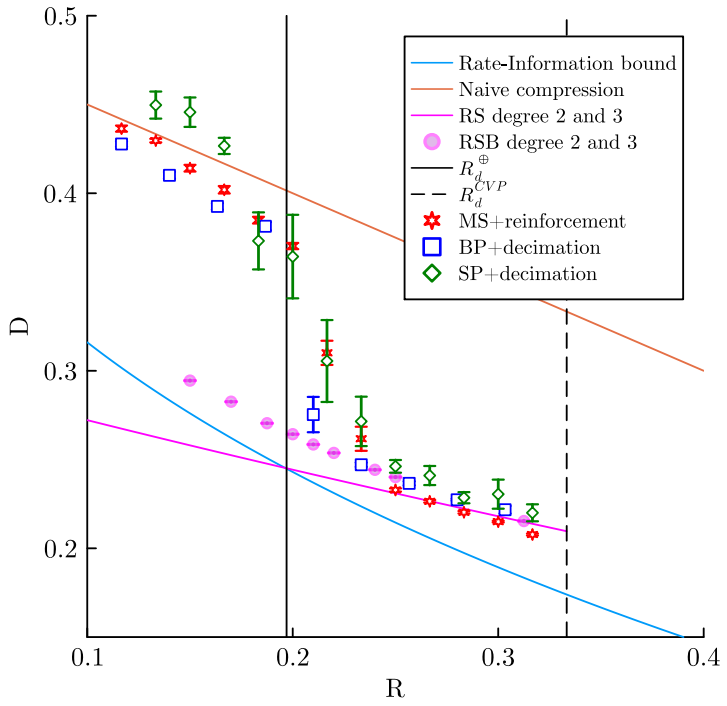


Fig. 2.7 Rate-distortion performance for max-sum with reinforcement (red stars), for Belief Propagation with decimation at $\beta = 3$ (blue squares), and survey propagation at y maximizing the 1RSB free energy (green diamonds), on graphs of size $n = 1800$, degree profile $\Lambda = \{\lambda_2, \lambda_3\}$, $P = \{p_3 = 1\}$. Points are the average over 15 random graphs and source vectors.

2.6.2 Algorithmic results

Figure 2.7 is perhaps the most significant plot of this chapter. It illustrates the performance of three algorithms: max-sum with reinforcement (represented by red stars), belief propagation with decimation at a finite but large inverse temperature $\beta = 3$ (blue squares), and survey propagation with decimation (green diamonds) at $y = y^{\mathrm{opt}}(R)$ (refer to the discussion in 2.6.4). Additionally, the results from the cavity method using the 1RSB ansatz are shown as pink circles.

The two critical values of the rate $R_d^{\oplus} = 0.25, R_d^{\mathrm{CVP}} = 1/3$ divide the phase diagram into three regions:

- For rate $R > R_d^{\mathrm{CVP}}$, the problem is in a replica symmetric phase: RS algorithms are exact, 1RSB ones collapse on the trivial RS solution.
- In the intermediate regime $R_d^{\oplus} < R < R_d^{\mathrm{CVP}}$ the CVP is in a dynamical 1RSB phase, but the underlying XORSAT problem is not. Quite surprisingly, BP and MS keep performing fairly well, and resorting to SP does not provide a significant advantage.
- For $R < R_d^{\oplus}$ also the underlying XORSAT problem crosses a dynamical transition. The performance of all algorithms degrades: it is comparable with the trivial compression scheme.

It is worth emphasizing that the algorithmic transition for CVP does not coincide with the prediction by the 1RSB cavity method, which occurs at $R_d^{\mathrm{CVP}} = 1/3$. Instead, it aligns with the clustering transition in the XORSAT problem at $R_d^{\oplus} \approx 0.197$, which governs the constraints but is unrelated to the optimization function. In order to obtain a complete description of the transition happening in the space of codewords, we conjecture that a full RSB formalism [49, 50] might be needed. That, however, would be computationally prohibitive. We can nonetheless hypothesize what happens to the geometry of the space of solutions as follows.

In the range $[R_d^{\oplus}, R_d^{\mathrm{CVP}}]$, the set of codewords is well-connected. However, despite the simplicity of the optimization function (2.12), the problem is found in a glassy phase, where the energy landscape features many local minima separated by free-energy barriers. Optimization algorithms can still function efficiently: the

[3] This threshold can be computed by means of the leaf removal algorithm [12].

solution space is well-connected through paths of solutions, even if they are not highly optimized, and the continuous phase transition induces correlations that polynomial algorithms can effectively approximate. As the clustering transition at $R_d^{\oplus}$ approaches, the solution space takes on a sponge-like structure, with small corridors connecting regions that will soon become clusters. With the space of solutions organized with such a topology, the effect of the external field is strong, creating free energy barriers hard to overcome by message-passing algorithms. When $R < R_d^{\oplus}$, the most abundant solutions cluster, making jumps between solutions increasingly difficult and creating substantial algorithmic barriers. In this regime, the effects of the random first-order transition (RFOT) become evident, and all algorithms become trapped in high-distortion solutions.

What we find most interesting about the results in fig.2.7 is that BP and MS, whose degree of approximation is that of a RS ansatz, still perform well inside the dynamical 1RSB phase. We propose two possible explanations. The first follows from the geometric argument above: although the "thermodynamically correct" solutions are already separated by barriers, local algorithms can exploit a network of well-connected, although suboptimal, solutions. The second explanation is that these versions of BP and MS are not actually local, due to the strategy we used in case of non-convergence (finding a basis for the set of codewords, fixing the independent variables according to their beliefs, the dependent ones are fixed consequently[4]): computing a basis for the space of codewords is a global operation, made possible thanks to the vector space structure of the codewords. In this regime we indeed made heavy used of this strategy, as BP and MS often failed to converge.

Finally, we observe that in fig.2.7, for rates around $R = 0.3$, max-sum achieves a distortion lower than the 1RSB prediction for $n \to \infty$. This might also due to the uncorrectedness of the 1RSB picture.

2.6.3 Exact enumeration

While in the case of cycle codes the theoretical prediction provided by the cavity method could be matched by algorithmic results, in this regime none of our polynomial-time heuristic is able to reach low enough distortions. We therefore turned to an exhaustive approach. We conducted an exact enumeration for small graph sizes and

[4]See section 2.2.4

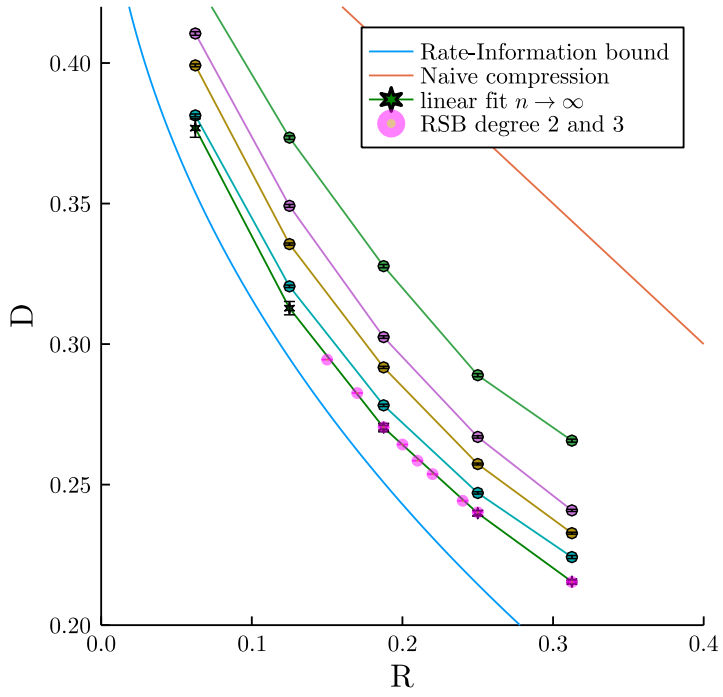


Fig. 2.8 Exact enumeration. The four top lines with points correspond to (from top to bottom) $n = 16, 32, 48, 96$. Further down, the $n \to \infty$ predictions are made by means of a linear fit in $1/n$. In pink: prediction of the zero-temperature 1RSB cavity method.

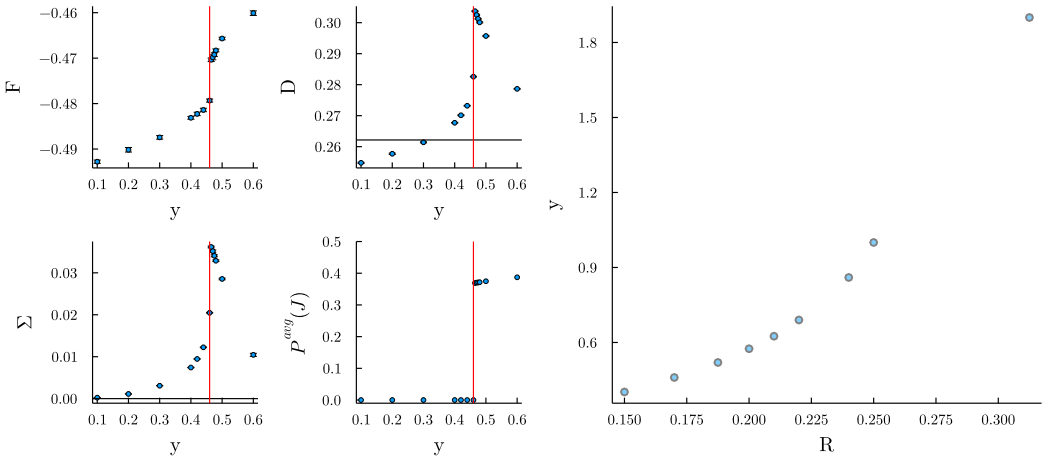


Fig. 2.9 Left panel: y dependence of the RSB solution for $R = 0.17$. Top left: 1RSB free-energy F_e^{1rsb}, top right: internal distortion $D_{\text{int}}(y)$, where the horizontal black line represents the rate-distortion bound $D_I(R)$. Bottom left: complexity $\Sigma_e(y)$, bottom right: weight $P^{\text{avg}}(J)$. Right panel: optimal value of y computed for several rates. Population size is $5 \cdot 10^5$ and $J = 20$.

a finite-size study of the random graph ensemble with degree profile $\Lambda = \{\lambda_2, \lambda_3\}$ and $P = \{p_3 = 1\}$ to compare with the results of the 1RSB cavity method.

Figure 2.8 presents the exact results for graph sizes $n \in \{16, 32, 48, 96\}$. The data are averaged over several random instances from the graph ensemble $\mathbb{G}_n(\Lambda, P)$. For each instance, the solution set of the associated XORSAT problem is computed exactly, and the solution with the minimal distortion is extracted. A linear extrapolation is applied to approximate the large-size limit, which shows good agreement with the 1RSB cavity prediction.

This exact enumeration procedure allows us to provide predictions for rates smaller than $R = 0.15$, below which the 1RSB cavity method fails to yield a physical solution. We suspect that the correct physical solution may require an infinite replica symmetry breaking, so instabilities in the 1RSB solution at very small rates are unsurprising.

2.6.4 Instability in the zero-temperature solution

To numerically solve the 1RSB cavity cavity equations, we employed the softened measure (2.13), which allows us to represent populations of Max-Sum messages as

finite vectors of size $2J+1$. In the large β limit, the softened measure (2.13) focuses on configurations that minimize the energy, as described in (2.14). In the 1RSB phase, these configurations are organized into an exponential number of clusters, separated by free-energy barriers.

Choosing the appropriate value of y to correctly describe this cluster decomposition is a delicate task. Based on the foundational work in [51], the 1RSB free-energy F_e^{1rsb}, defined in (2.84), should be *maximized* with respect to y. In practice, for each value of the rate shown in Fig. 2.6, we analyzed the y-dependence of the 1RSB equation solutions (2.86) for the random graph ensemble with degree profiles $\Lambda = \{\lambda_2, \lambda_3\}$ and $P = \{p_3 = 1\}$. Figure 2.9 (left panel) illustrates this y-dependence for rate $R = 0.17$, i.e. in the clustered phase $R < R_d^{CVP}$. In the right panel, the optimal value of y is shown as a function of the rate. For each rate, the 1RSB free-energy $F_e^{\text{1rsb}}(y)$ and complexity $\Sigma_e(y)$ (2.85) were computed. Additionally, the internal distortion $D_{\text{int}}(y)$, derived from the internal energy as $D_{\text{int}}(y) = 2O(y) - 1$, where $O(y)$ represents the overlap corresponding to the internal energy of the outer model (see (2.85)), was evaluated. With the correct choice of y, $D_{\text{int}}(y)$ provides a prediction for the minimal distortion. Furthermore, the averaged distribution of cavity fields was computed as:

$$P^{\text{avg}}(h) = \int \mathrm{d}\mathscr{P}^{\text{1rsb}}(P)P(h). \tag{2.93}$$

From this analysis, we determined the optimal value of y for each rate, $y_{\text{opt}}(R)$, as the value that maximizes the free energy $F_e^{\text{1rsb}}(y)$, while adhering to the following constraints: the complexity is positive, $\Sigma_e(y) > 0$; the internal distortion exceeds the rate-distortion bound, $D_{\text{int}}(y) > D_I$, where D_I is the solution of equation (2.5); and the averaged distribution P^{avg} has no weight at $\pm J$.

This last constraint is essential because the Max Sum beliefs h_i represent the difference between the ground-state energies when the variable σ_i is flipped: $h_i = E_i^{\min}(+) - E_i^{\min}(-)$. If $h_i = \pm J$, flipping the variable i either requires rearranging $O(J)$ other variables or violating a constraint (see the definition of the soft energy function in 2.14). Therefore, we discard solutions with non-zero weight at $\pm J$, as they might correspond to configurations that do not satisfy all constraints.

For the case of $R = 0.17$, we observe that $F_e^{\text{1rsb}}(y)$ increases smoothly with y, up to $y = 0.46$, where there is a sharp transition toward a solution with non-zero

weight in $P^{\text{avg}}(\pm J)$. Since all the constraints are met at $y = 0.46$, the optimal value is $y^{\text{opt}}(R = 0.17) = 0.46$. However, as the rate decreases, determining y_{opt} becomes increasingly challenging due to the growing sharpness of the jump in $F_e^{\text{1rsb}}(y)$, reducing the accuracy of this determination.

For rates smaller than $R = 0.15$, we were unable to find a value of y that satisfied all the constraints. Despite our efforts, we could not identify a physical 1RSB solution, leading us to hypothesize that a transition to a 2RSB or even a full RSB phase may occur at smaller rates. Further investigation into this phenomenon is left for future work.

2.7 Conclusion

In this chapter, we explored the Closest Vector Problem (CVP), an emblematic task in discrete optimization with applications in cryptography and complexity theory, and a connection to data compression and satisfiability formulas. Despite its simple formulation, the problem presents a rich and complex structure, characterized by phase transitions and clustering phenomena. By mapping the problem to that of finding the ground state of a p-spin Ising model, we were able to analyze the problem using the cavity method. Unlike in many other constrained satisfaction problems, the set of constraints in CVP has a vector space structure. This allowed us to leverage linear algebraic tools such as gaussian elimination to help in the study.

For a certain family of random graphs, called cycle codes, we show that the problem is well described by a replica symmetric ansatz, and can be solved exactly in polynomial time. For more complicated ensembles, finding both the minimum energy and a minimizing configuration becomes highly non-trivial. Using replica symmetry breaking-level approximations, we explored how phase transitions in the geometric properties of the space of solutions affect both analytical predictions and algorithmic performance.

Overall, the work presented in this chapter constitutes a step in understanding the inherent complexity of CVP and offered insights into solving it through polynomial-time heuristics.

Part II

Non-equilibrium

Chapter 3

Reweighted Markov processes on graphs

While in the previous chapter we employed the cavity method to study a system at equilibrium, we now turn to a dynamical problem. In particular, we will be interested in a family of stochastic dynamics which are simple enough to study, but present consistent computational hurdles once their "reweighted" version is considered. We will see which physical systems are well described by such reweighted processes, what the mathematical challenges are, and finally present an approximate method to compute marginal distributions and other observables.

3.1 Markov processes on graphs

Markov processes on graphs have been used extensively in the literature to describe a variety of phenomena such as epidemic spreading, reaction-diffusion systems in biology, opinion dynamics, dynamics of spin systems, and more. See e.g. [52, 53] for extensive reviews. We will be concerned with Markov processes for discrete variables, with discrete time evolution.

Let us consider a set of discrete variables $\mathbf{x} = x_1, \dots, x_N$, $x_i \in \{1, 2, \dots, q_i\}$, found in some configuration at time $t = 0$, then evolving through T discrete time steps, according to some stochastic rule. Any single realization of such process, called a *trajectory*, is uniquely identified by the state in which every variable is found at

every time step. The space of all trajectories is therefore denoted mathematically as

$$\chi = \{x_i^t\}_{\substack{i\in\{1,2,\dots,N\}\\ t\in\{0,1,\dots,T\}}}. \tag{3.1}$$

In the rest of this chapter, we will mostly use bold letters to indicate multiple variable indices $\mathbf{x}_A \equiv \{x_j\}_{j\in A}$ (the set of all variables is $\mathbf{x} \equiv \mathbf{x}_{\{1,2,\dots,N\}}$) and overbars for multiple times indices $\bar{x}_i \equiv \{x_i^t\}_{t=0:T}$. A full trajectory is denoted by $\bar{\mathbf{x}}$.

A stochastic process is nothing but a probability distribution over the space of trajectories $p : \chi \to \{0,1\}$. Such distribution assigns a probability to each realization (trajectory) of the process.

By means of repeated applications of the definition of conditional probability, one can always re-write a stochastic process in an "autoregressive" form

$$p(\bar{\mathbf{x}}) = \prod_{t=0}^{T} p(\mathbf{x}^t | \{\mathbf{x}^u\}_{u<t}) \tag{3.2}$$

where the 0-th term involves no conditioning. Whenever the conditional probabilities $p(\mathbf{x}^t|\{\mathbf{x}^u\}_{u<t})$ are known in closed form, it means that there exists a simple (stochastic) time-evolution rule expressing the system's state at time t given the state at previous times[1]. This is often informally referred to as *causality* property. A prominent example is given by *Markov* processes, the main subject of this chapter, for which it holds that

$$p(\mathbf{x}^t | \{\mathbf{x}^u\}_{u<t}) = p(\mathbf{x}^t | \mathbf{x}^{t-1}). \tag{3.3}$$

In words, the rule for the time evolution between times t and $t+1$ only depends on the state of the system at time t and not on the full history. Then the distribution for the full trajectory can be written as

$$p(\bar{\mathbf{x}}) = p(\mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^T) = p(\mathbf{x}^0) \prod_{t=0}^{T-1} p(\mathbf{x}^{t+1} | \mathbf{x}^t). \tag{3.4}$$

In the class of models considered in this work, the time evolution is determined by interactions among the variables, which are encoded in a graph $\mathscr{G} = (V,E)$. Here $V = 1,2,\dots,N$ are the vertices, each one associated to a variable, and $E \subset V \times V$ are

[1] Here we are not taking into account the complexity of variable $\mathbf{x}$ which could in principle take a prohibitively large number of values.

edges connecting pairs of vertices. Crucially, in this work we will consider diluted, in computer science lingo *sparse*, graphs, i.e. where the number of neighbors for each node is small enough to be considered a small constant with respect to the total number of variables. The reason for this choice, which will become clear in the following, boils down to computational tractability. Taking into account the structure of the graph, transition rules take the following form

$$p(\mathbf{x}^{t+1}|\mathbf{x}^t) = \prod_{i=1}^{N} w_i^{t+1}(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) \tag{3.5}$$

where $\partial i = \{j \in V : (ij) \in E\}$ is the set of neighbors of vertex i in the graph $\mathscr{G}$. The functions $\{w_i^{t+1}\}_{i,t}$ are normalized with respect to x_i^{t+1}, hence the conditional probability notation.

The factorization in (3.5) implies that, given the state of the system at time t, variables at time $t+1$ are statistically independent from one another. In other words, (3.5) provides a rule to update the state of all variables simultaneously, what is commonly known as a *parallel* update rule. It is important to notice that there are case where instead a *sequential* update is relevant. See e.g. the discussion in appendix B.1.1.

We further ask that the probability for the initial state to be factorized

$$p(\mathbf{x}^0) = \prod_{i=1}^{N} w_i^0(x_i^0). \tag{3.6}$$

All in all, the class of processes studied in this work are of the form

$$p(\overline{\mathbf{x}}) = \prod_{i=1}^{N} \left[w(x_i^0) \prod_{t=0}^{T-1} w_i^{t+1}(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) \right]. \tag{3.7}$$

3.1.1 Examples of Markov processes on graphs

We will mostly be concerned with two instances from the class of processes just described: the Susceptible-Infectious-Susceptible (SIS) model of epidemic spreading (and some of its variants), and parallel Glauber dynamics for classical binary spin systems.

SIS epidemiological model

A family of epidemiological models which were successful despite their simplicity assigns a discrete variable to each individual in a population, according to their state of infection to a given disease. One of the simplest is the SIS model, where the state of each individual at a given time is $x_i^t \in \{S, I\}$, i.e. they are either healthy (S) or not (I). Each individual starts with a probability γ_i of being infectious at time zero. Then at each time step a susceptible node i can be infected by each of its infectious neighbors $j \in \partial i$ with probability λ_{ji}, and an infectious node can recover with probability ρ_i. Optionally, one can also introduce an auto-infection probability α_i, which has no physical meaning but can prove useful when dealing with some computational issues. Mathematically, following the notation from earlier in this section we have:

$$w_i^0(x_i^0) = \gamma, \quad \forall i \tag{3.8}$$

and

$$w_i^{t+1}(x_i^{t+1} = I | \mathbf{x}_{\partial i}^t, x_i^t) = \begin{cases} 1 - (1-\alpha_i) \prod_{j \in \partial i} (1-\lambda_{ji})^{\delta(x_j^t, I)} & \text{if } x_i^t = I \\ 1 - \rho_i & \text{if } x_i^t = S. \end{cases} \tag{3.9}$$

Variants of SIS such as SIRS (Susceptible-Infectious-Recovered-Susceptible) can be easily obtained by introducing further states and parameters for the corresponding transitions.

Parallel Glauber dynamics

Glauber dynamics [54] is a continuous-time stochastic process for binary $\{\pm 1\}$ variables. It is a kinetic extension to the Ising model of magnetism. Here we consider a variation of the original formulation where time is discretized and variables are updated synchronously, hence the name parallel. Parameters of the model are: inverse temperature $\beta \in \mathbb{R}$, pair-wise coupling strengths $\{J_{ij}\}_{(ij) \in E}$, $J_{ij} = J_{ji} \in \mathbb{R}$ and external magnetic fields $\{h_i\}_{i \in V}$, $h_i \in \mathbb{R}$. The Markov transition rules are given by

$$w_i^{t+1}(\sigma_i^{t+1} | \sigma_{\partial i}^t) = \frac{e^{\beta \sigma_i^{t+1} \left(\sum_{j \in \partial i} J_{ij} \sigma_j^t + h_i \right)}}{2 \cosh \left[\beta \left(\sum_{j \in \partial i} J_{ij} \sigma_j^t + h_i \right) \right]}. \tag{3.10}$$

It is worth noting that, while continuous-time Glauber dynamics, as well as the discrete-time version with asynchronous update, converge to an equilibrium distribution

$$p(\sigma) \propto e^{\beta\left[\sum_{(ij)\in E} J_{ij}\sigma_i\sigma_j + \sum_{i\in V} h_i\sigma_i\right]} \tag{3.11}$$

which is the Boltzmann distribution of an Ising model, this is not true for the parallel update rule. However, some equivalence can be restored at the level of single-site and pair observables. More details on the links between the sequential and parallel rules of Glauber dynamics can be found in appendix B.1.1.

Other models

Other instances of Markov dynamics on graphs include: bootstrap percolation (which can be shown to be equivalent to Glauber dynamics [55]), kinetically constrained systems used to model glassy materials [56], exclusion processes in biology [57], the voter model of opinion dynamics [58], linear threshold and cascade models [59, 60].

3.1.2 Discrete versus continuous time

Models such as SIS are often introduced in a continuous-time framework as being governed by ODEs [61, 62]. While continuous-time processes can exhibit substantially different behavior from discrete ones, in many cases one can make contact between the two. After all, any modern ODE-based strategy will eventually rely on a numerical solver applying some time discretization. The continuous SIS model is often presented in terms of a master equation for variables $x_i(t) \in \{0,1\}$ where $0 \equiv S, 1 \equiv I$

$$\frac{d}{dt}x_i(t) = -\tilde{\rho}_i x_i(t) + (1 - x_i(t)) \sum_{j\in\partial i} \tilde{\lambda}_{ji} x_j(t). \tag{3.12}$$

where $\tilde{\rho}_i, \tilde{\lambda}_{ij}$ have units of time^{-1} and are instantaneous rates of recovery and transmission, respectively.

In models such as this one, where the probability of an event (infection, recovery) happening in a time unit goes to zero as the length of the unit shrinks, the connection with the discrete-time variant can be made by picking $\rho_i = \tilde{\rho}_i \Delta t, \lambda_{ij} = \tilde{\lambda}_{ij}\Delta t$ in (3.9) and sending $\Delta t \to 0$. In practice, dealing with the discrete-time version with small Δt is often enough to well approximate continuous-time behavior (see e.g. fig. 3.7).

More in general, continuous-time behavior can be simulated by introducing a further transition which does nothing, i.e. the variable doesn't change its state, with a certain probability p_0. In this case, the transition takes the form

$$\tilde{w}_i^{t+1}(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) = (1-p_0)w_i^{t+1}(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) + p_0\delta(x_i^{t+1}, x_i^t). \tag{3.13}$$

It is worth noting that a small value of p_0 implies that it takes a large total number of time-steps T in order to describe the same physical time interval. This can become computationally prohibitive.

3.1.3 Mathematical tractability

A Markov process of the form (3.7) is fairly convenient to treat computationally, as several basic operations can be performed efficiently:

- **Normalization**: the distribution is already normalized.
- **Sampling**: to sample a trajectory, one first samples an initial state for each variable independently from $w_i^0(x_i^0)$. Then, given the state at time t, each variable at time $t+1$ is sampled independently from $w_i^{t+1}(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t)$. The computational cost is manifestly linear in T. It is exponential in the degree (number of neighbors) of each node, however this is not a problem if the graph is sparse.
- **Observables**: the ubiquitous operation of computing the average of some trajectory-dependent observable $\mathscr{A}$ involves summing over the whole trajectory space

  $$A = \langle \mathscr{A} \rangle = \sum_{\overline{\mathbf{x}}} p(\overline{\mathbf{x}})\mathscr{A}(\overline{\mathbf{x}}), \tag{3.14}$$

  an operation whose cost scales exponentially both with the system size N and the time horizon T. However, good approximations can often be obtained by a Monte Carlo approach: drawing M samples from p, then replacing the extensive average with statistics over the samples

  $$\hat{A} = \frac{1}{M}\sum_{\mu=1}^{M} A(\overline{\mathbf{x}}^{(\mu)}), \quad \mathbf{x}^{(\mu)} \sim p. \tag{3.15}$$

As a notable example of observable average, one is often interested in computing marginal distributions, i.e. the distribution for the state of a single node i at a single time-step t, $\forall i,t$

$$p_i^t(x) = \sum_{\overline{\mathbf{x}}} p(\overline{\mathbf{x}})\delta(x, x_i^t). \tag{3.16}$$

Alternatively to the Monte Carlo approach for estimating observables, several analytical approximation techniques have been proposed, especially in the statistical physics community, mostly relying on some assumption of site-based and/or time-based independence. We review them in section 3.3.1.

3.2 Reweighted dynamics

An accurate description of the type of stochastic dynamics described above can answer questions like "What is the threshold value of transmission probability for which an epidemic becomes endemic in a population?" or "What will be the long-time behavior of Glauber dynamics at low temperature?". But what about questions like "What is the probability that a certain individual was the zero-patient of a pandemic given the results of tests performed throughout the population at different moments in time?" or the more abstract "What is the probability of a very rare trajectory which will never be seen in practice and whose contribution to averages is negligible?". These and other tasks belong to the class of inverse problems, which are the main scope of this chapter. Provided that one accepts to follow a Bayesian prescription, the questions above can be formulated as the computation of a posterior probability distribution

$$p(\overline{\mathbf{x}}|\mathbf{O}) = \frac{1}{p(\mathbf{O})} p(\overline{\mathbf{x}}) p(\mathbf{O}|\overline{\mathbf{x}}) \tag{3.17}$$

where we are denoting by $\mathbf{O}$ a set of observations performed on the system during the time evolution. $p(\overline{\mathbf{x}})$ is the prior over the trajectories, which we take of the form (3.7). $p(\mathbf{O}|\overline{\mathbf{x}})$ is the likelihood, i.e. the probability of observing O for a realization $\overline{\mathbf{x}}$ of the dynamics.

We will make the assumption that the likelihood factorizes over sites and time-steps

$$p(\mathbf{O}|\overline{\mathbf{x}}) = \prod_{i,t} p(O_i^t|x_i^t), \tag{3.18}$$

and provide examples below for why this is reasonable.

The posterior probability finally takes the form of a *reweighted* version of the unconditioned process (the prior) by means of reweighting factors $\phi_i^t(x_i^t) \equiv p(O_i^t|x_i^t)$:

$$q(\overline{\mathbf{x}}) = \frac{1}{Z}\prod_{i=1}^{N}\left[w(x_i^0)\prod_{t=0}^{T-1} w(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t)\phi_i^{t+1}(x_i^{t+1})\right]. \tag{3.19}$$

where we dropped the dependence on the observation set, which is always given. Z is the appropriate, observation-dependent, normalization constant corresponding to the marginal likelihood $p(\mathbf{O})$.

In other words, the posterior is a new process which is obtained from the prior by multiplying the probability of each trajectory according to the observations encoded in the reweighting factors. Furthermore, the result must be properly normalized. This seemingly harmless operation is in fact disguising a consistent amount of computational intricacy. Before diving into that, let us mention some phenomena that are well described by reweighted Markov dynamics.

3.2.1 Examples of reweighted dynamics

Epidemic reconstruction from partial observations

Suppose to have a model of the form (3.7) for how an infectious disease spreads over a known network of contacts, together with data coming from the results of tests performed over a generic subset of the population. Data is in the form $\mathbf{O} = \{O_i^t\}$ where $O_i^t \in \{S, I\}$ is the result of the test performed on individual i at time t. Under the reasonable assumption that test results are statistically independent, the likelihood takes the form

$$p(\mathbf{O}|\overline{\mathbf{x}}) = \prod_{i,t} \alpha^{\delta(O_i^t, x_i^t)}(1-\alpha)^{1-\delta(O_i^t, x_i^t)} \tag{3.20}$$

where $\alpha \in [0,1]$ is the accuracy of tests, i.e. the probability of giving the correct result.

In the limit $\alpha = 1$ of perfectly reliable tests one gets $p(\mathbf{O}|\overline{\mathbf{x}}) = \delta(O_i^t, x_i^t)$. Notice that this amounts to giving zero probability weight to any trajectory which is not compliant with all of the observation, and renormalizing the weight for the trajectories which are compatible with the observations.

Of particular interest in this case are the marginals at time zero $\{p_i^0(x_i^0)\}_i$, as they reveal with which probability an individual was the seed, or zero-patient, of the epidemic [63, 64].

Large deviations: zooming in on rare trajectories

A typical (pun intended) scenario where reweighted stochastic dynamics appear naturally is the analysis of rare events.

When dealing with high-dimensional probability distributions, most of the probability weight ends up being assigned to a small fraction of *typical* instances [65]. While such typical realization of a process contribute the most to the computation of averages, there are cases where one is interested in studying the properties of *atypical* events whose probability is exponentially small in the dimensionality of the problem. In analogy with traditional approaches in equilibrium statistical mechanics [66, 67], the task can be approached mathematically by introducing a tilted (reweighted) version of the original distribution. We will display such rather general idea with an example from Glauber dynamics.

Consider the parallel Glauber dynamics (3.10) for an underlying Ising system of binary $\{\pm 1\}$ variables living on a diluted graph of size $N \gg 1$, evolving from a certain initial condition for T time-steps

$$W(\overline{\sigma}) = \prod_{i=1}^{N} \left[w(\sigma_i^0) \prod_{t=0}^{T-1} w(\sigma_i^{t+1} | \sigma_{\partial i}^t) \right] \tag{3.21}$$

where $w(\sigma_i^{t+1}|\sigma_{\partial i}^t) \propto e^{J\sigma_i^{t+1}\sum_{j\in\partial i}\sigma_j^t}$. Suppose one is interested in computing statistics of the magnetization at final time $\frac{1}{N}\sum_{i=1}^N \sigma_i^T$. This can be done by introducing an external field h at final time in the form of reweighting factors $\prod_i \phi_i^T(\sigma_i^T) = \prod_i e^{h\sigma_i^T}$.

The full process then has the form

$$q(\overline{\sigma}) = \frac{1}{Z(h)} W(\overline{\sigma}) e^{h \sum_{i=1}^{N} \sigma_i^T} \tag{3.22}$$

where we stressed the dependence of the normalization constant on the external field. By defining a field-dependent "free energy" one gets

$$f(h) = -\frac{1}{N} \log Z(h) \tag{3.23}$$

$$= -\frac{1}{N} \log \sum_{\overline{\sigma}} W(\overline{\sigma}) \prod_{i=1}^{N} e^{h \sigma_i^T}. \tag{3.24}$$

The next step is to re-organize the summation over the input space grouping trajectories by their magnetization at final time

$$f(h) = -\frac{1}{N} \log \int_{-1}^{1} dm \sum_{\overline{\sigma}: \sum_i \sigma_i^T = Nm} W(\overline{\sigma}) e^{h \sum_{i=1}^{N} \sigma_i^T} \tag{3.25}$$

$$= -\frac{1}{N} \log \int_{-1}^{1} dm e^{Nhm} \sum_{\overline{\sigma}: \sum_i \sigma_i^T = Nm} W(\overline{\sigma}) \tag{3.26}$$

$$= -\frac{1}{N} \log \int_{-1}^{1} dm e^{-N(g(m) - hm)} \tag{3.27}$$

where we defined

$$g(m) = -\frac{1}{N} \log \sum_{\overline{\sigma}: \sum_i \sigma_i^T = Nm} W(\overline{\sigma}) \tag{3.28}$$

In the large N limit we invoke Laplace's approximation to replace the integral over m by a maximization and get

$$f(h) = -\frac{1}{N} \log \max_m e^{-N(g(m) - hm)} \tag{3.29}$$

$$= -\frac{1}{N} \log e^{-N \min_m \{g(m) - hm\}} \tag{3.30}$$

$$= \min_m \{g(m) - hm\}. \tag{3.31}$$

In regions where $g(m)$ is convex, the Legendre transform (3.31) can be inverted to obtain a large deviation law for the probability $p(m) \sim e^{-Ng(m)}$ of observing the

system at final time with magnetization m. This, however, implies being able to compute $f(h)$, which as we will see can be a complicated task.

3.2.2 The computational challenge

Reweighted dynamics turn out to be quite more challenging than their free counterpart. Not only the Markov property ceases to hold, the resulting process is in fact not even causal anymore! Indeed, the reweighted process (3.19) loses the good properties listed in 3.1.3:

- **Normalization**: evaluating the distribution (3.19) at a point in trajectory space requires first computing the normalization constant

$$Z = \sum_{\bar{\mathbf{x}}} \prod_{i=1}^{N} \left[w(x_i^0) \prod_{t=0}^{T-1} w(x_i^{t+1} | \mathbf{x}_{\partial i}^t, x_i^t) \phi_i^{t+1}(x_i^{t+1}) \right] \tag{3.32}$$

  which costs $\mathcal{O}(e^{NT})$ operations.

- **Sampling**: the fact that the process is not Markov forbids any obvious strategy for efficient sampling.

- **Observables**: as in the non-reweighted case, computing averages is computationally expensive. The back-up strategy mentioned earlier of estimating them via a Monte Carlo approach is not viable either, since obtaining samples from q is difficult.

Let us again stress that, as in the non-reweighted case, the quantity one is most often interested in computing are the marginal distributions (3.16). And as in the non-reweighted case the task is more complicated when dealing with recurrent dynamics.

3.3 Related work

This section reviews statistical physics inspired method that have been used to estimate marginals of Markov dynamics on graphs, overcoming some of the issues mentioned above such as recurrent dynamics and reweighting. However, no approximation

strategy has, to the best of our knowledge, been able to address all of the limitations. This is what motivated us to work on an alternative method, Matrix Product Belief Propagation, which is the main content of this chapter.

3.3.1 Mean-field and message-passing methods

"Mean-field" refers to a broad class of methods, commonly developed within the statistical physics community, that aim to approximate the behavior of complex interacting systems. Although the definition varies, mean-field typically refers to an approximation in which a structured multivariate probability distribution is simplified through some form of factorization. In dynamical contexts, this often means assuming statistical independence between neighboring variables at a given time.

Pair-based mean-field [68, 69], Dynamic Message Passing (DMP) [70–72], and the Cavity Master Equation (CME) [73, 74] are fast, approximate methods, originally formulated as ODEs in continuous time for vectors of single-edge quantities (such as cavity magnetizations). DMP and CME are exact on acyclic graphs and non-recurrent models (such as SI or SIR), but they are only approximate on recurrent models. Crucially, none of the three is able to deal with reweighted processes. The n-step Dynamic Message Passing [75] assumes an n-Markov structure on messages, focusing mainly on $n = 1$. Its characteristics are essentially the same as DMP but apply to discrete time evolution, describing interactions at a distance n in time.

Different variations of the Cluster Variational Method (CVM) [76, 77] approximate dynamics by treating exactly correlations between variables that are close either in time or space. It is in principle possible to extend CVM to work with reweighted processes, however, to the best of our knowledge this has not been done.

The so-called dynamic cavity (DC) method [78] is an extension of the cavity approach to dynamical problems. It assumes a tree-like factorization and computes messages which are now defined over joint trajectories in time of pairs of neighboring nodes. Messages are computed sequentially in time, requiring as many steps as the epochs in the dynamics. Since it is based on a cavity assumption, DC is exact on graphs with no cycles, and can be declined in an ensemble version in order to work directly in the thermodynamic limit (as explained in 1.2.4 for the static case). On the other hand, it works with distributions over single-site or pair trajectories,

which require as much as $\mathcal{O}(e^T)$ parameters, which is often impractical. For non-recurrent models, where variables can only progress sequentially through a finite set of q states without returning to previously visited states, the space of trajectories grows polynomially with the time horizon. For instance, with $q = 4$, a trajectory $(1,2,2,2,3,4,4)$ over epochs $t = 0,\ldots,6$ can be represented by the tuple $(1,4,5)$, indicating the epochs when the variable transitions to the next state. Examples of non-recurrent models include SI, SIR, and SEIR compartmental models in epidemiology, where individuals transition between states such as Susceptible, Exposed, Infective, and Recovered. Although non-recurrent models are commonly used, more realistic descriptions often require accounting for re-infections, in which case recurrent models like SIS and SIRS are needed.

In recent work [79, 80], a variant of DC was applied to Glauber dynamics on a random regular graph with degree 3. Instead of representing messages exactly, this method approximates the trajectory of single variables and pairs of variables using the matrix product ansatz from quantum mechanics [81–83]. This type of approximation, also known as tensor train in mathematics, has been used before also in out-of-equilibrium statistical physics [57, 84], and machine learning [85, 86], and will appear again later in this work. While promising, the matrix product dynamic cavity approach has two significant limitations: first, it is computationally expensive, as updating a node of degree z requires operations of order M^{2z-1}, where M is the matrix size, or bond dimension [80]. Second, the method is designed to analyze free dynamics without reweighting.

Inverse problems that involve reweighted dynamics have been explored in [59, 87], though primarily for non-recurrent models. Their approach, called Pair Trajectory Belief Propagation, is closely related to dynamic cavity methods. Large deviations were studied in [88] using perturbation theory in the specific case of Glauber dynamics on a chain. Recently, inference in epidemic processes has been tackled by fitting a non-reweighted process to the reweighted one [89], however this method requires sampling to estimate gradients in a variational inference framework. Finally, in the spirit of the cavity approximation, an expansion for small infection rates has been proposed in [90]. Although possible in principle, this method has not been generalized beyond the SIRS model of epidemics.

Table 3.1 summarizes the features of a pool of the discussed methods. The features considered include:

	Reweighting	Recurrent models	Autocorrelations
BP for non-recurrent models	Y	N	Y
IBMF, DMP, CME	N	Y	N
Dynamic Cluster Variational	*	Y	Two-times only
Matrix Product Dynamic Cavity	N	Y	Y
Matrix Product Belief Propagation	Y	Y	Y

Table 3.1 Features of existing analytical methods for the description of stochastic dynamics on graphs. Y = yes, N = no. Asterisks indicate that while the method could, in principle, include the considered feature, this has not, to the best of our knowledge, been implemented. IBMF stands for Individual-Based Mean Field, DMP for Dynamic Message Passing, and CME for Cavity Master Equation. We did not include the perturbative approach [88] as it focuses on a very specific setting.

- Handling of reweighted dynamics
- Application to recurrent models
- Estimation of autocorrelations, i.e., correlations between a variable and itself at a later time

3.3.2 Importance sampling Monte Carlo

Although it is not viable to extract samples directly from the reweighted process, a Monte Carlo approach to the estimation of observables is still possible in the form of importance sampling. This was proposed in the context of epidemic source reconstruction in [63]. The idea of importance sampling is to draw samples from a simple distribution, then reweight them to reproduce, modulo some possible approximation, the desired statistics. In the case of reweighted dynamics, this amounts to draw M samples $\left\{\overline{\mathbf{x}}^{(\mu)}\right\}_{\mu}$ independently from the prior (3.7), then reweight them with their likelihood (the reweighting factors). For a generic observable

$\mathscr{A}$, this looks like

$$\hat{A} = \frac{\sum_{\mu=1}^{M} \mathscr{A}(\overline{\mathbf{x}}^{(\mu)}) \prod_{i,t} \phi_i^t \left((x_i^t)^{(\mu)} \right)}{\sum_{\mu=1}^{M} \prod_{i,t} \phi_i^t \left((x_i^t)^{(\mu)} \right)}. \tag{3.33}$$

As always when performing importance sampling on distribution whose normalization is unknown, both numerator and denominator are unbiased estimators of the corresponding quantities in (3.19). Their quotient, however, is not.

The main issue with this strategy is that, as N, T and the number of observations increase, it becomes very (exponentially) unlikely that a trajectory sampled from the prior is given large weight by all of th ϕ terms. In other words, trajectories typical of the prior process are given a very small weight in the posterior, with high probability. As a result, it takes an exceedingly large amount of samples for the prior process to realize a trajectory which is typical of the posterior and by so to contribute meaningfully to the computation of observables.

3.4 Matrix Product Belief Propagation

In this section we present Matrix Product Belief Propagation (MPBP), introduced in [47], a method to compute marginals of reweighted Markov dynamics on diluted graphs. MPBP is essentially a two-layer approximation. To address the complexity given by the interaction of the N degrees of freedom, we employ a cavity approximation inspired by the dynamic cavity [78] or equivalently the pair trajectory belief propagation [59] formalism. On top of that, to address the complexity in T coming from the (in general) recurrent character of the dynamics, we adopt the matrix product approximation of [79, 80].

3.4.1 A slightly more general framework

Before diving into the details of the method, let us mention that the ideas in the next pages are not limited to dynamics. In fact, the technique we propose, despite being motivated by Markov dynamics of the form (3.19), applies more in general. For completeness, let us recap here the mathematical context in full detail.

Consider a graph $\mathscr{G} = (V,E)$ with $V = \{1,2,\ldots,N\}, E \subset V \times V$. Consider a multivariate random variable $\overline{\mathbf{x}} = \{x_i^t\}_{\substack{i\in V \\ t\in\{0,1,2,\ldots,T\}}}$ where $x_i^t \in \{1,2,\ldots,q_i\}$, i.e. variables are discrete and do not need to have domains of the same size. We will be interested in probability distributions over such a variable of the form

$$q(\overline{\mathbf{x}}) = \frac{1}{Z}\prod_{t=0}^{T-1}\prod_{i=1}^{N} f_i^{t+1}\left(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t\right). \tag{3.34}$$

Connection with the framework of dynamics is readily done by identifying $f_i^{t+1}(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t) = w(x_i^0)^{\delta(t,0)} w(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t)\phi_i^{t+1}(x_i^{t+1})$. However, the "time" axis doesn't need to be interpreted as such: nothing prevents us to use it to denote a further spatial dimension, as long as interactions in the systems are well described by a factorization like the one in (3.34). Exploration in this direction, as well as establishing connections with previous work that goes as back as the forties [91] is beyond the scope of this manuscript and is left as future investigation.

Here, the goal is to describe the distribution (3.34) as closely as possible, albeit in an approximate manner. This can be done by estimating the following quantities:

- Marginals
$$q_i^t(x) = \sum_{\overline{\mathbf{x}}} q(\overline{\mathbf{x}})\delta(x, x_i^t), \quad \forall i,t. \tag{3.35}$$

- The partition function, or normalization constant
$$Z = \sum_{\overline{\mathbf{x}}}\prod_{t=0}^{T-1}\prod_{i=1}^{N} f_i^{t+1}\left(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t\right). \tag{3.36}$$

- Pairwise marginals for neighboring variables
$$q_{ij}^t(x,y) = \sum_{\overline{\mathbf{x}}} q(\overline{\mathbf{x}})\delta(x, x_i^t)\delta(y, x_j^t), \quad \forall t, (ij) \in E. \tag{3.37}$$

- Autocorrelations
$$c_i^{t,s} = \sum_{\overline{\mathbf{x}}} q(\overline{\mathbf{x}}) f(x_i^t, x_i^s), \quad \forall i,t,s \tag{3.38}$$
for some two-time observable f.

In the following we will be mostly concerned with the computation of marginals, although all of the other quantities can be estimated as well.

3.4.2 Belief Propagation equations

Once presented with equation (3.34) and the goal of computing marginals, what we propose is to take a path such as the one in [59]. One would like to apply the cavity approximation which, however, was originally formulated for static problems. The solution is to first group together the trajectory of each node i of the graph into a macro-variable $\overline{x}_i = \{x_i^t\}_{t=0,1,\dots,T}$. At this point the full distribution looks like a static distribution for the new trajectory-wide variables

$$q(\overline{\mathbf{x}}) \propto \prod_{i=1}^{N} f_i\left(\overline{x}_i, \overline{\mathbf{x}}_{\partial i}\right) \tag{3.39}$$

upon identifying $f_i\left(\overline{x}_i, \overline{\mathbf{x}}_{\partial i}\right) = \prod_{t=0}^{T-1} f_i^{t+1}\left(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t\right)$. It is easy to see that the new variables interact, under (3.39), on a certain factor graph. It is therefore possible to apply, at least formally, the cavity (BP) approximation to it.

The factor graph associated with (3.39) would present many small loops due to the presence of both $\overline{x}_i$ and $\overline{x}_j$ in factors f_i and f_j, even in cases where the original graph is acyclic. Therefore, we work on the so-called dual factor graph where variables are pair of trajectories $(\overline{x}_i, \overline{x}_j)$ living on the edges of the original graph. The procedure in sketched in fig. 3.1 for a toy graph with three nodes. For more details about this step we refer the reader to [59, fig. 3, eqns 8,9].

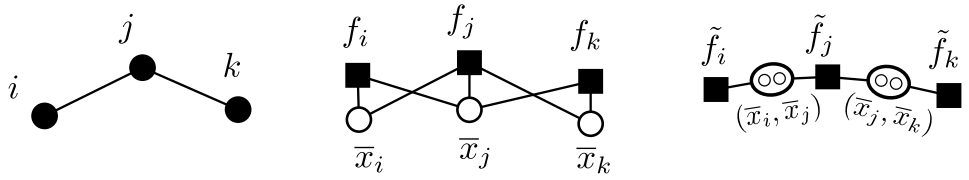


Fig. 3.1 Construction of the dual factor graph for a toy dynamics with three variables interacting on a tree. Left: the original graph of pair-wise interactions. Center: the factor graph resulting from representing the distribution (3.39). Right: the dual factor graph with the new pair variables. Each factor $\tilde{f}_i$ should also contain additional terms ensuring that trajectory $\overline{x}_i$ takes the same values on all edges.

The BP equations on the dual factor graph read

$$m_{i\to j}(\bar{x}_i,\bar{x}_j) = \frac{1}{z_{i\to j}} \sum_{\bar{x}_{\partial i\setminus j}} \prod_{t=0}^{T-1} f_i^{t+1}(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t) \prod_{k\in\partial i\setminus j} m_{k\to i}(\bar{x}_k,\bar{x}_i). \tag{3.40}$$

Since the number of joint trajectories $(\bar{x}_i,\bar{x}_j)$ is exponentially large in T, an exact representation of the messages is in general computationally unfeasible. Here, similarly to [79], we parametrize messages in terms of matrix product states [83, 82, 81], also known as tensor trains in the mathematical literature [92]. Following the jargon of tensor networks, in the rest of the paper we will refer to the size of the matrices as bond dimension.

3.4.3 The matrix product ansatz

Following the ideas in [79], messages are parametrized in terms of matrix products

$$m_{i\to j}(\bar{x}_i,\bar{x}_j) \propto \prod_{t=0}^{T} A_{i\to j}^t(x_i^t, x_j^t) \tag{3.41}$$

where, for any (x_i^t, x_j^t), $A_{i\to j}^t(x_i^t, x_j^t)$ is a real-valued matrix. We set A^0 to have one row and A^T to have one column, so that the whole product gives a scalar. Since notation at this point usually creates some confusion in the reader, let us re-write (3.41) with the indices for matrix multiplication spelled out:

$$m_{i\to j}(\bar{x}_i,\bar{x}_j) \propto \sum_{a^1,a^2,\dots,a^T} \prod_{t=0}^{T} \left[A_{i\to j}^t(x_i^t, x_j^t)\right]_{a^t,a^{t+1}} \tag{3.42}$$

with $a^0 \equiv a^{T+1} \equiv 1$ to ensure that the result is of dimension 1×1.

Plugging the ansatz (3.41) into the RHS of the BP equation (3.40) gives

$$m_{i\to j}(\bar{x}_i,\bar{x}_j) \propto \prod_{t=0}^{T} B_{i\to j}^t(x_i^{t+1}, x_i^t, x_j^t) \tag{3.43}$$

where, at fixed $i, j, t, x_i^{t+1}, x_i^t, x_j^t$, $B^t_{i\to j}(x_i^{t+1}, x_i^t, x_j^t)$ is a matrix

$$B^t_{i\to j}(x_i^{t+1}, x_i^t, x_j^t) = \sum_{\{x_k^t\}_{k\in\partial i\setminus j}} f_i^{t+1}(x_i^{t+1}, \mathbf{x}^t_{\partial i}, x_i^t) \left[\bigotimes_{k\in\partial i\setminus j} A^t_{k\to i}(x_k^t, x_i^t) \right] \tag{3.44}$$

with $\otimes$ the Kroenecker product $(A\otimes B)_{(i,j),(k,l)} = A_{i,j}B_{k,l}$.

The previous equation is useful as a compact representation. To be fully precise, one should specify the values for the B matrices also in the corner cases $t = 0, T$. These, with matrix indices made explicit, read:

$$\left[B^0_{i\to j}(x_i^1, x_i^0, x_j^0)\right]_{\{a_k^1\}_{k\in\partial i\setminus j}} = \sum_{\{x_k^0\}_{k\in\partial i\setminus j}} f_i^1(x_i^1|\mathbf{x}^0_{\partial i}, x_i^0) \prod_{k\in\partial i\setminus j} \left[A^0_{k\to i}(x_k^0, x_i^0)\right]_{a_k^1} \tag{3.45}$$

$$\left[B^t_{i\to j}(x_i^{t+1}, x_i^t, x_j^t)\right]_{\{a_t^k, a_{t+1}^k\}_{k\in\partial i\setminus j}} = \sum_{\{x_k^t\}_{k\in\partial i\setminus j}} f_i^{t+1}(x_i^{t+1}, \mathbf{x}^t_{\partial i}, x_i^t) \prod_{k\in\partial i\setminus j} \left[A^t_{k\to i}(x_k^t, x_i^t)\right]_{a_k^t, a_k^{t+1}}$$
$$\forall t \in \{1, \dots, T-1\} \tag{3.46}$$

$$\left[B^T_{i\to j}(x_i^T, x_j^T)\right]_{\{a_k^T\}_{k\in\partial i\setminus j}} = \sum_{\{x_k^T\}_{k\in\partial i\setminus j}} \prod_{k\in\partial i\setminus j} \left[A^T_{k\to i}(x_k^T, x_i^T)\right]_{a_k^T}. \tag{3.47}$$

In order to close the BP equations under the ansatz (3.41), some more work is required. First, the B matrices defined above depend on three variables instead of only x_i^t, x_j^t. Second, and more importantly, the Kroenecker product in (3.44) makes it explicit that the size of the matrices in the outgoing message is larger compared to the incoming ones. For a node with degree $|\partial i| = z$, the Kroenecker product of $z-1$ incoming matrices of bond dimension M results in a B matrix of bond dimension M^{z-1}. Clearly if this procedure were to be iterated, the size of the matrices would quickly become unmanageable.

The following shows how one can take care of the first issue by means of successive singular value decompositions (SVDs). Such a procedure also serves as a preliminary step for a second series of SVDs, aimed at reducing the size of the matrices. These ideas, already discussed in [79] are reminiscent of the density matrix renormalization group (DMRG) algorithm from many-body quantum physics [93].

Intermezzo 1: the singular value decomposition

The SVD decomposes a complex-valued matrix A as $A = U\Lambda V^\dagger$, or, with indices spelled out

$$A_{ij} = \sum_{k,l=1}^{M} U_{ik}\Lambda_{kl}V^*_{jl} \tag{3.48}$$

where M is the rank of A, $*$ denotes complex-conjugation and † complex-conjugation and matrix transpose. Here we will be working with real-valued matrices but the dagger notation helps preventing confusion with the T symbol for the time horizon.

The matrices U, Λ, V enjoy some special properties. Λ is zero everywhere except on the diagonal where it contains the so-called singular values. Without loss of generality, one can pick the order of rows and columns such that the singular values are sorted as $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_M \geq 0$. Singular values are real and non-negative by construction because they correspond to the eigenvalues of $AA^\dagger$, in turn equal to those of $A^\dagger A$. Furthermore, the matrix U is said to be left-orthogonal because it holds $U^\dagger U = \mathbb{1}$. Similarly, V is right-orthogonal, as $VV^\dagger = \mathbb{1}$.

The SVD can be used to produce low-rank approximations to a given matrix A which conveniently come with a rigorous estimate of the approximation error. A compressed version of A is constructed by keeping only the $M' < M$ largest singular values. Call $\tilde{\Lambda}$ the diagonal matrix which is equal to Λ except it has zeros instead of the last $M - M'$ elements. Then the truncated version of A is $\tilde{A} = U\tilde{\Lambda}V^\dagger$. As matrix elements:

$$\tilde{A}_{ij} \coloneqq \sum_{k=1}^{M'} U_{ik}\lambda_k V_{jk}. \tag{3.49}$$

The truncation error, as measured by the Frobenius norm[2] is given by

$$\begin{aligned}
\varepsilon =& \|A-\widetilde{A}\|_F^2 \\
=& \mathrm{Tr}\left[\left(\mathrm{A}-\widetilde{\mathrm{A}}\right)^\dagger\left(\mathrm{A}-\widetilde{\mathrm{A}}\right)\right] \\
=& \mathrm{Tr}\left[\mathrm{A}^\dagger\mathrm{A}-\mathrm{A}^\dagger\tilde{\mathrm{A}}-\tilde{\mathrm{A}}^\dagger\mathrm{A}+\tilde{\mathrm{A}}^\dagger\tilde{\mathrm{A}}\right] \\
=& \mathrm{Tr}\left[\mathrm{V}\Lambda\mathrm{U}^\dagger\mathrm{U}\Lambda\mathrm{V}^\dagger-\mathrm{V}\Lambda\mathrm{U}^\dagger\mathrm{U}\tilde{\Lambda}\mathrm{V}^\dagger-\mathrm{V}\tilde{\Lambda}\mathrm{U}^\dagger\mathrm{U}\Lambda\mathrm{V}^\dagger+\mathrm{V}\tilde{\Lambda}\mathrm{U}^\dagger\mathrm{U}\tilde{\Lambda}\mathrm{V}^\dagger\right] \\
=& \mathrm{Tr}\left[\Lambda^2\right]-\mathrm{Tr}\left[\Lambda\tilde{\Lambda}\right]-\mathrm{Tr}\left[\tilde{\Lambda}\Lambda\right]+\mathrm{Tr}\left[\tilde{\Lambda}^2\right] \\
=& \sum_{k=1}^{M}\lambda_k^2-\sum_{k=1}^{M'}\lambda_k^2-\cancel{\sum_{k=1}^{M'}\lambda_k^2}+\cancel{\sum_{k=1}^{M'}\lambda_k^2} \\
=& \sum_{k=M'+1}^{M}\lambda_k^2
\end{aligned} \tag{3.50}$$

i.e. the sum of the squares of the discarded singular values.

To understand the usefulness of this strategy, notice that, once the smallest $M-M'$ singular values have been truncated to zero, one might as well delete the corresponding $M-M'$ columns of U and of V, as they would be multiplied by zero anyway. In practice, one then stores the truncated $\tilde{U},\tilde{\Lambda},\tilde{V}$ to take advantage of the reduced dimensionality.

Intermezzo 2: discrete variables as tensor indices

A crucial observation in understanding the following steps is this: a function of discrete variables $f(x_1,x_2,x_3,x_4)$ is, as a piece of data, equivalent to a multidimensional array (tensor[3]) F_{x_1,x_2,x_3,x_4}. The connection is particularly evident when the function f is stored on a computer by enumerating its value at all inputs. It turns out that both DMRG and the matrix product dynamic cavity method make heavy use of the fact that tensor indices and dependence on discrete variables can be interchanged.

Consider the following problem: given a matrix whose entries depend on two discrete variables $A(x,y)$, the goal is two factorize it as the product of two matrices, each depending on one of the two variables. In other words, one looks for $B(x),C(y)$

[2]The Frobenius norm of a matrix M is $\|M\|_F=\sqrt{\mathrm{Tr}(\mathrm{M}\mathrm{M}^\dagger)}=\sqrt{\sum_{ij}M_{ij}^2}$.

[3]We will not speak about tensors in the differential geometric sense in this work. Whenever the reader encounters a tensor, they should always think "multidimensional array"

such that

$$[A(x,y)]_{ij} = \sum_k [B(x)]_{ik} [C(y)]_{kj}. \tag{3.51}$$

This is easily done by first reshaping the data in A into a matrix $\hat{A}$ whose row indices are the concatenation of the row index of A, and the values of variable x. Similarly for the columns. Explicitly:

$$\hat{A}_{(i,x),(j,y)} = [A(x,y)]_{ij}. \tag{3.52}$$

Then, a SVD is performed on $\hat{A}$

$$\hat{A}_{(i,x),(j,y)} \overset{\text{SVD}}{=} \sum_k \hat{B}_{(i,x),k} \hat{C}_{k,(j,y)}. \tag{3.53}$$

Finally, the inverse reshaping operation is performed on $\hat{B}, \hat{C}$ to get

$$\begin{cases} [B(x)]_{ik} \coloneqq \hat{B}_{(i,x),k} \\ [C(y)]_{kj} \coloneqq C_{k,(j,y)}. \end{cases} \tag{3.54}$$

One must pay a price for such a factorization, which lies in the range of index k. If, for simplicity, A is $N \times N$ and x, y take one of q values, the number of columns of B (equivalently, rows of C) is $M \leq qN$.

SVD sweeps

We have left the BP update with messages in the form of B matrices (3.44). Now a first sweep of SVDs is done from left to right $t = 0, 1, 2, \ldots, T-1$ by performing a decomposition

$$B^t_{i\to j}(x_i^{t+1}, x_i^t, x_j^t) \overset{\text{SVD}}{=:} C^t_{i\to j}(x_i^t, x_j^t) \Lambda^t \left[V^t(x_i^{t+1})\right]^\dagger \tag{3.55}$$

where, following the procedure described above, x_i^{t+1} is incorporated as column index, x_i^t, x_j^t as column indices.

At this point, the C^t matrix is in the sought form. The remaining factors are multiplied to the right to redefine matrix B^{t+1} as $\Lambda^t \left[V^t(x_i^{t+1})\right]^\dagger B^{t+1}_{i\to j}(x_i^{t+2}, x_i^{t+1}, x_j^{t+1})$. Then, the decomposition (3.55) is repeated on the new B^{t+1}, and so on. Once arrived

to the end of the tensor train at $t = T$, we have that $B^t_{i\to j}(x_i^{t+1}, x_i^t, x_j^t) \equiv B^t_{i\to j}(x_i^t, x_j^t) =: C^t_{i\to j}(x_i^t, x_j^t)$ (see (3.47)). Therefore, at the end of this first sweep, the message looks like

$$m_{i\to j}(\bar{x}_i, \bar{x}_j) = \prod_{t=0}^{T} C^t_{i\to j}(x_i^t, x_j^t) \tag{3.56}$$

where, thanks to the properties of the SVD, it holds that

$$\sum_{x_i^t x_j^t} \left[C^t_{i\to j}(x_i^t, x_j^t)\right]^\dagger C^t_{i\to j}(x_i^t, x_j^t) = \mathbb{1}, \quad \forall t \in \{0, 1, \ldots, T-1\}. \tag{3.57}$$

At this point the form (3.41) is recovered: the BP equations are closed under a matrix product ansatz for messages.

Again following [79], a second sweep of SVD is performed, this time with the goal of reducing the size of the matrices (bond dimension), while attempting to preserve most of the information. Going right to left $t = T, T-1, \ldots, 1$, incorporating x_i^t, x_j^t as column indices one has

$$C^t_{i\to j}(x_i^t, x_j^t) \overset{\text{SVD,trunc}}{=:} U^t \Lambda^t A^t_{i\to j}(x_i^t, x_j^t) \tag{3.58}$$

with truncation to a prescribed number of singular values. This step is followed as before by the redefinition of C^{t-1} as $C^{t-1}_{i\to j}(x_i^{t-1}, x_j^{t-1}) U^t \Lambda^t$, and so on until the leftmost end of the chain of matrices is reached.

At this point the reader might wonder why we do not directly truncate during the first sweep of SVDs. The reason is the following: imagine having two copies of the same tensor train and performing a low-rank approximation on a single matrix in one of the two. No matter how small the truncation error made on the single matrix, the distance between the two tensor trains as wholes can be arbitrary large, since even a small error will likely propagate considerably after multiplication with the other matrices. This is not the case, however, if all the other matrices are isometries: in that case the global error can be shown to be equivalent to the local one. Let us look at the generic t-th step of the second sweep, from right to left. The MPS is in the so-called mixed-canonical form [93]:

$$C^0 \cdots C^t A^{t+1} \cdots A^T \tag{3.59}$$

with $C^0 \cdots C^{t-1}$ left-orthogonal ($C^\dagger C = \mathbb{1}$) and $A^{t+1} \cdots A^T$ right-orthogonal ($AA^\dagger = \mathbb{1}$). C^t is the matrix about to be truncated according to (3.58). The global error, computed as the (square of the) 2-norm of the difference between the tensor train before and after replacing C^t with the truncated A^t is

$$\begin{aligned} \varepsilon^t =& \|C^0 \cdots C^t A^{t+1} \cdots A^T - C^0 \cdots \widetilde{C}^t A^{t+1} \cdots A^T\|_F^2 \\ =& \|C^t - A^t\|_F^2 \\ =& \sum_{k=M'+1}^{M} \lambda_k^2 \end{aligned} \tag{3.60}$$

where in the second line we used the semi-orthogonality of A and C matrices, and in the last line we used (3.50).

In summary, the total error for a full sweep of truncations is

$$\varepsilon \leq \sum_{t=0}^{T} \varepsilon^t. \tag{3.61}$$

How to pick the number of singular values to be retained is a design choice. It can be done adaptively as in [79] by selecting a threshold η on the maximum error one is willing to make for each matrix, and then computing M' such that $\varepsilon^t \leq \eta$. Since, at difference with the matrix product cavity method, MPBP is an iterative algorithm, we picked a different strategy. During the first iterations messages are still far from the fixed point, hence there is no particular advantage in being too accurate by picking a large bond dimension. Then, the strategy we adopted most often was to increase the bond dimension to get a finer result as the iterations proceed and messages are closed to a fixed point. Hybrid strategies are also possible: one can truncate according to a threshold but without exceeding a fixed maximum bond dimension.

SVDs constitute the bottleneck of the whole MPBP algorithm, even considering the improvements presented in section 3.6. Although we have not explored them directly, we mention that, whenever one is interested in truncating a matrix of size M to $M' \ll M$, it can be beneficial to employ some iterative techniques [94, 95] that never explicitly compute the singular values that will eventually be discarded.

3.4.4 Bond dimension

Excessive truncation of the matrix product can sometimes result in an ill-defined probability distribution, manifesting in negative values. This issue was encountered during our experiments and, as expected, increasing the bond dimension consistently resolved the problem. Figure 3.2 illustrates the effect of varying the bond dimension in two settings that will appear in the results section 3.7.

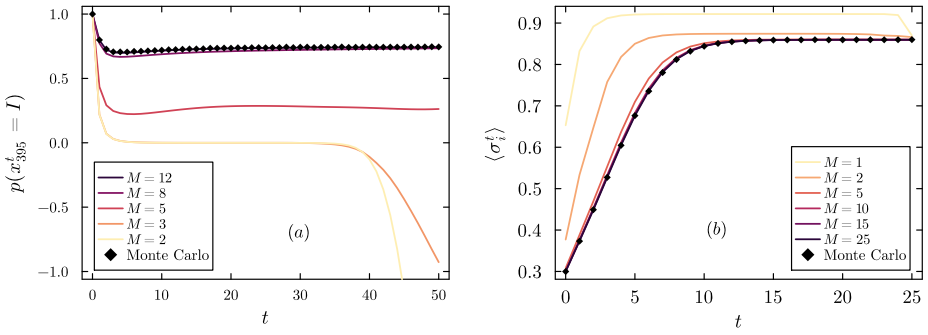


Fig. 3.2 Effect of varying the bond dimension M on the accuracy of the approximation. (a) SIS model on a tree, with the same settings as in Figure 3.4. A small bond dimension results in unreasonable outcomes. (b) Glauber dynamics on an infinite random regular graph of degree 8, with the same settings as Figure 3.10.

Considering the expressive power of the MPS ansatz, it is reasonable to expect that conservative truncation (i.e., using a larger bond dimension) will yield increasingly accurate approximations. In fact, it is known that tensor trains with arbitrarily large bond dimensions can represent any distribution exactly [92]. However, establishing a precise quantitative relationship between bond dimension and the complexity of the distribution being captured remains challenging and an open research question.

Based on discussions from quantum mechanics (e.g., [93, section 4.2.2]), it is plausible that strong, long-range correlations, temporal in our case, spatial in quantum systems, require larger matrices for accurate representation. Nevertheless, this is not the complete picture. A clear counterexample is the SI epidemic model, whose trajectories can be represented by MPS with finite bond dimensions[4], despite involving infinite-range correlations.

[4] see appendix B.2

3.4.5 Convergence

The BP equations are iterated until they converge to a fixed point. We chose to implement an asynchronous update scheme, as it generally exhibits better convergence properties compared to a synchronous one. However, both methods are largely interchangeable. As is common with BP, the procedure is highly parallelizable, with the synchronous approach offering even greater potential for parallel execution.

To assess convergence to a fixed point, we calculated the marginal distributions at all nodes and epochs, denoted as $b_i^t(x_i^t)$ (see (3.63)), and checked whether, for iteration *it* and the subsequent one:

$$\max_{i\in\{1,\dots,N\}} \max_{t\in\{0,\dots,T\}} \max_{x_i^t} \left| \left[b_i^t(x_i^t)\right]^{(it+1)} - \left[b_i^t(x_i^t)\right]^{(it)} \right| < \varepsilon \tag{3.62}$$

for some small threshold ε. A more stringent criterion can be employed by computing $\max_{(i,j)\in E} \sum_{\bar{x}_i,\bar{x}_j} \| m_{i\to j}(\bar{x}_i,\bar{x}_j)^{(it+1)} - m_{i\to j}(\bar{x}_i,\bar{x}_j)^{(it)} \|_F$. Both criteria yield similar results[5].

It is important to note that, in the case of non-reweighted dynamics, one can incrementally build the messages from time 0 to time T, as in matrix product dynamic cavity [79], without needing to iterate until convergence. Since each SVD sweep across the T matrices runs in linear time with respect to T, the total computational complexity of this scheme scales quadratically with T. In contrast, our method, which initializes messages for all T epochs and performs N_{iter} iterations, has a computational cost of $\mathcal{O}(N_{iter}T)$. The two approaches are essentially equivalent, as we typically observed that the number of iterations required for convergence is on the order of T.

3.4.6 Infinite graphs

Just as it was mentioned in section 1.2.4 for static problems, BP lends itself to computations in the thermodynamic limit. This is possible on models whose parameters are homogeneous enough: on random regular graphs with uniform properties, one message is enough (the same idea was exploited in message-passing schemes for dynamical processes [79, 73]).

[5] The interested reader can check https://github.com/stecrotti/MatrixProductBP.jl/blob/ae7bd61cda354bff29deea7bbe4887773e9e25e1/notebooks/convergence.ipynb

On systems with disorder, e.g. on Erdos-Renyi graphs, a population dynamics approach can be employed, analogous to the one explained for the static case in section 2.4.4.

3.5 Observables

This section shows how observables can be computed once the BP equations have reached a fixed point. In particular, we are interested in the distribution of single-variable and pair trajectories and marginalizations therein, and in the bethe free energy. The latter can be used to estimate the normalization constant for the joint distribution. As we saw earlier, it corresponds to the marginal likelihood in a Bayesian inference setting.

Single variable beliefs and pair beliefs, the BP estimates for single-variable and pair trajectory, respectively, are given by

$$b_i(\bar{x}_i) = \frac{1}{z_i} \sum_{\bar{x}_{\partial i}} \prod_{t=0}^{T-1} f_i^{t+1}(x_i^{t+1}, \mathbf{x}_{\partial i}^t, x_i^t) \times \prod_{k \in \partial i} m_{k \to i}(\bar{x}_k, \bar{x}_i). \tag{3.63}$$

and

$$b_{ij}(\bar{x}_i, \bar{x}_j) = \frac{1}{z_{ij}} m_{i \to j}(\bar{x}_i, \bar{x}_j) m_{j \to i}(\bar{x}_j, \bar{x}_i). \tag{3.64}$$

It is easy to see that both types of beliefs admit a matrix product representation which follows directly from the ansatz on messages. As a result, evaluating terms like single-time marginals $p_i^t(x_i^t)$ or autocorrelations $\langle x_i^t x_i^{t+\Delta t} \rangle$ reduces to computing the marginals of trajectory distributions that are parametrized by matrix product states (equiv. tensor trains). This can be done efficiently [92] and is reported in the following.

3.5.1 Efficient computations on matrix product distributions

Given a joint distribution in matrix product form

$$p(x^0, x^1, \ldots, x^T) = \frac{1}{Z} \sum_{a^1, a^2, \ldots, a^T} \left[A^0(x^0)\right]_{a^1} \left[A^1(x^1)\right]_{a^1, a^2} \cdots \left[A^{T-1}(x^{T-1})\right]_{a^{T-1}, a^T} \left[A^T(x^T)\right]_{a^T} \tag{3.65}$$

one can efficiently compute: normalization, marginals, autocorrelations. Notice that replacing the tuple (x_i^t, x_j^t) with a single variable x^t implies no loss of generality, since the joint state of two discrete variables can always be merged into a single, larger one.

Normalization and marginals Marginalizing at time t gives

$$p^t(x^t) = \sum_{\{x^s\}_{s\neq t}} p(x^0, x^1, \dots, x^T) \tag{3.66}$$

$$= \frac{1}{Z} \sum_{a^t, a^{t+1}} \left[L^{t-1}\right]_{a^t} \left[A^t(x^t)\right]_{a^t, a^{t+1}} \left[R^{t+1}\right]_{a^{t+1}} \tag{3.67}$$

where we defined partial normalizations from the left and from the right

$$\begin{cases} [L^t]_{a^{t+1}} \coloneqq & \sum_{a^1,\dots,a^t} \prod_{s=0}^{t} \sum_{x^s} [A^s(x^s)]_{a^s, a^{s+1}} = \sum_{a^t} \left[L^{t-1}\right]_{a^t} \sum_{x^t} [A^t(x^t)]_{a^t, a^{t+1}} \\ [R^t]_{a^t} \coloneqq & \sum_{a^{t+1},\dots,a^T} \prod_{s=t}^{T} \sum_{x^s} [A^s(x^s)]_{a^s, a^{s+1}} = \sum_{a^{t+1}} \sum_{x^t} [A^t(x^t)]_{a^t, a^{t+1}} \left[R^{t+1}\right]_{a^{t+1}} \end{cases} \tag{3.68}$$

which are computed recursively, starting from initial conditions

$$\begin{cases} \left[L^0\right]_{a^1} \coloneqq & \sum_{x^0} \left[A^0(x^0)\right]_{a^1} \\ \left[R^T\right]_{a^T} \coloneqq & \sum_{x^T} \left[A^T(x^T)\right]_{a^T} \end{cases}. \tag{3.69}$$

The normalization is given by

$$Z = \sum_{a^t} \left[L^t\right]_{a^{t+1}} \left[R^{t+1}\right]_{a^{t+1}} \quad \forall t \in 0, 1, \dots, T-1. \tag{3.70}$$

Autocorrelations

Further define "middle" partial normalizations from t to s ($t < s$ without loss of generality)

$$\left[M^{t,s}\right]_{a^{t+1},a^u} = \sum_{a^{t+2},\ldots,a^{u-1}} \prod_{u=t+1}^{s-1} \sum_{x_i^u,x_f^u} \left[A^u(x^u)\right]_{a^u,a^{u+1}} \tag{3.71}$$

$$= \sum_{a^{s-1}} \left[M^{t,s-1}\right]_{a^{t+1},a^{s-1}} \left(\sum_{x^{u-1}} \left[A^{s-1}(x^{s-1})\right]_{a^{s-1},a^s} \right) \tag{3.72}$$

with initial condition

$$\left[M^{t,t+1}\right]_{a,b} = \delta(a,b) \quad \forall t \in \{0,1,\ldots,T-1\}. \tag{3.73}$$

Finally, the joint distribution for the variable at two epochs is

$$p^{t,s}(x^t,x^s) = \sum_{\{x^u\}_{u\neq t,s}} p(x^0,x^1,\ldots,x^T) \tag{3.74}$$

$$= \frac{1}{Z} \sum_{\substack{a^t,a^{t+1} \\ a^s,a^{s+1}}} \left[L^{t-1}\right]_{a^t} \left[A^t(x^t)\right]_{a^t,a^{t+1}} \left[M^{t,s}\right]_{a^{t+1},a^s} \left[A^s(x^s)\right]_{a^s,a^{s+1}} \left[R^{s+1}\right]_{a^{s+1}}. \tag{3.75}$$

3.5.2 Bethe Free Energy

The BP formalism also gives access to the Bethe Free Energy, an approximation to (minus the logarithm of) the normalization of (3.34), which can be interpreted as the likelihood of the parameters of the dynamics (e.g. infection rates, temperature,...). In cases where such parameters are unknown, one can imagine inferring them via a maximum-likelihood procedure [96].

The Bethe Free Energy for a graphical model with pair-wise interactions is given by

$$F = -\sum_i \log z_i + \frac{1}{2} \sum_i \sum_{j\in\partial i} \log z_{ij} \tag{3.76}$$

where

$$z_i = \sum_{\bar{x}_i, \bar{\mathbf{x}}_{\partial i}} \prod_{t=0}^{T-1} f_i^{t+1}(x_i^{t+1} | \mathbf{x}_{\partial i}^t, x_i^t) \prod_{k \in \partial i} m_{k \to i}(\bar{x}_k, \bar{x}_i) \tag{3.77}$$

$$z_{ij} = \sum_{\bar{x}_i, \bar{x}_j} m_{i \to j}(\bar{x}_i, \bar{x}_j) m_{j \to i}(\bar{x}_j, \bar{x}_i). \tag{3.78}$$

It is useful to define

$$z_{i \to j} = \sum_{\bar{x}_i, \bar{x}_j} \sum_{\bar{\mathbf{x}}_{\partial i \setminus j}} \prod_{t=0}^{T-1} f_i^{t+1}(x_i^{t+1} | \mathbf{x}_{\partial i}^t, x_i^t) \prod_{k \in \partial i \setminus j} m_{k \to i}(\bar{x}_k, \bar{x}_i) = \frac{z_i}{z_{ij}}. \tag{3.79}$$

Finally,

$$F = \sum_i \left[\left(\frac{d_i}{2} - 1 \right) \log z_i - \frac{1}{2} \sum_{j \in \partial i} \log z_{i \to j} \right]. \tag{3.80}$$

The last manipulation is useful in that this expression of the free energy only depends on $\{z_i, z_{i \to j}\}$. These quantities are computed in any case when one normalizes messages and beliefs, resulting in no computational overhead with respect to just performing a BP iteration.

3.6 Reduction in computational complexity

Similarly to what happens for the matrix product cavity method [79], the matrices prior to truncation have dimensions of M^{z-1}, where M is the matrix size in the incoming messages, and z is the degree. The computational bottleneck arises from the sweeps of singular value decompositions (SVDs), resulting in a cost of $\mathcal{O}(M^{3(z-1)})$ for a single node BP update. Although it demonstrated in [80, section 6] that for Glauber dynamics this cost could be reduced to $\mathcal{O}(M^{2z-1})$, the exponential dependence on the degree remains problematic, even for graphs with moderate connectivity. Here, we present an enhanced method that, for a broad class of models, such as many in epidemiology and Glauber dynamics for some choice

of couplings, achieves a computational cost of $\mathcal{O}(M^6)$. The degree dependence becomes polynomial and is determined by the specifics of the model.

In many cases, transition probabilities $w(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t)$ depend on $\mathbf{x}_{\partial i}^t$ only through an intermediate variable that encodes the aggregate interaction with all neighboring nodes. For instance, in epidemic models such as SI, SIR, and SIRS, the transition probability only depends on whether at least one neighbor has infected node i. In Glauber dynamics, the transition probability relies on the local field, which is a weighted sum of neighboring spins. Formally, let us define intermediate scalar variables y_A^t, where $A \subseteq \partial i$, that summarize information about $\mathbf{x}_A^t$. From the definition of conditional probability, we can write:

$$p\left(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t\right) = \sum_{y_{\partial i}} p\left(x_i^{t+1}|y_{\partial i}^t, x_i^t\right) p\left(y_{\partial i}^t|\mathbf{x}_{\partial i}^t, x_i^t\right). \tag{3.81}$$

If it is the case that given $\mathbf{x}_{\partial i}^t$, the corresponding y variables are independent, then some simplifications are possible. More formally, if the following holds:

$$\begin{aligned} p\left(y_{A\cup B}^t|x_{A\cup B}^t, x_i^t\right) &= \sum_{y_A, y_B} p\left(y_{A\cup B}^t|y_A^t, y_B^t, x_i^t\right) \\ &\quad \times p\left(y_A^t, y_B^t|x_{\partial i}^t, x_i^t\right) \qquad (3.82) \\ &= \sum_{y_A, y_B} p\left(y_{A\cup B}^t|y_A^t, y_B^t, x_i^t\right) \\ &\quad \times p\left(y_A^t|x_A^t, x_i^t\right) p\left(y_B^t|x_B^t, x_i^t\right) \qquad (3.83) \end{aligned}$$

for any $A \cup B \subseteq \partial i$, then the outgoing messages can be computed recursively starting from just two quantities:

- $p\left(y_j^t|x_j^t, x_i^t\right)$
- $p\left(y_{A\cup B}^t|y_A^t, y_B^t, x_i^t\right)$.

This approach is more efficient than the naive implementation, provided the number of values each y can take does not grow exponentially with the number of x's involved.

Rewriting the BP equation (omitting ϕ terms for simplicity) using auxiliary variables $\{y^t_A\}_{A\subseteq\partial i}$ yields:

$$m_{i\to j}(\bar{x}_i,\bar{x}_j) \propto \sum_{\bar{\mathbf{x}}_{\partial i\setminus j}} \prod_t w(x_i^{t+1}|\mathbf{x}^t_{\partial i\setminus j},x_i^t,x_j^t) \prod_{k\in\partial i\setminus j} m_{k\to i}(\bar{x}_k,\bar{x}_i) \tag{3.84}$$

$$\propto \sum_{\bar{\mathbf{x}}_{\partial i\setminus j}} \sum_{\bar{y}_{\partial i\setminus j}} \prod_t p(x_i^{t+1}|y^t_{\partial i\setminus j},x_i^t,x_j^t) p(y^t_{\partial i\setminus j}|\mathbf{x}^t_{\partial i\setminus j},x_i^t) \prod_{k\in\partial i\setminus j} m_{k\to i}(\bar{x}_k,\bar{x}_i) \tag{3.85}$$

$$\propto \sum_{\bar{y}_{\partial i\setminus j}} \prod_t p(x_i^{t+1}|y^t_{\partial i\setminus j},x_i^t,x_j^t) \tilde{m}_{\partial i\setminus j\to i}(\bar{y}_{\partial i\setminus j},\bar{x}_i) \tag{3.86}$$

where $\tilde{m}_{\partial i\setminus j\to i}(\bar{y}_{\partial i\setminus j},\bar{x}_i) = \sum_{\bar{\mathbf{x}}_{\partial i\setminus j}} \prod_t p(y^t_{\partial i\setminus j}|\mathbf{x}^t_{\partial i\setminus j},x_i^t) \prod_{k\in\partial i\setminus j} m_{k\to i}(\bar{x}_k,\bar{x}_i)$.

Now, $\tilde{m}_{\partial i\setminus j\to i}$ can be determined as the aggregation of all messages $\tilde{m}_{k\to i}$ for $k<j$ and $k>j$:

$$\tilde{m}_{\partial i\setminus j\to i}(\bar{y}_{\partial i\setminus j},\bar{x}_i) = \sum_{\bar{y}_{<j}} \sum_{\bar{y}_{>j}} \prod_t p(y^t_{\partial i\setminus j}|y^t_{<j},y^t_{>j},x_i^t) \tilde{m}_{<j}(\bar{y}_{<j},\bar{x}_i) \tilde{m}_{>j}(\bar{y}_{>j},\bar{x}_i). \tag{3.87}$$

This can be naturally expressed in matrix product form as:

$$\begin{aligned} &\left[\tilde{A}^t_{\partial i\setminus j\to i}(y^t_{\partial i\setminus j},x_i^t)\right]_{(a^t,b^t),(a^{t+1},b^{t+1})} = \\ &\sum_{y^t_{<j}} \sum_{y^t_{>j}} p(y^t_{\partial i\setminus j}|y^t_{<j},y^t_{>j},x_i^t) \left[\tilde{A}_{<j}(y^t_{<j},x_i^t)\right]_{a^t,a^{t+1}} \left[\tilde{A}_{>j}(y^t_{>j},x_i^t)\right]_{b^t,b^{t+1}} \end{aligned} \tag{3.88}$$

where the matrix subscripts correspond to the messages in (3.87). The matrix on the left-hand side has double the size (double the number of rows, columns) of those on the right-hand side, necessitating SVD-based truncations as outlined earlier. The computational bottleneck arises from performing SVD on $\tilde{A}^t_{\partial i\setminus j\to i}$, which is reshaped such that (a^t,b^t) form the row index and $(a^{t+1},b^{t+1},y^t_{\partial i\setminus j},x_i^t)$ form the column index. The cost is $\mathscr{O}(nM^6)$, where n is the number of values taken by $y^t_{\partial i\setminus j}$. The latter depends on the model. Provided n grows polynomially with the degree $z=|\partial i|$, the exponential cost is avoided.

The messages $\tilde{m}$ can be recursively computed, as they exhibit similar properties to (3.83):

$$\tilde{m}_{A\cup B}(\overline{y}_{A\cup B},\overline{x}_i)=\sum_{\overline{y}_A,\overline{y}_B}\prod_t p(y^t_{A\cup B}|y^t_A,y^t_B,x^t_i)\tilde{m}_A(\overline{y}_A,\overline{x}_i)\tilde{m}_B(\overline{y}_B,\overline{x}_i) \tag{3.89}$$

starting with $\tilde{m}_{\{k\}\to i}(\overline{y}_{\{k\}},\overline{x}_i)=\sum_{\overline{x}_k}\prod_t p(y^t_{\{k\}}|x^t_k,x^t_i)m_{k\to i}(\overline{x}_k,\overline{x}_i)$ and $\tilde{m}_{\emptyset\to i}(\overline{y}_\emptyset,\overline{x}_i)\propto 1\,\forall\,(\overline{y}_\emptyset,\overline{x}_i)$. Then, using (3.86), we can compute $m_{i\to j}(\overline{x}_i,\overline{x}_j)$ for all j. As shown in (3.44), the matrices now depend on both x^{t+1}_i and x^t_i:

$$B^t_{i\to j}(x^{t+1}_i,x^t_i,x^t_j)=\sum_{y^t_{\partial i\setminus j}} p(x^{t+1}_i|y^t_{\partial i\setminus j},x^t_i,x^t_j)\tilde{A}_{\partial i\setminus j\to i}(y^t_{\partial i\setminus j},x^t_i). \tag{3.90}$$

These matrices are processed similarly to the generic BP implementation detailed in the main text. At this point, the already computed quantities can be used to retrieve the belief at node i:

$$b_i(\overline{x}_i)\propto\sum_{\overline{y}_{\partial i}}\prod_t p(x^{t+1}_i|y^t_{\partial i},x^t_i)\tilde{m}_{\partial i\to i}(\overline{y}_{\partial i},\overline{x}_i) \tag{3.91}$$

where j is any neighbor of node i.

The procedure is linear in degree, yielding an overall cost of $\mathcal{O}(znM^6)$ to update all outgoing messages from a node. In cases where no convenient choice of auxiliary variables y exists, the scheme can still be applied with $y^t_A=\otimes_{a\in A}\{x^t_A\}$ and $n\sim q^z$, recovering the exponential dependence on degree.

In the following we give the details of the efficient update scheme for the SIS model and some class of Glauber dynamics.

SIS model

For the SIS model (and similarly for SIRS), we define y_A^t as the event that at least one neighbor in A infects node i:

$$p\left(y_k^t|x_k^t,x_i^t\right) = \begin{cases} \lambda_{ki}\delta(y_j^t,I) + (1-\lambda_{ki})\delta(y_k^t,S) & \text{if } x_k^t = S \\ \delta(y_k^t,S) & \text{otherwise} \end{cases} \tag{3.92}$$

$$p\left(y_{A\cup B}^t|y_A^t,y_B^t,x_i^t\right) = \delta(y_{A\cup B}^t,I)\mathbb{1}\left[y_A^t = I \vee y_B^t = I\right] + \delta(y_{A\cup B}^t,S)\mathbb{1}\left[y_A^t = S \wedge y_B^t = S\right]. \tag{3.93}$$

In this scenario, all y variables are binary, resulting in a computational cost of $\mathscr{O}(zM^6)$ to update z messages.

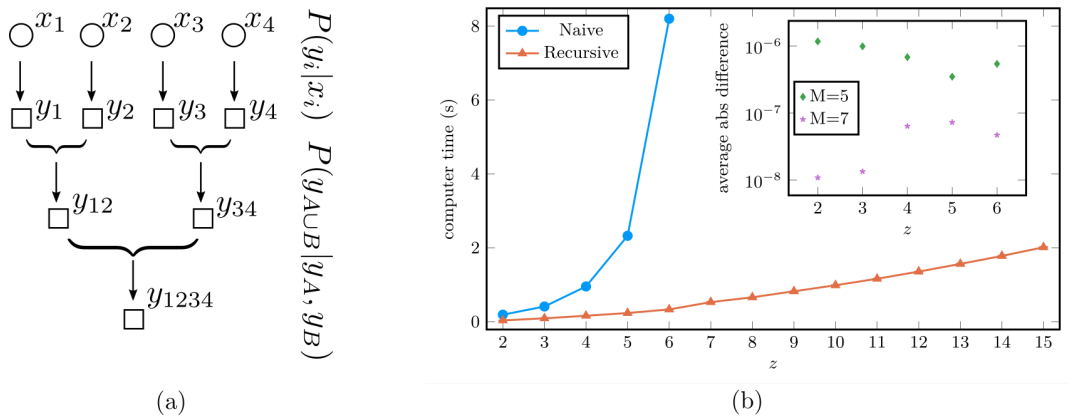
Glauber dynamics

In the case of parallel Glauber dynamics, the most general setting where these simplifications hold involves couplings with constant absolute values $\left|J_{ij}\right| \equiv J$ and arbitrary external fields, often referred to as the $\pm J$ Ising model. The case $J_{ij} \equiv J$, $h = 0$ is automatically covered. The transition probability (3.10) takes the form:

$$e^{\beta\sigma_i^{t+1}\left[J\left(\sum_{j\in\partial i}\operatorname{sign}(J_{ij})\sigma_j^t\right)+h_i\right]} \propto e^{\beta\sigma_i^{t+1}\left[J\left(y_{\partial i\setminus j}^t+\operatorname{sign}(J_{ij})\sigma_j^t\right)+h_i\right]} \tag{3.94}$$

where $y_A^t = \sum_{k\in A}\operatorname{sign}(J_{ik})\sigma_k^t$. It can be easily seen that $p\left(y_{\{k\}}^t|\sigma_k^t,\sigma_i^t\right) = \delta\left(y_{\{k\}}^t,\operatorname{sign}(J_{ik})\sigma_k^t\right)$ and $p\left(y_{A\cup B}^t|y_A^t,y_B^t,\sigma_i^t\right) = \delta\left(y_{A\cup B}^t,y_A^t+y_B^t\right)$. In this case, y_A^t can take values from $-|A|,-|A|+2,\ldots,|A|-2,|A|$, resulting in $2|A|+1$ possible values. The maximum range is achieved when $A = \partial i\setminus j$, leading to a computational cost of $\mathscr{O}(z^2M^6)$ to update z messages.

Figure 3.3 illustrates the recursive procedure and displays the time required to execute 10 iterations of MPBP for an SIS model on a star graph (one central node connected to z others) of varying sizes. The naive approach shows exponential growth in computational time, while the recursive strategy results in linear behavior.



(a) (b)

Fig. 3.3 (a) Graphical depiction of the recursive procedure described in this section. (b) Computer time to run 10 iterations of MPBP with the naive vs recursive update for a SIS model on a star graph of degree z, $\lambda = 0.2, \rho = 0.1, \gamma = 0.05$, no reweighting, bond dimension 5, average over 20 random instances. Inset: absolute difference between values of the marginals for the two methods, averaged over epochs, sites, and instances, for two values of bond dimension. Such small errors happen because the recursive method truncates after incorporating each of the intermediate y-dependent messages.

3.7 Results

In this section we illustrate the effectiveness of MPBP applied to dynamics of epidemic spreading and of the kinetic Ising model. We first focus on free dynamics, showing results that are at least comparable with the existing methods, often more accurate. Then we move to reweighted processes, where our approach really represents an innovation.

3.7.1 Non-reweighted epidemic processes

As examples of free dynamics, we estimate the marginal probability of an individual being in the infectious state under the SIS model, in several settings. As a comparison we report the curves obtained using a discrete-time version of Dynamic message Passing (DMP) [72], Individual-Based Mean Field (IBMF) [97], and Cavity Master Equation (CME) [74], which were originally devised for continuous time evolution. These methods were described earlier in section 3.3.1, while the connection between continuous-time processes and their discretized counterpart in section 3.1.2. In the legends of figures 3.5, 3.4, 3.6, discretized version are denoted by a "-d" prefix.

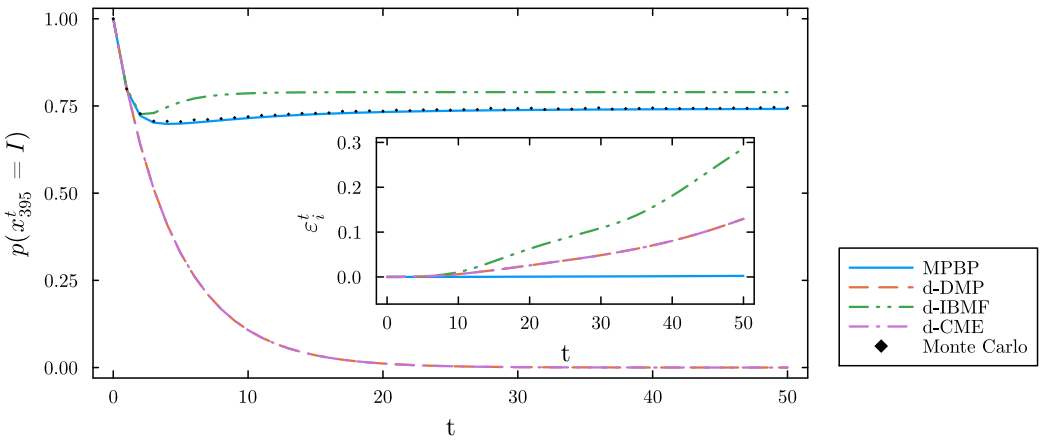


Fig. 3.4 Marginal of node 395, the most connected one of a random tree with $N = 1000$ nodes, $\lambda = 0.3, \rho = 0.2$. Node 395 is the only infectious at time zero. Bond dimension 12. Inset: error with respect to Monte Carlo.

We evaluate the accuracy of each method by considering the average absolute error with respect to a Monte Carlo simulation

$$\varepsilon_i^t = \frac{1}{N}\sum_{i=1}^{N}\left|p(x_i^t = I) - p^{MC}(x_i^t = I)\right|. \tag{3.95}$$

Fig. 3.4 shows predictions for the probability of being in the infectious (I) state for a single node on a random tree. In this case a single node was picked as the sole infectious individual at time zero. At difference with the mean-field methods, MPBP is exact on trees, modulo the matrix product approximations. Here one can see how the agreement with the Monte Carlo estimate is good for modest bond dimension (10).

Fig. 3.5 shows the average probability for a single variable to be infectious on an Erdos-Renyi (ER) graph of size $N = 1000$ nodes. In this case the probability of being the zero patient was picked uniform for all nodes $\gamma_i \equiv \gamma$. Since ER graphs are asymptotically tree-like, we expect the cavity approximation to hold exactly in this case.

The same analysis is repeated on Zachary's karate club graph [98] (fig.3.6), the same benchmark used in [72, 73].

Finally, we compare MPBP against three continuous-time methods, DMP, IBMF and CME, on the karate club graph (fig. 3.7). The comparison is made by multiplying

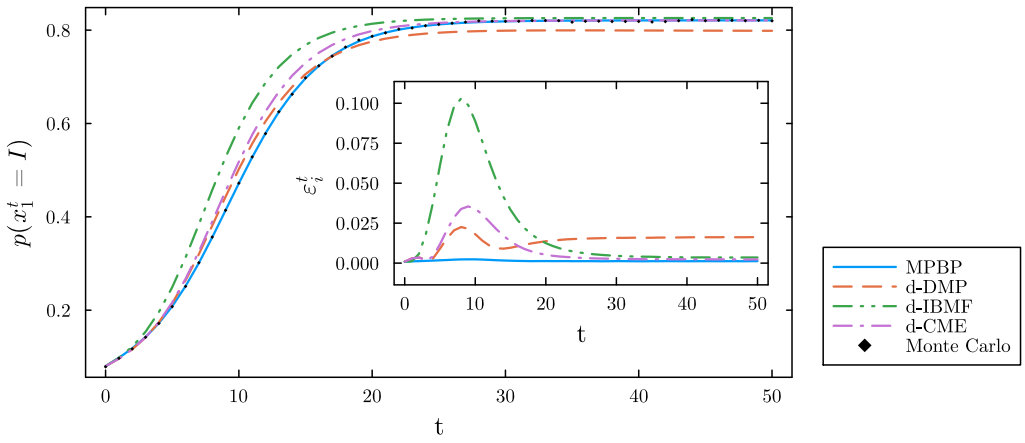


Fig. 3.5 Marginal of node 1 of a ER graph with $N = 1000$ nodes, average connectivity $c = 5$, $\lambda = 0.1, \rho = 0.05, \gamma = 0.08$. Bond dimension 10. Inset: error with respect to Monte Carlo.

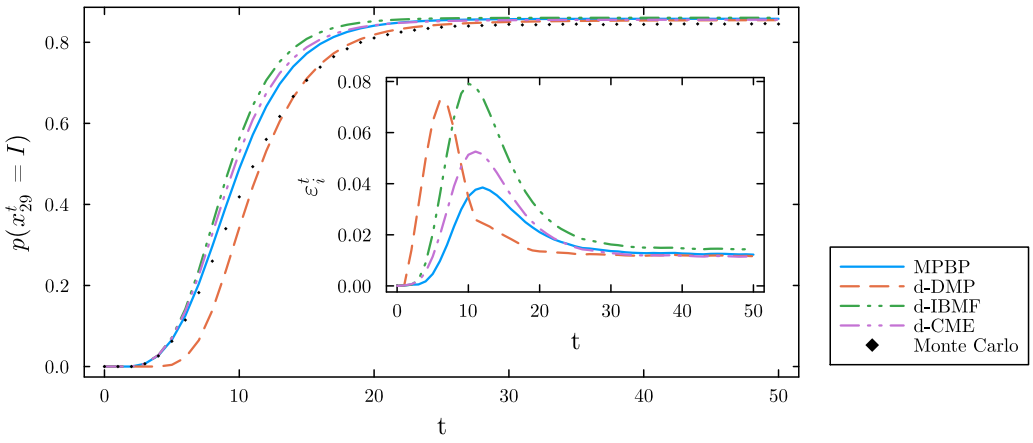


Fig. 3.6 Marginal of node 29 (zero-based numbering to match previous works) of Zachary's karate club network, $N = 34$ nodes, $\lambda = 0.1, \rho = 0.05$, node 0 is the only infectious at time zero. Bond dimension 10. Inset: error with respect to Monte Carlo.

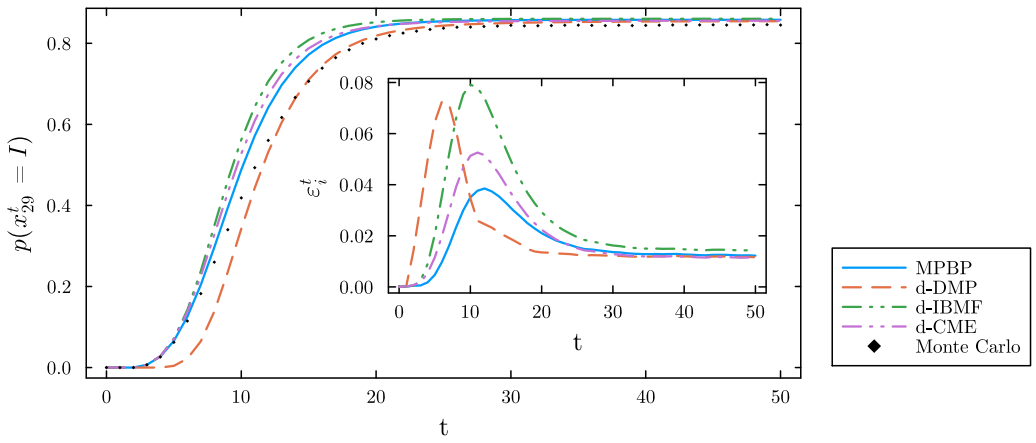


Fig. 3.7 Comparison with continuous-time mean-field methods. Marginal of node 29 (zero-based numbering to match previous works) of Zachary's karate club network, $N = 34$ nodes, $\lambda = 0.1, \rho = 0.05$, node 0 is the only infectious at time zero. Bond dimension 10. Inset: error with respect to Monte Carlo.

the transmission and recovery rates for the continuous setting λ, ρ by the time-step Δt (in this case $\Delta t = 1$) to turn them into probabilities to be handled by MPBP. MPBP gives the best overall prediction across the considered window.

It must be pointed out that although MPBP shows by far the best performance in these comparison, the other considered methods are significantly simpler. On the other hand, none of the other methods is devised to analyze reweighted dynamics.

3.7.2 Epidemic reconstruction

Moving to reweighted processes, fig.3.8 shows the efficacy of MPBP when performing inference of trajectories given some observations. On a small ($N = 23$) random graph, a 10-step trajectory $\overline{\mathbf{y}}$ was sampled from a SIS prior distribution with $\lambda = 0.15, \rho = 0.12, \gamma = 0.13$. We then observed the state of a random half $I \subset V$ of the nodes, added the corresponding reweighting factors $\prod_{i \in I} \phi_i^T(x_i^T) = \prod_{i \in I} \delta(y_i^T, x_i^T)$ and performed inference using (3.17). The MPBP estimate for the posterior marginals, obtained with matrices of size 3, agrees almost perfectly with Monte Carlo simulations. This is good indication that MPBP applied to sparse problems will keep giving accurate results even when on larger and/or more constrained instances where Monte Carlo methods fail, leaving little to compare against.

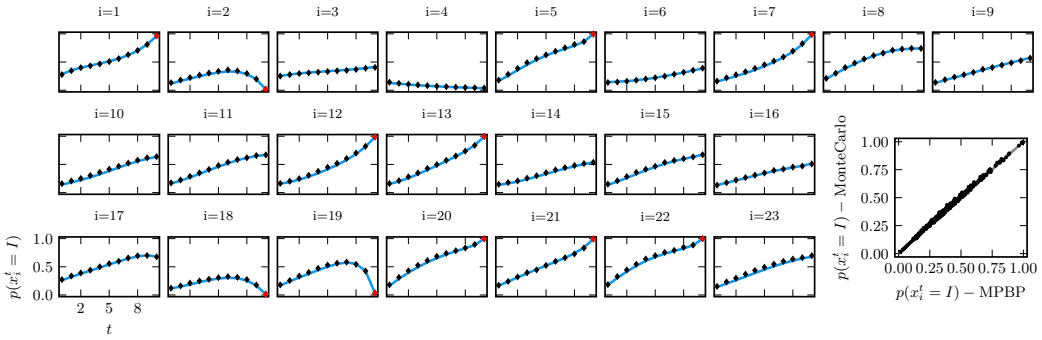


Fig. 3.8 MPBP (solid line) with bond dimension 3 correctly computes marginals of an SIS model defined on an Erdos-Renyi graph with 23 nodes and average connectivity 4, $\lambda = 0.15, \rho = 0.12, \gamma = 0.13$. The state of a random half of the variables was observed at final time $T = 10$ and used to reweight the distribution (red dots). Black dots are the average over 10^6 Monte Carlo simulations. (Bottom-right) Comparison of all points from the previous plots, the Pearson correlation coefficient is 0.9986.

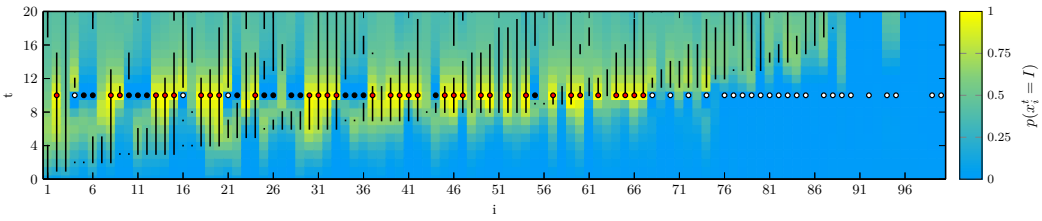


Fig. 3.9 Inference on a single epidemic outbreak sampled from a SIRS model on an Erdos-Renyi graph with average connectivity $c = 2.5$, $N = 100$. Bond dimension $M = 3$. The process to be inferred was drawn from a SIRS prior with $\lambda = 0.4, \rho = \sigma = 0.15, \gamma = 0.01$, the same parameters were used for the inference. The state of 75% of the nodes was observed at time 10 (white=S, red=I, black=R) and used to reweight the distribution. Black lines correspond to true infection periods.

Realistic scenarios are often better described by the Susceptible-Infectious-Recovered-Susceptible (SIRS) model where transmission of infections is analogous to the SIS case, but an infectious node i can recover with probability ρ_i and a recovered become susceptible again with probability σ_i. From a practical point of view, extending the SIR to SIRS in the MPBP framework takes little effort: it suffices to enrich the factors with the new transition $R \to S$. Fig. 3.9 shows the performance of MPBP at estimating the posterior trajectories for a single realization of an epidemic drawn from a prior whose parameters $\lambda, \rho, \sigma, \gamma$ are homogeneous and known. The state of a random three quarters of the system was observed at an intermediate time (colored dots). We see good agreement between the true infection

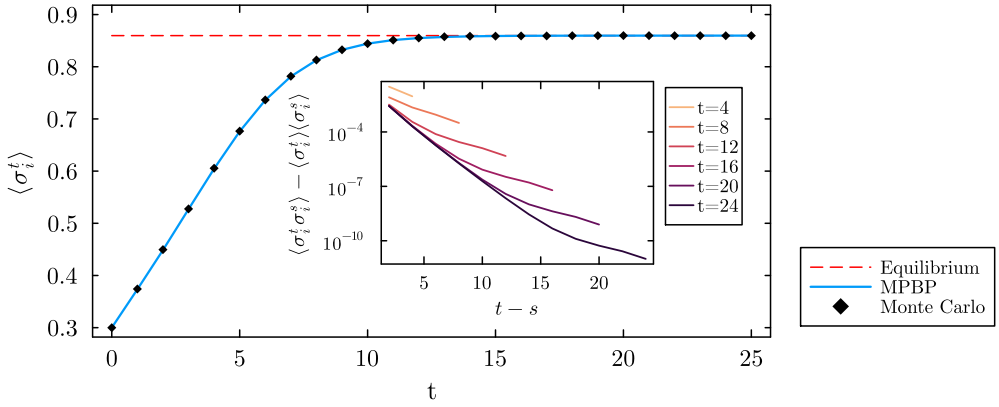


Fig. 3.10 Time evolution of the average magnetization $\langle \sigma_i^t \rangle$ for an infinite 8-Random Regular Graph, $\beta J = 0.2$, compared against Monte Carlo simulation on a graph of size $N_{MC} = 5000$. Bond dimension 25. The dashed horizontal line is the equilibrium value for the magnetization of the underlying Ising model. Inset: time autocovariances computed by MPBP.

times (black lines) and the marginal probabilities of being Infectious (in yellow). Nodes are sorted in increasing order of true first infection time.

3.7.3 Non-reweighted Glauber dynamics

As examples of free dynamics we consider the evolution of magnetization $\langle \sigma_i^t \rangle$ and time autocovariance $\langle \sigma_i^t \sigma_i^s \rangle - \langle \sigma_i^t \rangle \langle \sigma_i^s \rangle$ for pairs of epochs (t,s), on ferromagnetic, Random Field and spin-glass Ising Models, under the Markov transition (3.10). For the results in figures 3.10 and 3.12 we picked the same settings as in [77].

First we consider a model with uniform couplings $J_{ij} \equiv J$ on an infinite Random Regular Graph like the one studied in [79] but with degree 8 instead of 3, taking advantage of the computational cost linear in the number of neighbors. Fig. 3.10 shows the time evolution of the average magnetization, compared against Monte Carlo simulations on a large graph. We also report the equilibrium value for the magnetization of the underlying Ising model, computed by standard belief propagation, to show empirically that the dynamics converge to the correct equilibrium state at large times. A discussion on why this is expected to be the case can be found in appendix B.1.1. The inset shows the average time autocovariance for single nodes. We did not report Monte Carlo estimates here because the sampling error ($\sim \sqrt{N_{samples}}$) is orders of magnitude larger than the values themselves. Indeed,

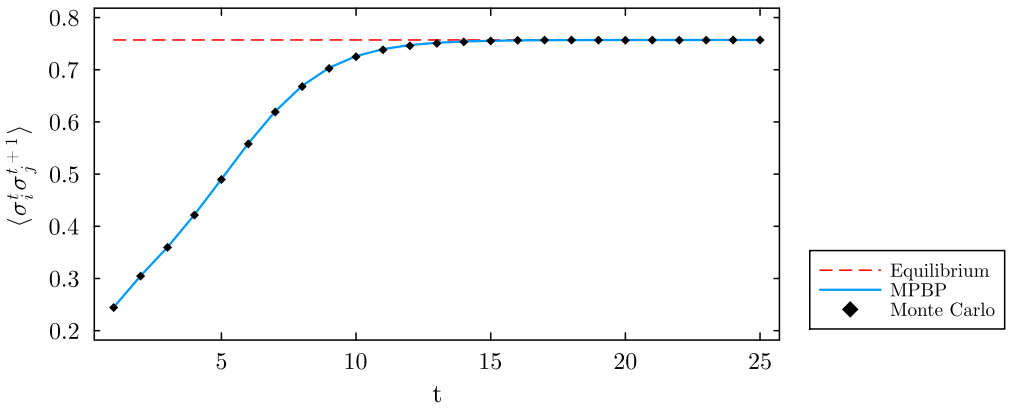


Fig. 3.11 Time evolution of the average pair-wise correlation $\langle \sigma_i^t \sigma_j^{t+1} \rangle$ for an infinite 8-Random Regular Graph, $\beta J = 0.2$, compared against Monte Carlo simulation on a graph of size $N_{MC} = 5000$. Bond dimension 25 (same settings as fig. 3.10). The dashed horizontal line is the equilibrium value for the pairwise correlation $\langle \sigma_i \sigma_j \rangle$ of the underlying Ising model.

analytical methods such as MPBP offer a precious alternative to Monte Carlo whenever one is interested in estimating quantities that are close to zero.

For the same system, we also compute pair-wise correlations at subsequent epochs $\langle \sigma_i^t \sigma_j^{t+1} \rangle$ for neighboring variables i, j, $(ij) \in E$. At large times, this is the correct quantity to look at when comparing to the equilibrium value of the correlation in the underlying Ising model. This is due to the nature of the parallel update (see appendix B.1.1 for further details).

Next, we apply MPBP to an infinite Erdos-Renyi graph, again with uniform couplings and in the ferromagnetic phase, using the population dynamics approach presented in section 2.4.4. Figure 3.12 shows the magnetization as a function of time. Similarly to the previous case, the equilibrium value of the magnetization can be obtained by population dynamics on the underlying, static, Ising model.

We move on to study a Random Field Ising Model (RFIM) with uniform couplings and random external fields $h_i = \pm h$ on a large finite graph. Magnetization and autocovariances are plotted in fig. 3.13.

It is interesting to investigate the behavior of Glauber dynamics when the interactions are non-reciprocal, i.e. $J_{ij} \neq J_{ji}$ in general. This does not have a static counterpart. Indeed, the dynamics does not converge to any equilibrium state. Figure 3.14 shows the time evolution of the average magnetization on a random regular

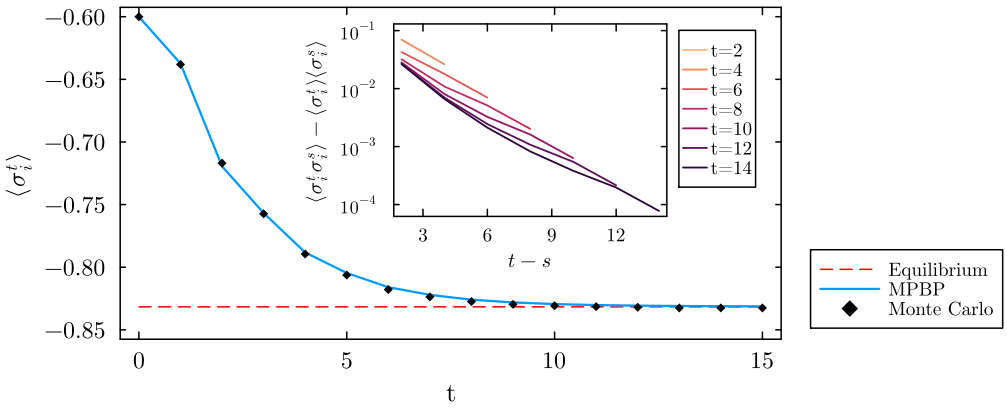


Fig. 3.12 Time evolution of the average magnetization $\langle \sigma_i^t \rangle$ for an infinite Erdos-Renyi graph with mean connectivity $c = 4$, $\beta J = 0.5$, compared against Monte Carlo simulation on a graph of size $N_{MC} = 5000$. Bond dimension 18. The dashed horizontal line is the equilibrium value for the magnetization of the underlying Ising model. Inset: time autocovariances computed by MPBP.

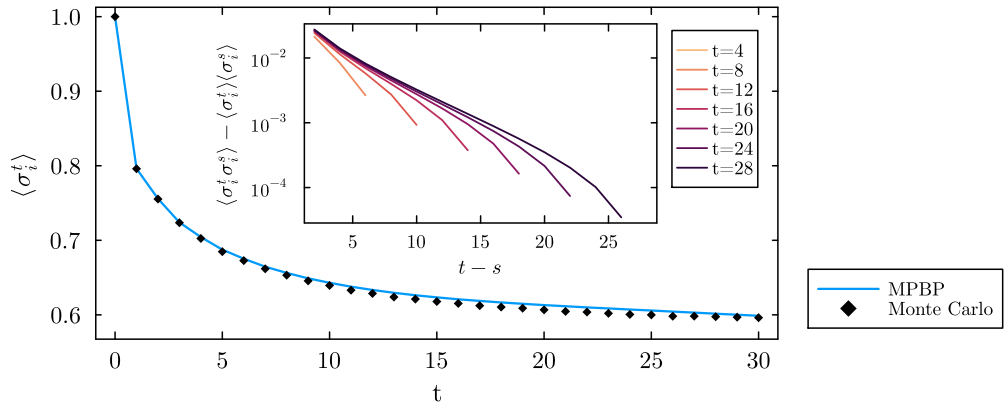


Fig. 3.13 Time evolution of the average magnetization $\langle \sigma_i^t \rangle$ for an on Erdos-Renyi graph with mean connectivity $c = 3, \beta J = 2/c$, $N = N_{MC} = 1000$ and $\beta h_i = \pm 0.6$ sampled uniformly, bond dimension 10. Inset: time autocovariances computed by MPBP.

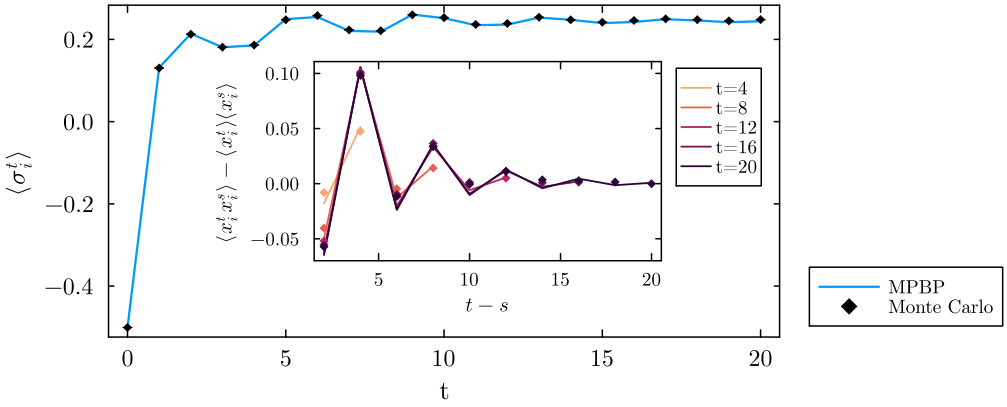


Fig. 3.14 Time evolution of the average magnetization $\langle \sigma_i^t \rangle$ for a random regular graph with connectivity $c = 4$, directed couplings βJ_{ij} sampled i.i.d. from $Uniform(\{\pm 1\})$, $N = N_{MC} = 1000$ and $\beta h = 0.5$. Bond dimension 12. Inset: time autocovariances computed by MPBP compared with Monte Carlo estimates (same color means same value of t).

graph of size $N = 50$, with directed coupling sampled uniformly and independently in $\{\pm 1\}$ and a uniform external field $h = 0.5$. Again, we see good agreement between the MPBP and Monte Carlo predictions, also for what concerns time autocovariances.

Finally, we consider an antiferromagnetic model with $J = -1$ at zero temperature (in practice we set a large number $\beta = 100$), focusing on the nearest-neighbor correlation $\langle \sigma_i^t \sigma_j^t \rangle$, $(i, j) \in E$ rather than the magnetization, which is null at steady state. Above the critical inverse temperature for an infinite random regular graph of degree 3 $\beta_c = \log\left(1 + \sqrt{2}\right)$ [99], the underlying Ising system is in a glassy phase. For this model we used Glauber updates (3.10) the modified version of the dynamics reported in (3.13) with $p_0 = 0.25$.

3.7.4 Large deviations of Glauber dynamics

As anticipated in section 3.2.1, the MPBP formalism allows for the description of rare trajectories. Here we study the large deviation behavior of a Glauber process $W(\overline{\sigma}) = \prod_{i=1}^{N} w(\sigma_i^0) \prod_{t=0}^{T-1} w(\sigma_i^{t+1} | \sigma_{\partial i}^t)$ by tilting it with an external field at final time $\prod_i \phi_i^T(\sigma_i^T) = \prod_i e^{h\sigma_i^T}$. In the thermodynamic limit $N \to \infty$ this allows to select a particular value for the magnetization at final time $m = \frac{1}{N} \sum_i \sigma_i^T$. As explained in

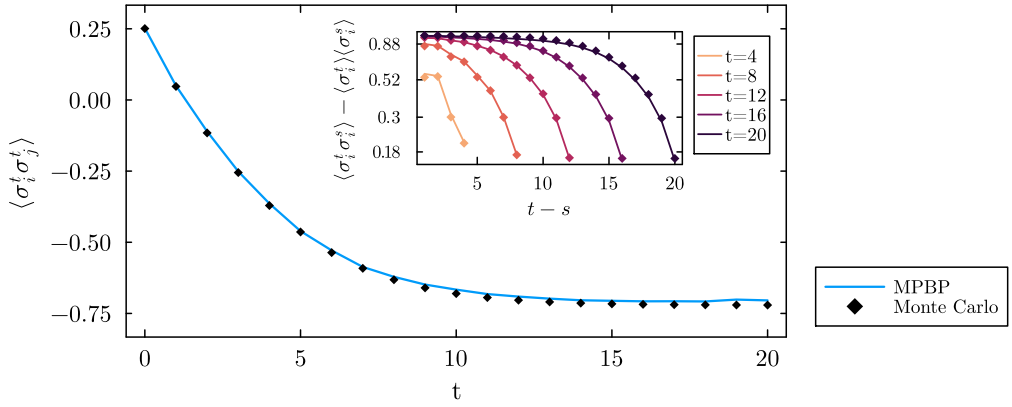


Fig. 3.15 Time evolution of nearest-neighbor correlations $\langle \sigma_i^t \sigma_j^t \rangle$ in an antiferromagnetic Ising model at zero temperature on a random regular graph with connectivity $c = 3$, $\beta J = -100$, $N = N_{MC} = 1000$. The Glauber dynamics here is the "damped" version (equation (3.13)) with $p_0 = 0.25$. Bond dimension 20. Inset: time autocovariances computed by MPBP compared with Monte Carlo estimates (same color means same value of t).

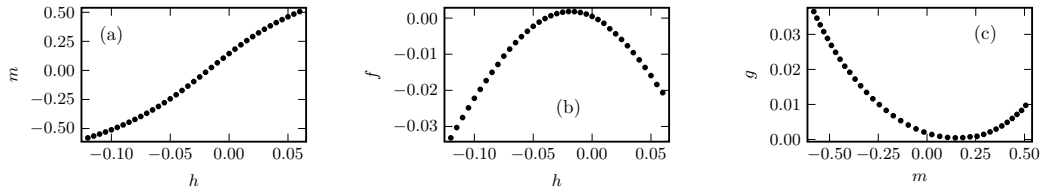


Fig. 3.16 Large deviation study of Glauber dynamics on an infinite 3-Random Regular Graph. Free dynamics with $\beta J = 0.6$, $T = 10$, magnetization at time zero $m^0 = 0.1$, zero external field, reweighted with an external field at final time $\prod_i \phi_i^T(\sigma_i^T) = \prod_i e^{h\sigma_i^T}$. (a) Magnetization vs reweighting field, (b) Bethe Free Energy vs reweighting field, (c) Magnetization-constrained free energy $g(m)$ vs magnetization. Bond dimension 25.

section 3.2.1, the Bethe free energy can be used to compute the probability

$$p(m) \sim e^{-Ng(m)} \tag{3.96}$$

that the system is found in a state with average magnetization m at the final time.

Fig. 3.16 shows the estimate of $g(m)$ for a ferromagnetic Ising model on an infinite random graph initialized at magnetization $m^0 = 0.1$ and evolving for $T = 10$ epochs. $p(m)$ has a minimum at $m \approx 0.145$ which corresponds to the free dynamics $h = 0$.

Such an analysis could not have been carried out by means of Monte Carlo methods since the probability of sampling a trajectory ending at m is infinitesimal, as is clear from the large deviation law in fig. 3.16.

3.8 Conclusion

In this chapter, we introduced Matrix Product Belief Propagation (MPBP) as a novel approach to computing marginals in reweighted Markov dynamics on sparse graphs. By leveraging the cavity approximation and matrix product states, MPBP offers a scalable solution to the computational challenges inherent in reweighted processes. The MPBP framework was shown to handle recurrent models and reweighted dynamics, thus overcoming limitations of earlier methods such as Dynamic Cavity and Dynamic Message Passing. Moreover, we demonstrated that MPBP can achieve accurate results with manageable computational complexity, upon careful selection of the bond dimension.

In conclusion, MPBP combines the strengths of tensor network methods with belief propagation to yield a versatile algorithm. As future developments, we envisage inferring the hyperparameters of the dynamics, such as the infection probability in epidemic processes.

Chapter 4

Steady state of Markov processes on graphs

Note: This final chapter contains results which were still preliminary at the date of submission of the manuscript. Later developments can be found in [100].

Staying in the realm of Markov processes on graphs, without reweighting, an interesting question concerns the steady state, i.e. the long time limit of the dynamics. The problem relates to questions such as "Will an epidemic process decay or be sustained on the long run?", or "How will a certain biological system behave after an initial transient?", or yet "Can we build computers out of animal neurons by reading off the results of calculations from the stationary response of their activations?"

This chapter proposes a method that, building on the matrix product cavity method [79] and matrix product belief propagation (MPBP) [47] discussed earlier in chapter 3, estimates the steady state distribution of Markov processes on graphs.

4.1 Introduction and related work

Given a homogeneous Markov process

$$p(\overline{\mathbf{x}}) = p(\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^T) = p(\mathbf{x}^0) \prod_{t=0}^{T-1} p(\mathbf{x}^{t+1} | \mathbf{x}^t) \tag{4.1}$$

with transitions encoded in a time-independent matrix W

$$p(\mathbf{x}^{t+1}|\mathbf{x}^t) = W_{\mathbf{x}^{t+1},\mathbf{x}^t}, \tag{4.2}$$

stationary states correspond to the dominant (right) eigenvectors of W. However, finding these is often computationally infeasible when $\mathbf{x}$ lives in a huge space, as in the cases we have considered so far. For a system of N interacting variables each taking one of q values, the transition matrix has q^{2N} entries.

It is well known that a certain class of Markov dynamics, those satisfying the detailed balance condition, converge to an equilibrium distribution [101]. This provides a consistent simplification, in that the equilibrium distribution is typically much simpler to analyze, for instance with the statistical mechanics-inspired methods presented in chapters 1 and 2. For dynamics without equilibrium, alternative methods must be employed.

Surely, one can adopt one of the method developed to track the full dynamics and let it run for how long it needs to reach a steady state. Such strategies, already presented in the previous chapter, include Monte Carlo simulations [63] and analytical approximations [70, 72, 73, 97, 87]. The main issue with sampling, besides the potentially arbitrary number of epochs one needs to wait before reaching the steady state, is that observables computed in this way suffer from an intrinsic uncertainty of the order of the square root of the number of samples. This, as we will see later with a practical example (see fig. 4.2), can become problematic whenever the value of the quantity of interest approaches zero. Mean-field approximations are typically more efficient than Monte Carlo, but suffer from a few limitations: difficulty of dealing with recurrent dynamics, failure to capture time correlations. The *backtracking* version of the dynamic cavity method [102] describes processes of the class we are interested in as they approach an attractor. However, it suffers from exponential computational complexity in the number of epochs considered.

We showed earlier how the proposed MPBP algorithm can overcome some of these limitations. The purpose of this chapter is to adapt MPBP using a slightly different ansatz for the messages, in order to directly describe the steady state without having to track the transient of the dynamics. Open source code for the algorithm described in the following will be available upon publication at [103].

4.2 A parametrization invariant to time translations

In the MPBP algorithm, messages are parametrized by a matrix product state, a.k.a tensor train. We report equation (3.41) here for completeness:

$$m_{i\to j}(\overline{x}_i,\overline{x}_j) \propto \prod_{t=0}^{T} A^t_{i\to j}(x^t_i,x^t_j) \quad . \tag{4.3}$$

We are interested in the $T\to\infty$ limit, where we expect any property of the system, including BP messages, to obey time-shift invariance. It therefore makes sense to restrict the ansatz to a unique time-independent tensor $A^0 = A^1 = \cdots = A^T = A$. We propose a class of distributions parametrized by a single tensor A, repeated infinitely many times, i.e.

$$\begin{aligned} q_A(\overline{x}) =& \mathrm{Tr}\left[\prod_{\mathrm{t}=0}^{\mathrm{T}} \mathrm{A}(\mathrm{x}^{\mathrm{t}})\right] \\ =& \sum_{\alpha^0,\alpha^1,\ldots,\alpha^T} \left[A(x^0)\right]_{\alpha^0,\alpha^1} \left[A(x^1)\right]_{\alpha^1,\alpha^2} \cdots \left[A(x^T)\right]_{\alpha^T,\alpha^0} \end{aligned} \tag{4.4}$$

where Tr is the matrix trace and the limit $T\to\infty$ is intended. Such a mathematical object is referred to as a Uniform MPS[1] (UMPS) [104, 105].

The infinite time limit is handled somehow non-rigorously here, as in other definitions of UMPSs from quantum mechanics [104, 105]. The trace operator, which amounts to imposing the periodic boundary conditions implied by the time-shift property, is often omitted as it is not really important in the $T\to\infty$ limit. Indeed, whatever finite subset of the UMPS we might be interested in, "sees" to its right and to its left an infinite train of matrices which effectively cancel the influence of boundary conditions.

It is not immediately clear why a UMPS assumption should be correct for a version of MPBP describing the steady state. We justify it by arguing that it is indeed the correct description for a process slightly different than the one we are interested in (4.1), then showing that any finite-time statistics are equal for the two processes. Consider a modified version of the homogeneous Markov chain (4.1) where one

[1]The corresponding more "mathematics" nomenclature *Uniform Tensor Train* is not commonly used.

adds a further transition that loops back from time T to time 0

$$p_T\left(\mathbf{x}^0,\ldots,\mathbf{x}^T\right) = \frac{1}{Z_T} w\left(\mathbf{x}^0|\mathbf{x}^T\right) \prod_{t=0}^{T-1} w\left(\mathbf{x}^{t+1}|\mathbf{x}^t\right). \tag{4.5}$$

Its marginal at time $t = 0$ is given by

$$\begin{aligned} p_T(\mathbf{x}^0 = \tilde{\mathbf{x}}) =& \frac{1}{Z_T} \sum_{\mathbf{x}^1,\ldots,\mathbf{x}^T} w\left(\mathbf{x}^0|\mathbf{x}^T\right) \prod_{t=0}^{T-1} w\left(\mathbf{x}^{t+1}|\mathbf{x}^t\right) \Big|_{\mathbf{x}^0=\tilde{\mathbf{x}}} \\ =& \frac{1}{Z_T} \left(W^{T+1}\right)_{\tilde{\mathbf{x}},\tilde{\mathbf{x}}} \end{aligned} \tag{4.6}$$

where $(W^n)_{i,j}$ is the i,j element of the n-th matrix power of the transition matrix (4.2).

On the other hand, consider the following conditional probability induced by the original process (4.1) ending at $t = T+1$

$$\begin{aligned} p(\mathbf{x}^{T+1} = \tilde{\mathbf{x}}|\mathbf{x}^0 = \tilde{\mathbf{x}}) =& \frac{1}{w^0(\mathbf{x}^0)} w^0(\mathbf{x}^0) \sum_{\mathbf{x}^1,\ldots,\mathbf{x}^T} \prod_{t=0}^{T-1} p(\mathbf{x}^{t+1}|\mathbf{x}^t) \\ =& \left(W^{T+1}\right)_{\tilde{\mathbf{x}},\tilde{\mathbf{x}}} \end{aligned} \tag{4.7}$$

which is equal to (4.6) up to normalization. Now, and this is the final step, provided that the transition W satisfies the hypotheses of the Perron-Frobenius theorem, one has that, in the limit $T \to \infty$, the process (4.1) converges to a unique stationary distribution π [106]. This means that the system at any large time is distributed according to π regardless of the initial condition, i.e.

$$p(\mathbf{x}^{T+1} = \tilde{\mathbf{x}}|\mathbf{x}^0 = \tilde{\mathbf{x}}) = \left(W^{T+1}\right)_{\tilde{\mathbf{x}},\tilde{\mathbf{x}}} \overset{T\to\infty}{\longrightarrow} \pi(\tilde{\mathbf{x}}). \tag{4.8}$$

The benefit of working with (4.5) instead of (4.1) is that the former is by construction invariant for cyclic shifts of its inputs, i.e. $p_T(\mathbf{x} = \tilde{\mathbf{x}}) = p_T(\mathbf{x} = \pi_t(\tilde{\mathbf{x}}))$ where π_t is a cyclic permutation $\pi_t(\mathbf{x}^0,\ldots,\mathbf{x}^T) = \mathbf{x}^t,\ldots,\mathbf{x}^{T+t \mod T}$. As a consequence, all MPBP messages will also be invariant to time shifts and hence admit a UMPS representation of the form (4.4). Indeed, it is known that any cyclic-invariant distribution can be parametrized by a UMPS [81, Theorem 3], albeit with bond dimension potentially exponentially large in T.

4.3 Belief Propagation and truncations

4.3.1 The BP equations

We consider processes on a graph $G = (V = \{1,2,\ldots,N\}, E \subset V \times V)$ where the Markov transition takes the form

$$w\left(\mathbf{x}^{t+1}|\mathbf{x}^{t}\right) = \prod_{i=1}^{N} w_i(x_i^{t+1}|\mathbf{x}_{\partial i}^{t}, x_i^{t}) \tag{4.9}$$

with ∂i denoting the neighborhood of node i in G. As anticipated, we propose a UMPS ansatz (4.4) for the messages

$$m_{i\to j}(\bar{x}_i, \bar{x}_j) = \mathrm{Tr}\left[\prod_{t=0}^{T} A_{i\to j}(x_i^t, x_j^t)\right] \tag{4.10}$$

with $T \to \infty$. Replacing this in the MPBP update equation (3.40), one obtains

$$B_{i\to j}(x_i^{t+1}, x_i^t, x_j^t) = \sum_{\{x_k^t\}_{k\in\partial i\setminus j}} w_i(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) \bigotimes_{k\in\partial i\setminus j} A_{k\to i}(x_k^t, x_i^t). \tag{4.11}$$

The outgoing message $\mathrm{Tr}\left[\prod_{t=0}^{T} B_{i\to j}(x_i^{t+1}, x_i^t, x_j^t)\right]$ can be re-cast in the form (4.4) by first performing a SVD, $B_{i\to j}(x_i^{t+1}, x_i^t, x_j^t) = U(x_i^t, x_j^t)\Lambda V^\dagger(x_i^{t+1})$, then multiplying each $\Lambda V^\dagger$ factor in the infinite product by the following U factor to get

$$A_{i\to j}(x_i, x_j) = \Lambda V^\dagger(x_i) U(x_i, x_j). \tag{4.12}$$

This closes the BP equations under the UMPS ansatz.

Just like the in transient case, the BP equations (4.11),(4.12) are iterated until a fixed point is found. Then, formulas (3.63), (3.64) are used to compute single-variable and pair beliefs. These are then further marginalized to compute single-time and n-points (in time) statistics. The details of how such calculations can be performed for UMPS messages are given later in section 4.4.

The algorithm naturally allows for a "cavity method" version suited for ensembles of infinite graphs.

4.3.2 Truncating an infinite message

The BP update with UMPS ansatz presents the same shortcoming of its finite-time counterpart: the bond dimension (matrix size) grows exponentially throughout the iterations. Trying to apply here the same SVD-based strategy presented in the previous chapter, however, is not feasible, since it does not preserve the homogeneity of the parametrization by a single matrix A instead of a time-dependent A^t.

Instead, we turn to the literature on many-body quantum mechanics, where UMPSs have been used to describe one-dimensional systems with (spatially) homogeneous properties. In that context, several truncation methods have been developed, such as infinite Time Evolving Block Decimation [107], infinite DMRG [108] and Variational Uniform Matrix Product State (VUMPS) [105]. Here we use the VUMPS algorithm presented in [105] and implemented in [109]. The objective in VUMPS is to variationally maximizes the dot product between the target MPS q_A and truncated one $q_{A'}$

$$\lambda = \langle q_A | q_{A'} \rangle = \sum_{\mathbf{x}} q_A(\mathbf{x}) q_{A'}(\mathbf{x}). \tag{4.13}$$

The search for the optimal A' is done within the manifold of UMPSs with a fixed bond dimension to which one wishes to truncate to.

Such an objective function is naturally suited for the quantum realm, where q_A and $q_{A'}$ would be complex-valued, L_2-normalized quantum states. Indeed, in that case the maximizer of the overlap is equal to the minimizer of a distance between the two states

$$\begin{aligned} d_{A,A'} =& \sum_{\mathbf{x}} |q_A(\mathbf{x}) - q_{A'}(\mathbf{x})|^2 \\ =& \sum_{\mathbf{x}} |q_A(\mathbf{x})|^2 + \sum_{\mathbf{x}} |q_{A'}(\mathbf{x})|^2 - 2 \sum_{\mathbf{x}} \mathrm{Re}\left[q_A^*(\mathbf{x}) q_{A'}(\mathbf{x})\right] \\ =& 2\left\{1 - \mathrm{Re}\left[\langle q_A | q_{A'} \rangle\right]\right\} \end{aligned} \tag{4.14}$$

where by * we denote complex-conjugation. We argue the overlap to be a sensible choice of objective function also in our real-valued, L_1-normalized case as, for instance, it is maximized by $A = A'$.

The VUMPS algorithm works iteratively by starting from an initial guess for A' with a given bond dimension, then progressively refines the estimate. One iteration consists in the following steps [105, 110]:

1. Bring both the target and truncated UMPSs in the so-called *mixed gauge* or *mixed canonical form*. The canonical form has proven to provide numerical stability besides simplifications in the calculations in many tensor network applications [93].

2. Compute the gradient of the overlap in the new parametrization and set it to zero. This results in a self-consistency matrix equation together with another equation ensuring that the truncated UMPS is in the mixed gauge.

3. Restore the single-matrix form of the truncated UMPS.

Notice that the usual description of the VUMPS algorithm is given in terms of the truncation of a quantum UMPS to which a Hamiltonian operator has been applied [105]. We achieved plain truncation by passing a trivial identity Hamiltonian to the `approximate` function of the MPSKit.jl library [109].

For more thorough explanations of VUMPS we refer the reader to [105, 110, 111].

4.4 Observables

This section is the counterpart of 3.5, but for uniform MPSs: we show how to efficiently compute quantities of interest from a distribution parametrized by a MPS, this time a time-shift invariant one of the form (4.4): $q_A(\bar{x}) \propto \mathrm{Tr}\left[\prod_{\mathrm{t}=0}^{\mathrm{T}} \mathrm{A}(\mathrm{x}^{\mathrm{t}})\right]$.

As a preliminary step, let us define

$$Z_A = \sum_x A(x) \tag{4.15}$$

and its left l and right r dominant eigenvectors given by

$$\begin{cases} Z_A^\dagger l = \lambda l \\ Z_A r = \lambda r. \end{cases} \tag{4.16}$$

In the jargon of tensor networks, l and r are the left and right *environment*.

Assuming that these leading eigenvectors are unique, repeated application of Z_A gives

$$\lim_{T\to\infty} Z_A^T \propto r l^\dagger. \tag{4.17}$$

Notice that if they are indeed unique, then they must also be real-valued, as they correspond to the result of multiplying a generic vector v infinitely many times by Z_A. In particular, one can take v real-valued.

Single-time marginals are then obtained as

$$\begin{aligned} q_A(x^t) = &\sum_{\ldots,x^{t-1},x^{t+1},\ldots} q_A(\overline{x}) \\ &\propto \mathrm{tr}\left[Z_A Z_A \cdots Z_A A(x^t) Z_A Z_A \cdots Z_A\right] \\ &\propto \mathrm{tr}\left[r l^\dagger A(x^t) r l^\dagger\right] \\ &= \mathrm{tr}\left[l^\dagger r l^\dagger A(x^t) r\right] \\ &\propto l^\dagger A(x^t) r. \end{aligned} \tag{4.18}$$

Similarly, the joint distribution of n consecutive epochs is given by

$$q_A(x^t, x^{t+1}, \ldots, x^{t+n-1}) \propto l^\dagger \left[\prod_{\tau=t}^{t+n-1} A(x^\tau)\right] r. \tag{4.19}$$

Once n-time marginals are computed, it is straightforward to extract quantities such as single-time averages and time autocorrelations.

A completely analogous reasoning goes for statistics derived from pair beliefs which can be obtained as in the finite-T case using (3.64).

4.5 Results

We first apply the algorithm to parallel Glauber dynamics of classical spin $\sigma \in \pm 1$ variables, governed by the transition (3.10)

$$w_i^{t+1}(\sigma_i^{t+1}|\sigma_{\partial i}^t) = \frac{e^{\beta \sigma_i^{t+1}\left(\sum_{j\in\partial i} J_{ij}\sigma_j^t + h_i\right)}}{2\cosh\left[\beta\left(\sum_{j\in\partial i} J_{ij}\sigma_j^t + h_i\right)\right]}, \tag{4.20}$$

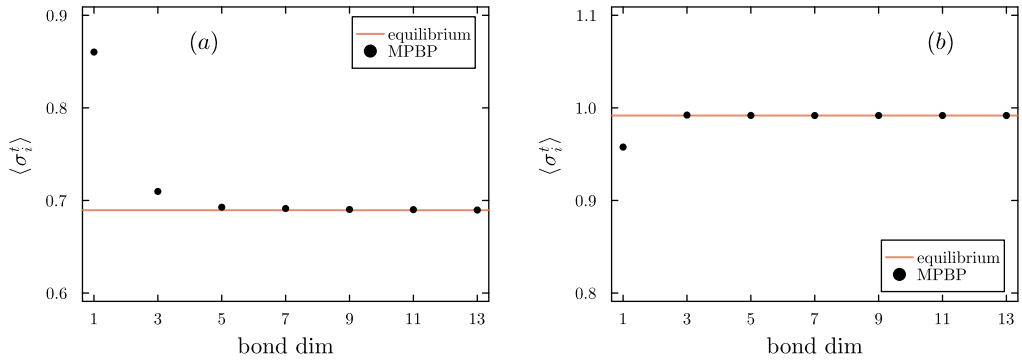


Fig. 4.1 Magnetization in Glauber dynamics on infinite Random 3-Regular Graph. (a) Paramagnetic regime $J = 0.4, h = 0.2$ (b) Ferromagnetic regime $J = 1, h = 0$. Comparison with equilibrium magnetization of the underlying Ising model.

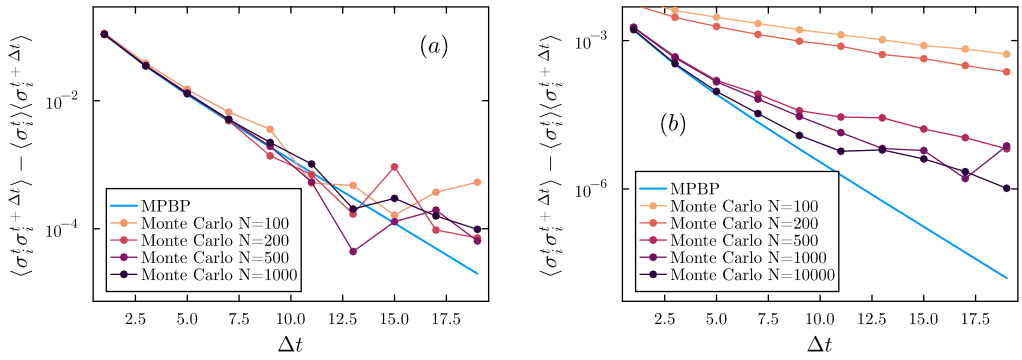


Fig. 4.2 Time autocovariance in Glauber dynamics on infinite Random Regular Graph. (a) Paramagnetic regime $J = 0.4, h = 0.2$ (b) Ferromagnetic regime $J = 1, h = 0$. Bond dimension 10.

with $J_{ij} = J_{ji} = J$, $h_i = h$, on an infinite Random Regular Graph (RRG) of degree 3. The process is known to converge to the equilibrium state of an Ising model [54, 47]. Fig. 4.1 compares the average magnetization with the equilibrium value that can be obtained via the standard cavity method. As expected, increasing the bond dimension results in improved accuracy.

Fig. 4.2 shows, for the same system, the steady-state autocovariance at distances up to $\Delta t = 20$ epochs of the dynamics. We compare with Monte Carlo estimates on increasingly large random graphs, whose accuracy degrades as the values of the autocovariance approach zero.

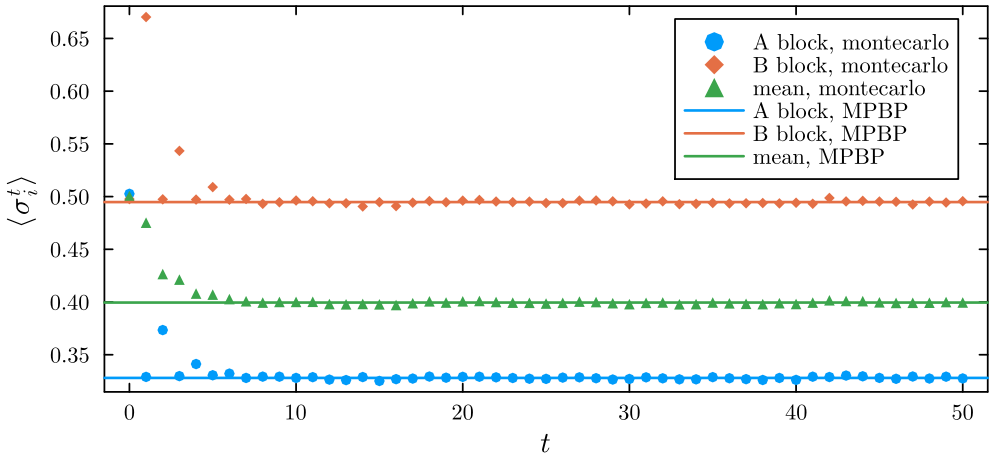


Fig. 4.3 Parallel Glauber dynamics on an infinite regular bipartite graph $G = (V = A \cup B, E)$ with non-reciprocal couplings $J_{B\to A} = 0.1, J_{A\to B} = 0.5$, external fiel $h = 0.2$, and degrees $k_A = 3, k_B = 4$. Points correspond to the transient of a Monte Carlo simulation on a random graph with $N = N_A + N_B$ vertices, $N_A = 1200, N_B = 900$. Bond dimension 5.

Next, we turn to dynamics with a non-equilibrium steady state, such as the modified Glauber dynamics with non-reciprocal interactions $J_{ij} \neq J_{ji}$. As a simple system presenting this feature, we analyze an infinite regular bipartite graph with vertices $V = A \cup B$ and edges $E \subset (ij) : i \in A, j \in B$. The non-reciprocal coupling strengths are $J_{ij} = J_{A\to B}$ if $i \in A, j \in B$, and $J_{ij} = J_{B\to A}$ otherwise. A further extent of asymmetry is added by picking different degrees for A and B vertices $k_A = 3, k_B = 4$. Fig. 4.3 shows the average magnetization for nodes in each block of the bipartition, as well as the global average. Good agreement is observed with Monte Carlo simulations on a large random graph.

Another example of dynamics whose steady state does not have a known analytical expression is the SIS model of epidemic spreading. The Markov rule for the time-discretized version of this model reads [112, 47]

$$w_i(x_i^{t+1}|\mathbf{x}_{\partial i}^t, x_i^t) = \begin{cases} \mathbb{1}\left[x_i^t = S\right] \prod_{j\in\partial i} \left(1 - \lambda \mathbb{1}\left[x_j^t = I\right]\right) + \mathbb{1}\left[x_i^t = I\right]\rho & x_i^{t+1} = S \\ 1 - \left\{\mathbb{1}\left[x_i^t = S\right] \prod_{j\in\partial i} \left(1 - \lambda \mathbb{1}\left[x_j^t = I\right]\right) + \mathbb{1}\left[x_i^t = I\right]\rho\right\} & x_i^{t+1} = I \end{cases} \tag{4.21}$$

with variables $x \in \{S, I\}$, λ and ρ the transmission and recovery probabilities. The transition probability for $x_i^{t+1} = I$ is obtained by normalization. At fixed $\rho = 0.1$ and for several values of λ, we compute the probability for a node to be infectious at the

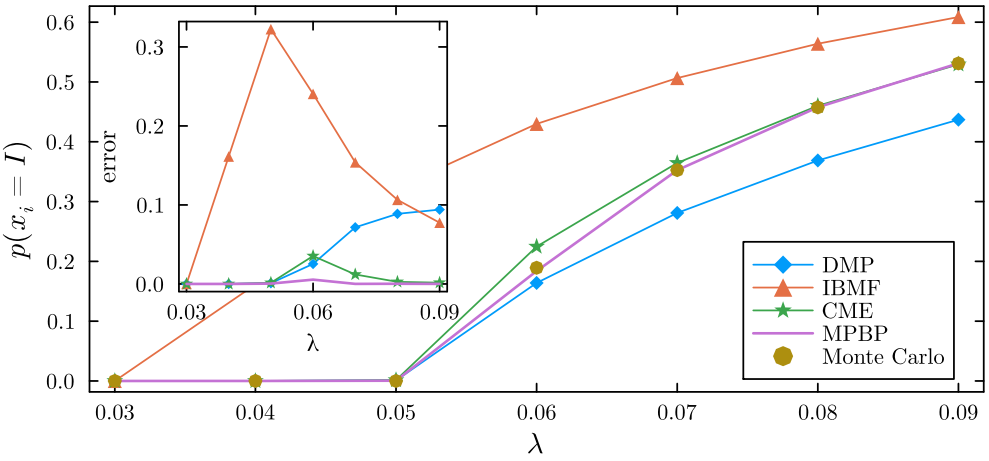


Fig. 4.4 SIS model on an infinite 3-RRG with recovery probability $\rho = 0.1$ and varying transmission probability λ. Probability of a single individual being infectious at steady state for our methods and the mean-field approaches described in the main body. Inset: absolute error $|p(x_i = I) - p_{MC}(x_i = I)|$ with respect to a Monte Carlo simulation. Monte Carlo and mean-field methods are run on a finite graph of size $N = 5 \cdot 10^3$, and for finite time horizons $T_{MC} = 4 \cdot 10^3$, $T_{MF} = 10^4$. Bond dimension 14.

steady state on an infinite 3-RRG and measure the absolute error with respect to an extensive Monte Carlo simulation. The performance is compared with the discretized version [47] three mean-field approaches: Dynamic Message Passing (DMP) [72], Independent-Based Mean Field (IBMF) [97] and Cavity Master Equation (CME) [73]. Our method achieves the best precision across the whole range of transmission probabilities.

Finally, we show how continuous-time dynamics can be approximated within our method by multiplying the parameters λ, ρ by a smaller and smaller time step Δt (see also section (3.1.2)). Fig. 4.5 shows how $\Delta t = 10^{-2}$ is enough to reproduce the steady state probability of being infectious as computed by a continuous-time Monte Carlo simulation [113]. Let us mention that, although the same procedure can and has been employed on the transient version of MPBP, in that case it comes at the cost of an increase by a factor Δt in the length of messages. The issue is removed by moving to the single-matrix ansatz.

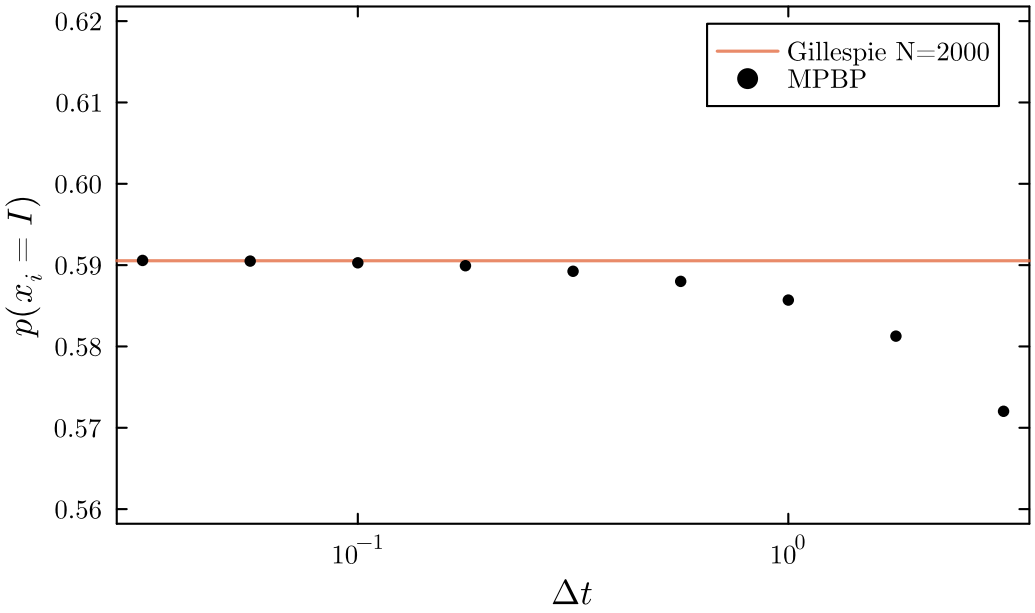


Fig. 4.5 Approximation of continuous-time SIS model on an infinite 3-RRG by rescaling the transmission and recovery probabilities $\lambda = \rho = 0.1$ by a factor Δt. Comparison with a Gillespie-like Monte Carlo simulation on a graph of size $N = 2 \cdot 10^3$.

4.6 Limitations

Accurately describing the dynamics when the system is close to a phase transition can pose some difficulties. We conjecture that this has to do with correlations in time becoming long-ranged and therefore a larger bond dimension is needed [93]. Fig. 4.6 displays the bond dimension needed to achieve a given precision on the computation of single-time marginals. In the case of Glauber dynamics on a k-regular graph, we know that the underlying Ising model in zero field exhibits a ferromagnetic transition at a critical value of coupling strength $J_C = \frac{\log \frac{k}{k-2}}{2}$ which is $J_C \approx 0.5493$ for $k = 3$. Fig. 4.6a shows how the bond dimension needed to get within 10^{-3} of the correct equilibrium magnetization grows sharply in the vicinity of J_C. This observation is consistent with finite-T results [80].

The situation is not much different when turning to the SIS model, always on a 3-regular graph. At fixed recovery probability $\rho = 0.1$, there is a critical value of transmission probability $0.5 < \lambda_C < 0.06$: under the threshold the only fixed point of the dynamics is the state where all individuals are in the susceptible state. Above λ_C, on an infinite graph, the epidemic stays sustained indefinitely. Fig. 4.6b shows

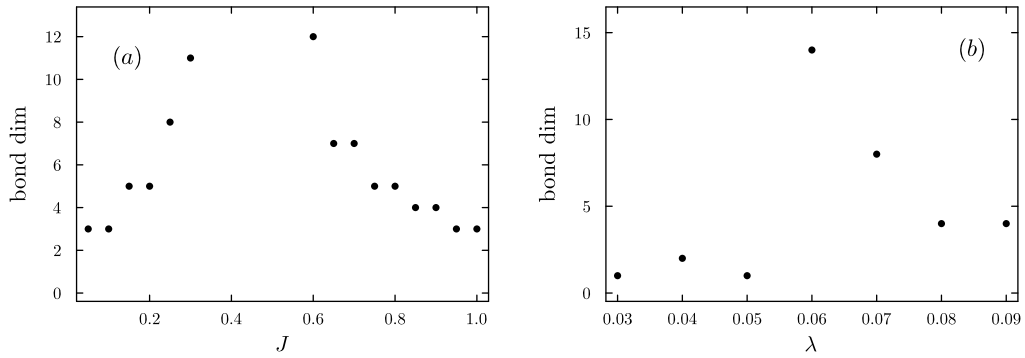


Fig. 4.6 Study of the bond dimension needed to get within a given precision on single-time, single-site observables. (a) Glauber dynamics on an infinite 3-RRG with a small symmetry-breaking external field $h = 0.1$, varying the coupling strength J. The target measure is a relative error of 10^{-3} on the single-site magnetization $\langle \sigma_i^t \rangle$ (b) SIS model on an infinite 3-RRG with recovery probability $\rho = 0.1$ and varying transmission probability λ. The target measure is a relative error of $8 \cdot 10^{-3}$ on the single-site probability of being infectious $p(x_i^t = I)$.

how the vicinity of the transition is the region of parameter space where it is the hardest to get accurate estimates.

4.7 Conclusion

In this chapter, we explored the steady-state behavior of Markov processes on graphs, building upon the Matrix Product Belief Propagation (MPBP) framework introduced in the previous chapter. We proposed a method for estimating the steady-state distribution by adapting MPBP with a time-translation-invariant ansatz, leveraging the concept of Uniform Matrix Product States (UMPS). This approach allowed us to bypass the need for simulating transient dynamics, enabling a more direct analysis of the system's long-time behavior.

Our method was successfully applied to various types of dynamics, including Glauber dynamics on infinite random graphs and epidemic spreading models such as the SIS model. In each case, the steady-state estimates produced by our algorithm were shown to match or surpass the accuracy of alternative approaches, including Monte Carlo simulations and mean-field methods. We also demonstrated the

method's ability to handle both equilibrium and non-equilibrium steady states, as well as to approximate continuous-time dynamics through appropriate time discretization.

While the proposed method offers significant computational advantages, especially for large systems and long time horizons, we also identified some limitations. Specifically, near phase transitions, where long-range time correlations become prominent, the accuracy of the method is diminished unless larger bond dimensions are used. Nonetheless, this chapter has presented a robust and efficient framework for analyzing the steady states of stochastic dynamical systems on graphs.

Future work may focus on improving the handling of phase transitions, potentially through adaptive bond dimension techniques, and further extending the applicability of this method to other types of stochastic dynamics and graph structures.

Conclusions

Chapter 5

Conclusions

This dissertation investigated the cavity method, a powerful framework originating in statistical physics, for analyzing systems of variables interacting on diluted graphs. Through a series of studies, we have demonstrated the versatility and efficacy of this approach in equilibrium and non-equilibrium contexts.

In chapter 1, the theoretical foundation of the cavity method was laid out, tracing its evolution from the Bethe-Peierls approximation to its modern algorithmic incarnations, such as belief propagation (BP). We detailed its application to the approximation of probability distributions on factor graphs, establishing a basis for subsequent chapters.

Chapter 2 focused on the Closest Vector Problem (CVP), a well-known optimization task in discrete mathematics and computer science. After mapping CVP to the minimization of the energy of a spin-glass model, the cavity method provided semi-analytical insights into the infinite-size limit and revealed phase transitions in the geometry of the space of solutions. Notably, the interplay between solution space clustering and algorithmic performance was characterized, highlighting the behavior of cavity-based algorithms in the various regimes.

In chapter 3, we turned to non-equilibrium dynamics, introducing the Matrix Product Belief Propagation (MPBP) algorithm. This novel approach is based on treating a system of interacting and time-evolving random variables as a static system whose degrees of freedom are time trajectories. It is then possible to apply the BP algorithm, combined with the matrix product approximation from quantum

mechanics, to tackle computational challenges in reweighted Markov processes, such as epidemic spreading and Glauber dynamics.

Chapter 4 extended the MPBP framework to describe the steady-state behavior of stochastic processes on graphs. Leveraging the Uniform Matrix Product State (UMPS) formalism, this method offers a promising direction for understanding steady states in systems with complex interactions.

Overall, this thesis illustrates the power of the cavity method across diverse domains, from combinatorial optimization to stochastic dynamics. The developed methodologies deepen our theoretical understanding of hard optimization problems and paradigmatic models of stochastic dynamics. Future research could explore further generalizations of the cavity method, particularly for what concerns inference and steady-state behavior in stochastic processes.

References

[1] Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.

[2] Jonathan S Yedidia, William Freeman, and Yair Weiss. Generalized belief propagation. *Advances in neural information processing systems*, 13, 2000.

[3] Hans A Bethe. Statistical theory of superlattices. *Proceedings of the Royal Society of London. Series A-Mathematical and Physical Sciences*, 150(871):552–575, 1935.

[4] Judea Pearl. Reverend bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, AAAI'82, page 133–136. AAAI Press, 1982.

[5] Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[6] Brendan J Frey and David MacKay. A revolution: Belief propagation in graphs with cycles. *Advances in neural information processing systems*, 10, 1997.

[7] David JC MacKay and Radford M Neal. Near shannon limit performance of low density parity check codes. *Electronics letters*, 33(6):457–458, 1997.

[8] Michele Leone, Andrea Montanari, Federico Ricci-Tersenghi, et al. Dynamic phase transition for decoding algorithms. *Physical Review E*, 66(4):046120, 2002.

[9] Andrea Montanari and Rudiger Urbanke. Modern coding theory: The statistical mechanics and computer science point of view. *arXiv preprint arXiv:0704.2857*, 2007.

[10] Marc Mezard, Giorgio Parisi, and Riccardo Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297:812 – 815, 2002.

[11] Roberto Mulet, Andrea Pagnani, Martin Weigt, and Riccardo Zecchina. Coloring random graphs. *Physical review letters*, 89(26):268701, 2002.

[12] Marc Mézard, Federico Ricci-Tersenghi, and Riccardo Zecchina. Two solutions to diluted p-spin models and xorsat problems. *Journal of Statistical Physics*, 111(3-4):505–533, 2003.

[13] Alfredo Braunstein, Marc Mézard, and Riccardo Zecchina. Survey propagation: An algorithm for satisfiability. *Random Structures & Algorithms*, 27(2):201–226, 2005.

[14] Florent Krzakała, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104(25):10318–10323, 2007.

[15] David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009.

[16] Andreas Engel. *Statistical mechanics of learning*. Cambridge University Press, 2001.

[17] Danny Bickson, O Shental, PH Siegel, JK Wolf, and D Dolev. Linear detection via belief propagation. In *Proc. 45th Allerton Conf. on Communications, Control and Computing*, 2007.

[18] Marc Mézard, Giorgio Parisi, and Miguel Angel Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

[19] Silvio Franz, Marc Mézard, Federico Ricci-Tersenghi, Martin Weigt, and Riccardo Zecchina. A ferromagnet with a glass transition. *Europhysics Letters*, 55(4):465, 2001.

[20] Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369, 2001.

[21] Yair Weiss and William T Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. In *Advances in neural information processing systems*, pages 673–679, 2000.

[22] Michael Chertkov. Exactness of belief propagation for some graphical models with loops. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10016, 2008.

[23] Mohsen Bayati, Devavrat Shah, and Mayank Sharma. Max-product for maximum weight matching: Convergence, correctness, and lp duality. *IEEE Transactions on Information Theory*, 54(3):1241–1251, 2008.

[24] Mohsen Bayati, Alfredo Braunstein, and Riccardo Zecchina. A rigorous analysis of the cavity equations for the minimum spanning tree. *Journal of mathematical physics*, 49(12), 2008.

[25] David Gamarnik, Devavrat Shah, and Yehua Wei. Belief propagation for min-cost network flow: Convergence and correctness. *Operations research*, 60(2):410–428, 2012.

[26] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[27] Guilhem Semerjian. On the freezing of variables in random constraint satisfaction problems. *Journal of Statistical Physics*, 130:251–293, 2008.

[28] Ryoichi Kikuchi. Superposition approximation and natural iteration calculation in cluster-variation method. *The Journal of Chemical Physics*, 60(3):1071–1080, 1974.

[29] Alessandro Pelizzola. Cluster variation method in statistical physics and probabilistic graphical models. *Journal of Physics A: Mathematical and General*, 38(33):R309, 2005.

[30] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *International Workshop on Artificial Intelligence and Statistics*, pages 308–315. PMLR, 2003.

[31] George T Cantwell and Mark EJ Newman. Message passing on networks with loops. *Proceedings of the National Academy of Sciences*, 116(47):23398–23403, 2019.

[32] Yijia Wang, Yuwen Ebony Zhang, Feng Pan, and Pan Zhang. Tensor network message passing. *Physical Review Letters*, 132(11):117401, 2024.

[33] Daniele Micciancio, Shafi Goldwasser, Daniele Micciancio, and Shafi Goldwasser. Closest vector problem. *Complexity of Lattice Problems: A Cryptographic Perspective*, pages 45–68, 2002.

[34] Miklós Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19, 1998.

[35] Daniele Micciancio and Shafi Goldwasser. Closest vector problem. In *Complexity of Lattice Problems*, pages 45–68. Springer, 2002.

[36] Uriel Feige and Daniele Micciancio. The inapproximability of lattice and coding problems with preprocessing. *Journal of Computer and System Sciences*, 69(1):45–67, 2004.

[37] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293, 1997.

[38] Claude E Shannon. Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec*, 4(142-163):1, 1959.

[39] Claude E Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.

[40] Claude E Shannon. Certain results in coding theory for noisy channels. *Information and control*, 1(1):6–25, 1957.

[41] A. Braunstein, F. Kayhan, and R. Zecchina. Efficient data compression from statistical physics of codes over finite fields. *Physical Review E*, 84(5):051111, November 2011.

[42] K. Nakamura, Y. Kabashima, and D. Saad. Statistical mechanics of low-density parity check error-correcting codes over Galois fields. *EPL*, 56(4):610, November 2001. Publisher: IOP Publishing.

[43] Silvio Franz, Michele Leone, Andrea Montanari, and Federico Ricci-Tersenghi. Dynamic phase transition for decoding algorithms. *Phys. Rev. E*, 66:046120, Oct 2002.

[44] S. Cocco, O. Dubois, J. Mandler, and R. Monasson. Rigorous decimation-based construction of ground pure states for spin-glass models on random lattices. *Phys. Rev. Lett.*, 90:047205, Jan 2003.

[45] Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on information Theory*, 47(2):585–598, 2001.

[46] Alfredo Braunstein, Farbod Kayhan, and Riccardo Zecchina. Efficient data compression from statistical physics of codes over finite fields. *Physical Review E*, 84(5):051111, 2011.

[47] Stefano Crotti and Alfredo Braunstein. Matrix product belief propagation for reweighted stochastic dynamics over graphs. *Proceedings of the National Academy of Sciences*, 120(47):e2307935120, 2023.

[48] Xiaofeng Gu, Kamesh Madduri, K. Subramani, and Hong-Jian Lai. Improved algorithms for detecting negative cost cycles in undirected graphs. In Xiaotie Deng, John E. Hopcroft, and Jinyun Xue, editors, *Frontiers in Algorithmics*, pages 40–50, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[49] Giorgio Parisi. A sequence of approximated solutions to the sk model for spin glasses. *Journal of Physics A: Mathematical and General*, 13(4):L115, 1980.

[50] Francesco Concetti. The full replica symmetry breaking in the ising spin glass on random regular graph. *Journal of Statistical Physics*, 173(5):1459–1483, 2018.

[51] Marc Mezard and Giorgio Parisi. The cavity method at zero temperature. *Journal of Statistical Physics*, 111:1–34, 2003.

[52] Adam B. Barrett and M. C. W. van Rossum. Optimal learning rules for discrete synapses. *PLoS Computational Biolology*, 4(11):e1000230, November 2008.

[53] Stefano Boccaletti, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang. Complex networks: Structure and dynamics. *Physics reports*, 424(4-5):175–308, 2006.

[54] Roy J Glauber. Time-dependent statistics of the ising model. *Journal of mathematical physics*, 4(2):294–307, 1963.

[55] Hiroki Ohta and Shin-ichi Sasa. A universal form of slow dynamics in zero-temperature random-field ising model. *Europhysics Letters*, 90(2):27008, 2010.

[56] Felix Ritort and Peter Sollich. Glassy dynamics of kinetically constrained models. *Advances in physics*, 52(4):219–342, 2003.

[57] Bernard Derrida, Martin R Evans, Vincent Hakim, and Vincent Pasquier. Exact solution of a 1d asymmetric exclusion model using a matrix formulation. *Journal of Physics A: Mathematical and General*, 26(7):1493, 1993.

[58] Vishal Sood and Sidney Redner. Voter model on heterogeneous graphs. *Physical review letters*, 94(17):178701, 2005.

[59] Fabrizio Altarelli, Alfredo Braunstein, Luca Dall'Asta, and Riccardo Zecchina. Large deviations of cascade processes on graphs. *Physical Review E*, 87(6):062115, June 2013.

[60] Paulo Shakarian, Abhivav Bhatnagar, Ashkan Aleali, Elham Shaabani, Ruocheng Guo, Paulo Shakarian, Abhinav Bhatnagar, Ashkan Aleali, Elham Shaabani, and Ruocheng Guo. The independent cascade and linear threshold models. *Diffusion in social networks*, pages 35–48, 2015.

[61] William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

[62] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925–979, 2015.

[63] Nino Antulov-Fantulin, Alen Lančić, Tomislav Šmuc, Hrvoje Štefančić, and Mile Šikić. Identification of patient zero in static and temporal networks: Robustness and limitations. *Physical review letters*, 114(24):248701, 2015.

[64] Andrey Y Lokhov, Marc Mézard, Hiroki Ohta, and Lenka Zdeborová. Inferring the origin of an epidemic with a dynamic message-passing algorithm. *Physical Review E*, 90(1):012801, 2014.

[65] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[66] Hugo Touchette. Introduction to large deviation theory: Theory, applications, simulations. *Physics Reports*, 478:1–69, 2009.

[67] Hugo Touchette. The large deviation approach to statistical mechanics. *Physics Reports*, 478(1-3):1–69, 2009.

[68] Eric Cator and Piet Van Mieghem. Second-order mean-field susceptible-infected-susceptible epidemic threshold. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 85(5):056111, 2012.

[69] Angélica S Mata and Silvio C Ferreira. Pair quenched mean-field theory for the susceptible-infected-susceptible model on complex networks. *Europhysics Letters*, 103(4):48003, 2013.

[70] Brian Karrer and Mark EJ Newman. Message passing approach for general epidemic models. *Physical Review E*, 82(1):016101, 2010.

[71] Piet Van Mieghem. The n-intertwined sis epidemic network model. *Computing*, 93(2-4):147–169, 2011.

[72] Munik Shrestha, Samuel V Scarpino, and Cristopher Moore. Message-passing approach for recurrent-state epidemic models on networks. *Physical Review E*, 92(2):022821, 2015.

[73] Ernesto Ortega, David Machado, and Alejandro Lage-Castellanos. Dynamics of epidemics from cavity master equations: Susceptible-infectious-susceptible models. *Phys. Rev. E*, 105:024308, Feb 2022.

[74] E. Aurell, G. Del Ferraro, E. Domínguez, and R. Mulet. Cavity master equation for the continuous time dynamics of discrete-spin models. *Phys. Rev. E*, 95:052119, May 2017.

[75] G. Del Ferraro and E. Aurell. Dynamic message-passing approach for kinetic spin models with reversible dynamics. *Phys. Rev. E*, 92:010102, Jul 2015.

[76] A Pelizzola and M Pretti. Variational approximations for stochastic dynamics on graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(7):073406, jul 2017.

[77] Eduardo Domínguez Vázquez, Gino Del Ferraro, and Federico Ricci-Tersenghi. A simple analytical description of the non-stationary dynamics in ising spin systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(3):033303, 2017.

[78] Isaak Neri and Désiré Bollé. The cavity approach to parallel dynamics of ising spins on a graph. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(08):P08009, 2009.

[79] Thomas Barthel, Caterina De Bacco, and Silvio Franz. Matrix product algorithm for stochastic dynamics on networks applied to nonequilibrium glauber dynamics. *Physical Review E*, 97(1):010104, 2018.

[80] Thomas Barthel. The matrix product approximation for the dynamic cavity method. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(1):013217, 2020.

[81] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac. Matrix product state representations. *Quantum Info. Comput.*, 7(5):401–430, jul 2007.

[82] Frank Verstraete and J Ignacio Cirac. Matrix product states represent ground states faithfully. *Physical review b*, 73(9):094423, 2006.

[83] Mark Fannes, Bruno Nachtergaele, and Reinhard F Werner. Finitely correlated states on quantum spin chains. *Communications in mathematical physics*, 144:443–490, 1992.

[84] Mari Carmen Bañuls and Juan P Garrahan. Using matrix product states to study the dynamical large deviations of kinetically constrained models. *Physical review letters*, 123(20):200601, 2019.

[85] Zhao-Yu Han, Jun Wang, Heng Fan, Lei Wang, and Pan Zhang. Unsupervised generative modeling using matrix product states. *Physical Review X*, 8(3):031012, 2018.

[86] Edwin Stoudenmire and David J Schwab. Supervised learning with tensor networks. *Advances in neural information processing systems*, 29, 2016.

[87] Fabrizio Altarelli, Alfredo Braunstein, Luca Dall'Asta, Alejandro Lage-Castellanos, and Riccardo Zecchina. Bayesian inference of epidemics on networks via belief propagation. *Physical review letters*, 112(11):118701, 2014.

[88] Gino Del Ferraro and Erik Aurell. Perturbative large deviation analysis of non-equilibrium dynamics. *Journal of the Physical Society of Japan*, 83(8):084001, 2014.

[89] Alfredo Braunstein, Giovanni Catania, Luca Dall'Asta, Matteo Mariani, and Anna Paola Muntoni. Inference in conditioned dynamics through causality restoration. *Scientific Reports*, 13(1):7350, 2023.

[90] Alfredo Braunstein, Giovanni Catania, Luca Dall'Asta, Matteo Mariani, Fabio Mazza, and Mattia Tarabolo. Small-coupling dynamic cavity: a bayesian mean-field framework for epidemic inference. *arXiv preprint arXiv:2306.03829*, 2023.

[91] Hendrik A Kramers and Gregory H Wannier. Statistics of the two-dimensional ferromagnet. part ii. *Physical Review*, 60(3):263, 1941.

[92] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[93] Ulrich Schollwöck. The density-matrix renormalization group in the age of matrix product states. *Annals of physics*, 326(1):96–192, 2011.

[94] Rasmus Munk Larsen. Lanczos bidiagonalization with partial reorthogonalization. *DAIMI Report Series*, (537), 1998.

[95] James Baglama and Lothar Reichel. Augmented implicitly restarted lanczos bidiagonalization methods. *SIAM Journal on Scientific Computing*, 27(1):19–42, 2005.

[96] Federico Florio. Inference of hyperparameters in agent-based dynamics. Master's thesis, Politecnico di Torino, 2024.

[97] Piet Van Mieghem, Jasmina Omic, and Robert Kooij. Virus spread in networks. *IEEE/ACM Transactions On Networking*, 17(1):1–14, 2008.

[98] Jerome Kunegis. Zachary karate club. http://konect.cc/networks/ucidata-zachary/. Accessed: 14.01.2023.

[99] Amin Coja-Oghlan, Philipp Loick, Balázs F Mezei, and Gregory B Sorkin. The ising antiferromagnet and max cut on random regular graphs. *SIAM Journal on Discrete Mathematics*, 36(2):1306–1342, 2022.

[100] Stefano Crotti, Thomas Barthel, and Alfredo Braunstein. Nonequilibrium steady-state dynamics of markov processes on graphs. *arXiv preprint arXiv:2411.19100*, 2024.

[101] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[102] Freya Behrens, Barbora Hudcová, and Lenka Zdeborová. Backtracking dynamical cavity method. *Physical Review X*, 13(3):031021, 2023.

[103] Stefano Crotti and Alfredo Braunstein. MatrixProductBP. https://github.com/stecrotti/MatrixProductBP.jl, 2023. accessed 09/10/2024.

[104] Jutho Haegeman, J Ignacio Cirac, Tobias J Osborne, Iztok Pižorn, Henri Verschelde, and Frank Verstraete. Time-dependent variational principle for quantum lattices. *Physical review letters*, 107(7):070601, 2011.

[105] Valentin Zauner-Stauber, Laurens Vanderstraeten, Matthew T Fishman, Frank Verstraete, and Jutho Haegeman. Variational optimization algorithms for uniform matrix product states. *Physical Review B*, 97(4):045145, 2018.

[106] F.R. Gantmakher. *The Theory of Matrices*. Number v. 1 in AMS Chelsea Publishing Series. Chelsea Publishing Company, 1959.

[107] Roman Orus and Guifre Vidal. Infinite time-evolving block decimation algorithm beyond unitary evolution. *Physical Review B*, 78(15):155117, 2008.

[108] Ian P McCulloch. Infinite size density matrix renormalization group, revisited. *arXiv preprint arXiv:0804.2509*, 2008.

[109] Maarten Van Damme, Lukas Devos, and Jutho Haegeman. MPSKit.jl, July 2024.

[110] Laurens Vanderstraeten, Jutho Haegeman, and Frank Verstraete. Tangent-space methods for uniform matrix product states. *SciPost Physics Lecture Notes*, page 007, 2019.

[111] Bram Vanhecke, Maarten Van Damme, Jutho Haegeman, Laurens Vanderstraeten, and Frank Verstraete. Tangent-space methods for truncating uniform mps. *SciPost Physics Core*, 4(1):004, 2021.

[112] Linda JS Allen. Some discrete-time si, sir, and sis epidemic models. *Mathematical biosciences*, 124(1):83–105, 1994.

[113] Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of computational physics*, 22(4):403–434, 1976.

[114] Matthew C Davey and David JC MacKay. Low density parity check codes over gf (q). In *1998 Information Theory Workshop (Cat. No. 98EX131)*, pages 70–71. IEEE, 1998.

[115] Alfredo Braunstein, Farbod Kayhan, and Riccardo Zecchina. Efficient LDPC codes over GF(q) for lossy data compression. In *IEEE International Symposium on Information Theory, 2009. ISIT 2009*, Seul, Korea, 2009.

[116] P Peretto. Collective properties of neural networks: a statistical physics approach. *Biological cybernetics*, 50(1):51–62, 1984.

Appendix A

Equilibrium

A.1 Lossy compression and channel coding

The problem of channel coding is formally identical to source coding in the sense that it is stated as the minimization of the same energy function (2.14) (see e.g. [42, equation 3]. However, there are two main differences that make one problem substantially harder than the other. The first difference is a design choice: a set of codewords which is good for channel coding is typically awful for source coding. In the communication problem one wants codewords to be as far as possible from one another, while for compression one wants them covering the whole space, so that any point is close enough to at least one codeword. The second difference lies in the distribution of the source vector. While here we are taking every bit of the source uniformly at random, in channel coding the source vector is made of a codeword with a small fraction of corrupted bits. This means that, provided the code is well-designed, there will always be one codeword significantly closer to the source than others, making the problem typically easier.

The results of the two problems are not directly comparable, however we give in Fig. A.1 some evidence that the compression problem becomes easy when the source vector is close enough to a codeword. We place ourselves in a regime where the compression problem with symmetric source is in a 1RSB phase, but modify the source starting from a codeword and then flipping a fraction ε of the bits. The compression is then performed by max-sum with reinforcement. For small values of ε, max-sum is able to solve the problem exactly, as the number of unsatisfied factors is zero and the distortion is equal to ε. Indeed, by construction, ε is the (normalized) Hamming distance from the starting codeword. By definition, the closest codeword will always be at a distance smaller or equal than ε. As ε increases, we observe a transition where the algorithm stops converging to solutions, and the vectors found (after fixing a set of independent variables as described in section 2.5.2) have a distortion which is even worse than the 1RSB prediction for the same graph (for a symmetric source).

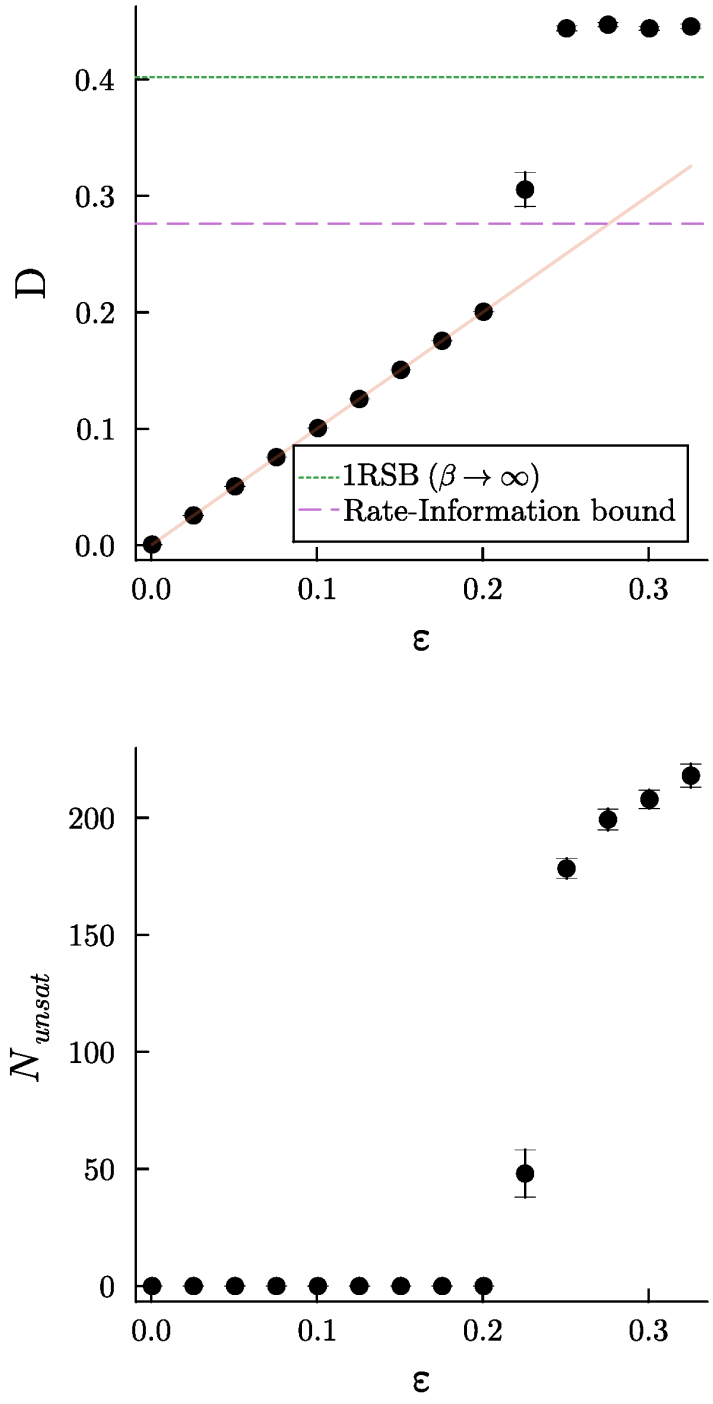


Fig. A.1 Performance of max-sum with reinforcement on source vectors made by perturbing the all-zeros codeword with εn bit flips. $n = 1800, \Lambda = \{\lambda_2 = 0.45, \lambda_3 = 0.55\}, P = \{p_3 = 1\}, R = 0.15$, average over 50 instances. The upper panel shows the average distortion, with the Shannon bound and 1RSB prediction for a symmetric source as references. The lower panel shows the number of unsatisfied factors.

A.2 Generalization to GF(q)

As shown in the context of channel coding [114] and source coding [115], distortion performance can be improved by working with variables in the finite field GF(q) with $q = 2^k, k \in \mathbb{N}$, as a generalization of binary numbers. In this case the parity check matrix H has elements $H_{ai} \in \{0, 1, \dots, q-1\}$, and similarly for the source vector $\underline{y}$.

The Boltzmann distribution (2.13) becomes:

$$\mu(\underline{x}) = \frac{1}{Z} \prod_{a=1}^{m} \mathbb{I}\left[\bigoplus_{j \in \partial a} H_{aj} x_j = 0 \right] \exp\left(-\beta \sum_{i=1}^{n} d_H(x_i, s_i) \right) \tag{A.1}$$

where by $\oplus$ we indicate the XOR function, which applies point-wise to the binary representation of GF(2^k) numbers. The Hamming distance $d_H(x,y)$ between two numbers in GF(2^k) is the number of ones in the binary representation of $x \oplus y$. The operation $H_{aj}x_j$ is intended as the GF(2^k) multiplication.

The max-sum equations in GF(q) read:

$$\begin{aligned} u_{a\to i}(x_i) &= \max_{\underline{x}_{a\setminus i} : \bigoplus_{k\in\partial a} H_{ak} x_k = 0} \left[\sum_{j \in \partial a \setminus i} h_{j\to a}(x_j) \right] - \widehat{C}_{a\to i} \\ h_{i\to a}(x_i) &= \sum_{b \in \partial i \setminus a} u_{b\to i}(x_i) - d_H(x_i, y_i) - C_{i\to a} \end{aligned} \tag{A.2}$$

where $C_{i\to a}$ and $\widehat{C}_{a\to i}$ are constants ensuring respectively that $\max_{x_i} h_{i\to a}(x_i) = 0$ and $\max_{x_i} u_{a\to i}(x_i) = 0$.

For a source vector $\underline{y} \in \{0,1\}^n$, the corresponding GF$(2^k)$ counterpart is obtained by grouping bits in groups of k and interpreting them as binary code. Of course vector sizes must be adjusted as to produce integer values. Likewise, given a solution for the problem on GF(2^k), one unwraps each element into k bits and concatenates them to obtain a binary vector.

Figure A.2 shows the results of max-sum algorithm with reinforcement, plotted against the Replica-Symmetric prediction. Increasing q, one observes indeed that the average minimal distortion decreases. The results of max-sum and the RS prediction starts to differ when q increases, especially at large rate R. This phenomenon was

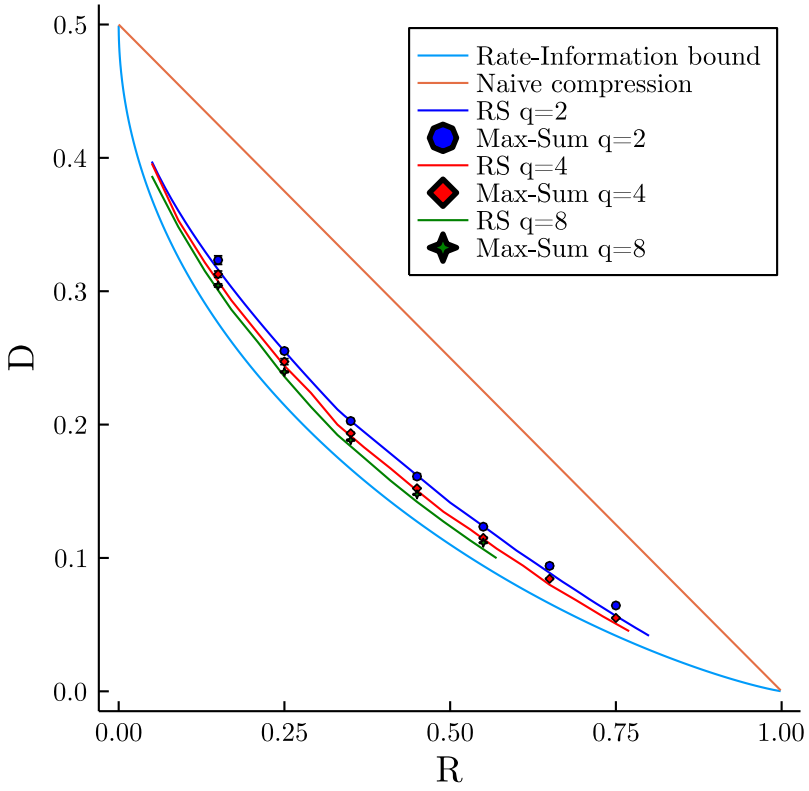


Fig. A.2 Rate-distortion performance for the GF(q) max-sum algorithm on codes of $n = 6720$ bits (points), plotted against the Replica-Symmetric prediction (plain lines). The degree profile is $\Lambda(x) = \{\lambda_2 = 1\}$, $P(x) = \{p_k, p_{k+1}\}$. Points are the average over 10 random graphs and source vectors. For each instance, max-sum was run from 5 random initializations of messages, the one that gave the minimum number of unsatisfied constraints was kept

already encountered for $q = 2$ (see main text, section 2.5.2 in particular Fig. 2.4.). We leave for future work the investigation of this discrepancy, as already argued in section 2.5.2 it is possible that the zero-temperature cavity method is not correct in this regime. Another possibility is the apparition of a 1RSB or full RSB phase transition as q, R increases for cycle codes.

A.3 Details of the max-sum equations for CVP

This section provides details of the calculation of factor-to-variable max-sum messages for the CVP

$$u_{ai}\sigma_i = \max_{\underline{\sigma}_{a\setminus i}} \left[J\sigma_i \prod_{j\in\partial a\setminus i} \sigma_j + \sum_{j\in\partial a\setminus i} h_{ja}\sigma_j \right]. \tag{A.3}$$

It is useful to notice that the set of configurations for the neighboring spins $\underline{\sigma}_{a\setminus i}$ can be split according to their product: $\underline{\sigma}_{a\setminus i} = \{\sigma_j | j \in \partial a \setminus i, \prod_j \sigma_j = \sigma_i\} \cup \{\sigma_j | j \in \partial a \setminus i, \prod_j \sigma_j = -\sigma_i\}$. Therefore

$$u_{ai}\sigma_i = \max \left\{ J + \max_{\prod_j \sigma_j = \sigma_i} \sum_j h_{ja}\sigma_j,\ -J + \max_{\prod_j \sigma_j = -\sigma_i} \sum_j h_{ja}\sigma_j \right\} \tag{A.4}$$

.

Taking a closer look to the inner max, there are two cases: it can be that the signs of the incoming fields have the same parity (sign) as σ_i, in which case the max is obtained by aligning each spin to its field. Otherwise, one must pick one variable for which the spin must take opposite sign with respect to the field: it must be the one which gives the smallest loss, i.e. $\min_j |h_{ja}|$

$$\begin{aligned} \max_{\prod_j \sigma_j = \sigma_i} \sum_j h_{ja}\sigma_j &= \begin{cases} \sum_j |h_{ja}| & \prod_j \operatorname{sign}(h_{ja}) = \sigma_i \\ \sum_j |h_{ja}| - 2\min_j |h_{ja}| & \text{otherwise} \end{cases} \\ &= \sum_j |h_{ja}| - 2\Theta\left(-\sigma_i \prod_j h_{ja} \right) \min_j |h_{ja}|. \end{aligned} \tag{A.5}$$

where Θ is the step function $\Theta(x) = \mathbb{1}[x > 0]$.

Similarly,

$$\max_{\prod_j \sigma_j = -\sigma_i} \sum_j h_{ja}\sigma_j = \sum_j |h_{ja}| - 2\Theta\left(\sigma_i \prod_j h_{ja} \right) \min_j |h_{ja}|. \tag{A.6}$$

Now (A.4) becomes

$$u_{ai}\sigma_i = \sum_j |h_{ja}| + \max\left\{ J - 2\Theta\left(-\sigma_i \prod_j h_{ja}\right) \min_j |h_{ja}|,\ -J - 2\Theta\left(\sigma_i \prod_j h_{ja}\right) \min_j |h_{ja}| \right\}. \tag{A.7}$$

To get u_{ai},

$$\begin{aligned}
u_{ai} &= \frac{1}{2}\left(u_{ai}(1) - u_{ai}(-1)\right) \\
&= \frac{1}{2}\max\left\{ J - 2\Theta\left(-\prod_j h_{ja}\right) \min_j |h_{ja}|,\ -J - 2\Theta\left(\prod_j h_{ja}\right) \min_j |h_{ja}| \right\} \\
&- \frac{1}{2}\max\left\{ J - 2\Theta\left(\prod_j h_{ja}\right) \min_j |h_{ja}|,\ -J - 2\Theta\left(-\prod_j h_{ja}\right) \min_j |h_{ja}| \right\} \\
&= \frac{1}{2}\Theta\left(-\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\max\left\{J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, -J\right\} - \max\left\{J, -J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|\right\}\right) \\
&+ \frac{1}{2}\Theta\left(+\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\max\left\{J, -J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|\right\} - \max\left\{J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, -J\right\}\right) \\
&=^{*} \frac{1}{2}\operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\max\left\{J, -J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|\right\} - \max\left\{J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, -J\right\}\right) \\
&= \frac{1}{2}\operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\max\left\{J, -J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|\right\} - \max\left\{J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, -J\right\}\right) \\
&=^{**} \frac{1}{2}\operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(J - \max\left\{J - 2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, -J\right\}\right) \\
&= \frac{1}{2}\operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(J + \min\left\{2\min_{j\in\partial a\setminus i}\left|h_{ja}\right| - J, J\right\}\right) \\
&= \frac{1}{2}\operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\min\left\{2\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, 2J\right\}\right) \\
&= \operatorname{sign}\left(\prod_{j\in\partial a\setminus i} h_{ja}\right)\left(\min\left\{\min_{j\in\partial a\setminus i}\left|h_{ja}\right|, J\right\}\right)
\end{aligned} \tag{A.8}$$

where at * we used the fact that $\operatorname{sign}(x) = \Theta(x) - \Theta(-x)$ and at ** the fact that $J > 0$.

A.4 Simplifications for the survey propagation equations

We report here simplifications that reduce the computational cost of the SP equations (2.79).

First SP equation For a generic factor degree $k+1$, the first equation reads

$$P(u) = \sum_{h_1,\dots,h_k} \delta\left(u, \min_{j=1,\dots,k} |h_j| \prod_{j=1}^{k} \operatorname{sign}(h_j)\right) \prod_{j=1}^{k} Q_j(h_j). \tag{A.9}$$

Since it looks just like the first equation in (2.59), one might be tempted to directly write down the same simplifications. However, notice that here the symmetry assumptions $p(u) = p(-u)$ and $q(h) = q(-h)$ do not hold anymore. What's more, at difference with (2.59), here the incoming messages need not be equal.

We first consider the case $u = 0$

$$\begin{aligned} P(0) &= \mathbb{P}\left(\text{at least one } h_j = 0\right) \\ &= 1 - \prod_{j=1}^{k} \left(1 - Q_j(0)\right). \end{aligned} \tag{A.10}$$

For $u \neq 0$ the trick is to write down the set over which the sum is performed as the difference of two other sets that are easier to treat:

$$\begin{aligned} A(u) &:= \left\{ h_1,\dots,h_k : \min_{j=1,\dots,k} |h_j| = |u|, \prod_{j=1}^{k} \operatorname{sign}(h_j) = \operatorname{sign}(u) \right\} \\ &= \left\{ |h_1|,\dots |h_k| \geq |u|, \prod_{j=1}^{k} \operatorname{sign}(h_j) = \operatorname{sign}(u) \right\} \setminus \left\{ |h_1|,\dots |h_k| > |u|, \prod_{j=1}^{k} \operatorname{sign}(h_j) = \operatorname{sign}(u) \right\} \\ &= \left\{ |h_1|,\dots |h_k| \geq |u|, \prod_{j=1}^{k} \operatorname{sign}(h_j) = \operatorname{sign}(u) \right\} \\ &\quad \setminus \left\{ |h_1|,\dots |h_k| \geq |u + \operatorname{sign}(u)|, \prod_{j=1}^{k} \operatorname{sign}(h_j) = \operatorname{sign}\left(u + \operatorname{sign}(u)\right) \right\} \end{aligned} \tag{A.11}$$

where we used the fact that $|h| > |u| \Rightarrow |h| \geq |u|+1 = |u+\text{sign}(u)|$ and $\text{sign}(u) = \text{sign}(u+\text{sign}(u))$. We can use this fact to build a recursion

$$\begin{aligned} a_n(u) &= \sum_{\substack{|h_1|,\dots|h_n|\geq|u| \\ \prod_{i=1}^n \text{sign}(h_i)=\text{sign}(u)}} \prod_{i=1}^n Q_i(h_i) \\ &= \sum_{|h_n|\geq|u|} Q_n(h_n) a_{n-1}\left(u\,\text{sign}(h_n)\right) \\ &= a_{n-1}(u) \sum_{h_n \geq u} Q_n(h_n) + a_{n-1}(-u) \sum_{h_n \leq -u} Q_n(h_n) \end{aligned} \tag{A.12}$$

with initial condition $a_0(u) = \delta\left(1-\text{sign}(u)\right)$. Finally,

$$P(u) = \begin{cases} a_k(u) - a_k(u+\text{sign}(u)) & u \neq 0 \\ 1 - \prod_{j=1}^k \left(1-Q_j(0)\right) & u = 0 \end{cases} \tag{A.13}$$

where the quantities a_k depend on the incoming Q's through (A.12).

Second SP equation For a generic variable node of degree $d+1$, the second SP equation reads

$$Q(h) = e^{y|h|} \sum_{u_1,\dots,u_d} \delta\left(h, s - \sum_{b=1,\dots,d} u_b\right) \prod_{b=1,\dots,d} P_b(u_b) e^{-y|u_b|}. \tag{A.14}$$

We have

$$Q(h) = e^{y|h|} q_d(h) \tag{A.15}$$

where q_d satisfies the recursion

$$q_b(h) = \sum_{u_b} f_b(u_b) q_{b-1}(h-u_b) \tag{A.16}$$

where $f_b(u_b) \coloneqq P_b(u_b) e^{-y|u_b|}$ and with initial condition ensuring the shift by s

$$q_0(h) = \delta(h-s). \tag{A.17}$$

Finally,

$$Q(h) = e^{y|h|} q_0 \circledast f_1 \circledast \cdots \circledast f_d \tag{A.18}$$

where we recall that $\circledast$ means convolution.

Survey As in the RS case, the survey update is very much similar to the one for Q

$$B(h) = e^{y|h|} q_0 \circledast f_1 \circledast \cdots \circledast f_{d+1}. \tag{A.19}$$

A.5 Simplification for the overlap computed by survey propagation

We report here simplification to the expression for the average overlap as computed by survey propagation through equations (2.82), (2.83).

Edge contribution

$$\begin{aligned}\langle F_{ia}(h_{ia},u_{ai})\rangle_{B_{ia}} &= \frac{1}{Z_{ia}}\sum_{u_{ai},h_{ia}} F_{ia}(h_{ia},u_{ai})e^{-yF_{ia}(h_{ia},u_{ai})}Q_{ia}(h_{ia})P_{ai}(u_{ai}) \\ Z_{ia} &= \sum_{u_{ai},h_{ia}} e^{-yF_{ia}(u_{ai},h_{ia})}Q_{ia}(h_{ia})P_{ai}(u_{ai})\end{aligned} \tag{A.20}$$

Variable contribution Adopting a lighter notation $a = 1,\dots,d \leftrightarrow a \in \partial i$ where $|\partial i| = d$ and $s_i \equiv s$, we have

$$\begin{aligned}Z_i &= \sum_{u_1,\dots,u_d} e^{-yF_i(u_1,\dots,u_d)}\prod_{a=1}^{d} P_a(u_a) \\ &= \sum_{u_1,\dots,u_d} e^{y\left(\left|s+\sum_{a=1}^{d}u_a\right|-\sum_{a=1}^{d}|u_a|\right)}\prod_{a=1}^{d} P_a(u_a) \\ &=^{*} \sum_h e^{y|h|}\sum_{u_1,\dots,u_d}\delta\left(h-s-\sum_a u_a\right)\prod_{a=1}^{d} P_a(u_a)e^{-y|u_a|} \\ &=^{**} \sum_h e^{y|h|}q_d(h)\end{aligned} \tag{A.21}$$

where at * we introduced an identity and ** is analogous to the steps leading to (A.18):

$$\begin{aligned}q_a(h) &= \sum_{u_a} P_a(u_a)e^{-y|u_a|}q_{a-1}(h-u_a) \\ q_0(h) &= \delta(h-s)\end{aligned} \tag{A.22}$$

Thus,

$$\begin{aligned}
\langle F_i(u_1,\ldots,u_d)\rangle_{B_i} =& \frac{1}{Z_i}\sum_{u_1,\ldots,u_d} F_i(u_1,\ldots,u_d)e^{-yF_i(u_1,\ldots,u_d)}\prod_{a=1}^{d}P_a(u_a)\\
=& \frac{1}{Z_i}\sum_{u_1,\ldots,u_d}\left(-\left|s+\sum_{a=1}^{d}u_a\right|+\sum_{a=1}^{d}|u_a|\right)e^{y\left(\left|s+\sum_{a=1}^{d}u_a\right|-\sum_{a=1}^{d}|u_a|\right)}\prod_{a=1}^{d}P_a(u_a)\\
=& \frac{1}{Z_i}\sum_h e^{y|h|}\sum_{u_1,\ldots,u_d}\delta\left(h-s-\sum_a u_a\right)\left(-|h|+\sum_{a=1}^{d}|u_a|\right)\prod_{a=1}^{d}P_a(u_a)e^{-y|u_a|}\\
=& -\frac{1}{Z_i}\sum_h e^{y|h|}|h|\sum_{u_1,\ldots,u_d}\delta\left(h-s-\sum_a u_a\right)\prod_{a=1}^{d}P_a(u_a)e^{-y|u_a|}\\
&+\frac{1}{Z_i}\sum_h e^{y|h|}\sum_{u_1,\ldots,u_d}\delta\left(h-s-\sum_a u_a\right)\sum_{b=1}^{d}|u_b|\prod_{a=1}^{d}P_a(u_a)e^{-y|u_a|}\\
=^{*}& -\frac{1}{Z_i}\sum_h e^{y|h|}|h|q_d(h)+\frac{1}{Z_i}\sum_h e^{y|h|}\sum_{b=1}^{d}\sum_{u_1,\ldots,u_d}\delta\left(h-s-\sum_a u_a\right)\\
&\times\prod_{a=1}^{d}P_a(u_a)e^{-y|u_a|}|u_b|
\end{aligned} \tag{A.23}$$

where at * we again used the recursion on the q_b's just like for Z_i. The second term is similar except for the additional sum.

In summary,

$$\begin{aligned}
\langle F_i(u_1,\ldots,u_d)\rangle_{B_i} =& -\frac{1}{Z_i}\sum_h e^{y|h|}|h|q_d(h)+\frac{1}{Z_i}\sum_h e^{y|h|}\sum_{b=1}^{d}\sum_{u_1,\ldots,u_d}\delta\left(h-s-\sum_a u_a\right)\\
&\times\prod_{a=1}^{d}P_a(u_a)e^{-y|u_a|}|u_b|\\
Z_i =& \sum_h e^{y|h|}q_d(h)
\end{aligned} \tag{A.24}$$

Factor contribution Adopting a lighter notation $i = 1,\dots,k \leftrightarrow i \in \partial a$ where $|\partial a| = k$, we have

$$
\begin{aligned}
Z_a &= \sum_{h_1,\dots,h_k} e^{-yF_a(h_1,\dots,h_k)} \prod_{i=1}^{k} Q_i(h_i) \\
&= \sum_{h_1,\dots,h_k} e^{-y\Theta\left(-\prod_{i=1}^{k} h_i\right) 2\min_{i=1,\dots,k}|h_i|} \prod_{i=1}^{k} Q_i(h_i) \\
&= \sum_{h_1,\dots,h_k} \left\{ \Theta\left(-\prod_{i=1}^{k} h_i\right) e^{-y2\min_{i=1,\dots,k}|h_i|} + \left[1 - \Theta\left(-\prod_{i=1}^{k} h_i\right)\right] \right\} \prod_{i=1}^{k} Q_i(h_i) \\
&= 1 + \sum_{h_1,\dots,h_k} \left\{ \Theta\left(-\prod_{i=1}^{k} h_i\right) \left(e^{-y2\min_{i=1,\dots,k}|h_i|} - 1\right) \right\} \prod_{i=1}^{k} Q_i(h_i) \\
&=^{*} 1 + \sum_{u=0}^{\infty} \left(e^{-2yu} - 1\right) \sum_{\substack{h_1,\dots,h_k \\ \min_{i=1,\dots,k} h_i = u \\ \prod_{i=1}^{k} h_i = -1}} \prod_{i=1}^{k} Q_i(h_i) \\
&=^{**} 1 + \sum_{u=1}^{\infty} \left(e^{-2yu} - 1\right) \left(a_k(-u) - a_k(-u-1)\right)
\end{aligned}
\tag{A.25}
$$

where at * we introduced an identity and ** uses the same recursion as (A.13) and the sum might as well start at $u = 1$.

With similar reasoning one gets

$$
\begin{aligned}
\langle F_a(\{h_{ia}\})\rangle_{B_a} =& \frac{1}{Z_a}\sum_{h_1,\ldots,h_k} F_a(h_1,\ldots,h_k)e^{-yF_a(h_1,\ldots,h_k)}\prod_{i=1}^{k}Q_i(h_i)\\
=& \frac{1}{Z_a}\sum_{h_1,\ldots,h_k}\Theta\left(-\prod_{i\in\partial a}h_{ia}\right)2\min_{i\in\partial a}|h_{ia}|\,e^{-y\Theta(-\prod_{i\in\partial a}h_{ia})2\min_{i\in\partial a}|h_{ia}|}\prod_{i=1}^{k}Q_i(h_i)\\
=& \frac{1}{Z_a}\sum_{h_1,\ldots,h_k}\left\{\Theta\left(-\prod_{i\in\partial a}h_{ia}\right)2\min_{i\in\partial a}|h_{ia}|\right.\\
&\left.\times\left[1-\Theta\left(-\prod_{i\in\partial a}h_{ia}\right)+\Theta\left(-\prod_{i\in\partial a}h_{ia}\right)e^{-y2\min_{i\in\partial a}|h_{ia}|}\right]\prod_{i=1}^{k}Q_i(h_i)\right\}\\
=^{*}& \frac{1}{Z_a}\sum_{h_1,\ldots,h_k}\left\{\Theta\left(-\prod_{i\in\partial a}h_{ia}\right)2\min_{i\in\partial a}|h_{ia}|\,e^{-y2\min_{i\in\partial a}|h_{ia}|}\prod_{i=1}^{k}Q_i(h_i)\right\}\\
=^{**}& \frac{1}{Z_a}\sum_{u=0}^{\infty}2ue^{-2yu}\sum_{\substack{h_1,\ldots,h_k\\ min_i|h_i|=u\\ \prod_i\mathrm{sign}(h_i)=-1}}\prod_{i=1}^{k}Q_i(h_i)\\
=^{***}& \frac{1}{Z_a}\sum_{u=1}^{\infty}2ue^{-2yu}\left(a_k(-u)-a_k(-u-1)\right)
\end{aligned}
\tag{A.26}
$$

where * is the usual delta trick, ** uses $\Theta^2\equiv\Theta$ and *** again invokes the a_i recursion and the sum might as well start from 1.

In summary:

$$
\begin{aligned}
\langle F_a(\{h_{ia}\})\rangle_{B_a} =& \frac{1}{Z_a}\sum_{u=1}^{\infty}2ue^{-2yu}\left(a_k(-u)-a_k(-u-1)\right)\\
Z_a =& 1+\sum_{u=1}^{\infty}\left(e^{-2yu}-1\right)\left(a_k(-u)-a_k(-u-1)\right).
\end{aligned}
\tag{A.27}
$$

Appendix B

Nonequilibrium

B.1 Parallel Glauber dynamics and equilibrium

B.1.1 Marginals and correlations in Parallel Glauber dynamics

It is well known that (fully symmetric) Glauber dynamics with asynchronous update converges to the equilibrium distribution for the underlying Ising model on a graph $G = (V, E)$ [54] (a fact that can be verified by checking the detailed balance condition)

$$p^{eq}(\sigma) \propto \exp\left\{\beta\left[\sum_{(ij)\in E} J_{ij}\sigma_i\sigma_j + \sum_{i=1}^{N} h_i\sigma_i\right]\right\}. \tag{B.1}$$

Parallel updates like the ones considered in this work, instead, lead to a stationary distribution [116]

$$p^{stat}(\sigma) \propto \exp\left\{\sum_i \left[\log\cosh\beta\left(\sum_{j\in\partial i} J_{ij}\sigma_j + h_i\right) + \beta h_i\sigma_i\right]\right\}. \tag{B.2}$$

Here we show that, provided that the underlying model lives on a bipartite graph:

1. The two distributions have the same marginals, i.e. $p^{eq}(\sigma_i) = p^{stat}(\sigma_i)$.
2. The joint distribution for neighboring variables $p^{eq}(\sigma_i, \sigma_j)$ is equal to $p(\sigma_i^{t+1}, \sigma_j^t)$ where σ^t, σ^{t+1} are configurations sampled using the parallel Glauber update at the stationary state.

To see why the two propositions are true, consider an augmented system $\tilde{G} = (\tilde{V}, \tilde{E})$ consisting of two copies of the vertices of the original graph. The new system is made of $2N$ variables $\{\sigma_1, \ldots, \sigma_N, \sigma'_1, \ldots, \sigma'_N\}$. Each variable σ_i interacts with the copies of its neighbors in the original graph $\{\sigma'_j\}_{j:(ij)\in E}$, and vice-versa. The new system is distributed according to

$$p^{aug}(\sigma, \sigma') \propto \exp\left\{\beta\sum_{i=1}^{N}\left[\sum_{j:(ij)\in E} J_{ij}\sigma_i\sigma'_j + \sum_i h_i(\sigma_i + \sigma'_i)\right]\right\}. \tag{B.3}$$

By marginalizing over σ or σ', it is easy to see that either subset is distributed according to p^{stat}. Moreover, because the original graph G was bipartite ($V = A \cup B, A \cap B = \emptyset$), the new graph $\tilde{G}$ is made of two disconnected components: the

first contains variables $\{\sigma_i\}_{i\in A}\cup\{\sigma'_j\}_{j\in B}$, the second the other half. By construction, the two subsets of variables corresponding to the two components are distributed independently and each according to p^{eq}. Without loss of generality, take $i\in A$. Since the two sets $\{\sigma_i\}_{i\in V}$ and $\{\sigma_i\}_{i\in A}\cup\{\sigma'_j\}_{j\in B}$, follow the same distribution, in particular they share the same marginal for the set $\{i\}\cup\partial i$, i.e. $p^{eq}(\sigma_i,\sigma_{\partial i}=\sigma'_{\partial i})=p^{aug}(\sigma_i,\sigma'_{\partial i})=p^{stat}(\sigma'_{\partial i})p^{aug}(\sigma_i|\sigma'_{\partial i})$. Marginalizing over the neighbors, one sees that $p^{aug}(\sigma_i)=p^{stat}(\sigma_i)=p^{eq}(\sigma_i)$, thereby proving the first claim. Moreover, $p^{aug}(\sigma_i|\sigma'_{\partial i})=\tilde{w}(\sigma_i^{t+1}=\sigma_i|\sigma^t_{\partial i}=\sigma'_{\partial i})$, the transition (3.10). By marginalizing over all neighbors but j, one obtains that i and j at two subsequent steps of the dynamics follow the equilibrium distribution, proving the second claim. As acyclic graphs are bipartite, these results hold for any acyclic graph, including the infinite size limits of Erdos-Renyi and Random Regular graphs considered in the article.

Note that the bipartiteness of G is not a serious restriction. Indeed, given an arbitrary graph G, possibly non bipartite, one can design a parallel dynamics converging to p^{eq} by considering an associated bipartite graph G' which is constructed from G as follows: for every edge (i,j), add a new spin σ_{ij} and replace (i,j) by a couple of edges $(i,(ij)),((ij),j)$ connected to ij with couplings $J_{i,ij}=+\infty, J_{ij,j}=J_{ij}$ (or alternatively, $J_{i,ij}=\tanh^{-1}\left[\sqrt{\tanh(|J_{ij}|)}\right], J_{ij,j}=J_{i,ij}\,\mathrm{sign}(J_{ij})$). Marginalizing over the extra spins $\{\sigma_{ij}\}$ one recovers the original p^{eq} and the new graph G' is clearly bipartite.

B.1.2 Self-coupling

A way of obtaining the equilibrium distribution of a given Ising Hamiltonian that is alternative to the $p_0\to\infty$ limit of (3.13) is given by self-couplings. One can enrich the dynamics by adding a coupling J_{ii} between a spin and itself at the successive epoch. The transition becomes

$$\tilde{w}(\sigma_i^{t+1}|\sigma^t_{\partial i},\sigma^t_i)=\frac{e^{\beta\sigma_i^{t+1}\left(\sum_{j\in\partial i}J_{ij}\sigma_j^t+J_{ii}\sigma_i^t+h_i\right)}}{2\cosh\left[\beta\left(\sum_{j\in\partial i}J_{ij}\sigma_j^t+J_{ii}\sigma_i^t+h_i\right)\right]} \tag{B.4}$$

and the stationary distribution

$$p^{stat}(\sigma)\propto\exp\left\{\sum_i\left[\log\cosh\beta\left(\sum_{j\in\partial i}J_{ij}\sigma_j+J_{ii}\sigma_i+h_i\right)+\beta h_i\sigma_i\right]\right\}. \tag{B.5}$$

In the limit $J_{ii} \gg 1$, one gets

$$\log\cosh\beta\left(\sum_{j\in\partial i} J_{ij}\sigma_j + J_{ii}\sigma_i + h_i\right) = \sum_{j\in\partial i} J_{ij}\sigma_i\sigma_j + J_{ii} + h_i\sigma_i + \mathscr{O}(e^{-J_{ii}}) \tag{B.6}$$

and the stationary distribution becomes

$$p^{stat}(\sigma) \propto \exp\left\{2\beta\sum_i\left[\frac{1}{2}\sum_{j\in\partial i} J_{ij}\sigma_i\sigma_j + h_i\sigma_i\right]\right\}. \tag{B.7}$$

By comparison with (B.1), we see that the resulting distribution is that of an Ising model at equilibrium at double the inverse temperature.

B.2 Models that admit tensor train representation

We show examples of models which can be represented exactly by a tensor train or matrix product state (MPS).

Models with mass on a finite support Any arbitrary distribution $p(\bar{x}) = p(x^0, x^1, \ldots, x^T)$ can in principle be represented via a MPS, albeit with bond dimension exponentially large in T: to see this, re-write p trivially as a superposition of delta distributions

$$p(\bar{x}) = \sum_{\bar{y}} p(\bar{y}) \prod_{t=0}^{T} \delta\left(x^t, y^t\right) \tag{B.8}$$

where the product over t is interpreted as a product of 1×1 matrices. Since the linear combination of two MPSs is itself a MPS [92]:

$$a\prod_t A^t(x^t) + b\prod_t B^t(x^t) = \prod_t C^t(x^t) \tag{B.9}$$

with

$$C^0(x^0) = \begin{bmatrix} aA^0(x^0) & bB^0(x^0) \end{bmatrix}, \quad C^t(x^t) = \begin{bmatrix} A^t(x^t) & 0 \\ 0 & B^t(x^t) \end{bmatrix}, \quad C^T(x^T) = \begin{bmatrix} A^T(x^T) \\ B^T(x^T) \end{bmatrix} \tag{B.10}$$

then p can be expressed by a MPS with bond dimension q^T, q being the number of values taken by each x^t. Now, if the distribution under consideration puts non-zero probability only over a small set $\mathscr{T}$ of trajectories, the number of components in the superposition, and hence the final bond dimension, is $|\mathscr{T}|$.

Any non-recurrent and Markovian model with q states such as SIR (Susceptible Infectious Recovered, $q = 3$), SEIR (Susceptible Exposed Infectious Recovered, $q = 4$), etc., allows only a sub-exponential fraction of the q^T potential trajectories. Take as an example the SIR model: each message $m_{i\to j}$ can be parametrized by the infection and recovery times for individuals i and j, for a total $\mathscr{O}(T^4)$ possible trajectories. The same reasoning goes for a generic non-recurrent Markovian model with q states, yielding bond dimension $\mathscr{O}(T^{q^2})$.

Chain models Consider $T+1$ variables each taking one in q values whose distribution is factorized over an open chain

$$p(x^0,x^1,\dots,x^T) \propto \prod_{i=0}^{T-1} \psi^t(x^t,x^{t+1}). \tag{B.11}$$

We show that there exists an equivalent formulation in MPS form, with matrices of size $q\times q$. Introduce additional variables $\{a^t\}_{t=1:T}$ with $a^t = x^t$ to get

$$p(x^1,x^2,\dots,x^T) \propto \sum_{a^1,a^2,\dots,a^T} \delta(x^0,a^1) \prod_{t=0}^{T-2} \left\{\psi^t(a^{t+1},x^{t+1})\delta(x^{t+1},a^{t+2})\right\} \psi^{T-1}(a^T,x^T) \tag{B.12}$$

$$\propto \sum_{a^1,a^2,\dots,a^T} \left[A^0(x^0)\right]_{a^1} \prod_{t=1}^{T-1} \left[A^t(x^t)\right]_{a^t,a^{t+1}} \left[A^T(x^T)\right]_{a^T} \tag{B.13}$$

$$\propto \prod_{t=0}^{T} A^t(x^t) \tag{B.14}$$

with

$$\begin{cases} \left[A^0(x^0)\right]_{a^1} & = \delta(x^0,a^1) \\ \left[A^t(x^t)\right]_{a^t,a^{t+1}} & = \psi^{t-1}(a^t,x^t)\delta(x^t,a^{t+1}) \quad \forall t \in 1,2,\ldots,T-1 \\ \left[A^N(x^T)\right]_{a^N} & = \psi^{T-1}(a^T,x^T) \end{cases} \tag{B.15}$$

where each a ranges over q values. We note the following implication: messages in the 1-step DMP method [75], which are parametrized as chain models, can be represented with matrices of size $q^2 \times q^2$.

One-particle trajectories in the SI model We show that the probability of any trajectory of an individual in the SI model can be represented by a MPS with matrices of size 2×2. It suffices to show that such probability factorizes over a chain. In the following we will sometimes use the convention $S = 0, I = 1$. The rule that once an individual i is infected at time t it can never recover is then encoded compactly as $\prod_{t=1}^{T} \mathbb{1}\left[x_i^{t+1} \geq x_i^t\right]$.

For a generic time t consider the conditional probability $p(x^{t+1}|x^0,x^1,\ldots,x^t)$. If $x^t = I$ then $p(x^{t+1}|x^0,x^1,\ldots,x^t) = \delta(x^{t+1},I)$. If $x^t = S$ then it must also be that $x^0 = x^1 = \ldots = x^{t-1} = S$. We conclude that the state at time $t+1$ depends on the previous states only through the state at time t: $p(x^{t+1}|x^0,x^1,\ldots,x^t) = p(x^{t+1}|x^t)$. Hence,

$$p(x^0,x^1,\ldots,x^N) = \prod_{t=0}^{T-1} p(x^{t+1}|x^0,x^1,\ldots,x^t) = \prod_{t=0}^{T-1} p(x^{t+1}|x^t) \tag{B.16}$$

The same thesis can be proven via a different argument: for "non-recurrent" models like SI, information about the trajectory can be encoded into a single parameter: the infection time. Infection at some time $t_i \in \{0,1,\ldots,T,\infty\}$ (we use the convention that no infection corresponds to $t = \infty$) corresponds to $x^0 = \ldots = x^{t-1} = S, x^t = \ldots, x^T = I$. It is sometimes convenient to switch between these two equivalent representations.

We propose a chain-factorized ansatz and show that it fully specifies the probability of a trajectory

$$p(x^0,x^1,\ldots,x^T) = \left[\prod_{t=0}^{T-1} \mathbb{1}\left[x^t \leq x^{t+1}\right] q^t(x^t)\right] q^T(x^T). \tag{B.17}$$

The probability of any of the allowed trajectories is

$$p(t_i = t) = p(x^0 = \ldots = x^{t-1} = S, x^t = \ldots, x^T = I) = \prod_{t=0}^{t-1} q^t(S) \prod_{t=t}^{T} q^t(I). \tag{B.18}$$

The ratio of probabilities of infection at times $t+1$ and t gives

$$\frac{p(t_i = t+1)}{p(t_i = t)} = \frac{q^t(S)}{q^t(I)}. \tag{B.19}$$

Parametrizing as $q^t(S) \propto 1, q^t(I) \propto e^{-h^t}$, we get

$$h^t = \log \frac{q^t(S)}{q^t(I)} = \log \frac{p(t_i = t+1)}{p(t_i = t)}. \tag{B.20}$$

In full detail, the resulting MPS is

$$\begin{cases} \left[A^0(x^0)\right]_{a^1} & = \delta(x^0, a^1) \\ \left[A^t(x^t)\right]_{a^t, a^{t+1}} & = \mathbb{1}[a^t \leq x^t] q^{t-1}(a^t) \delta(x^t, a^{t+1}) \quad \forall t \in 1, \ldots, T-1 \\ \left[A^T(x^T)\right]_{a^T} & = \mathbb{1}\left[a^T \leq x^T\right] q^{T-1}(a^T) q^T(x^T) \end{cases} \tag{B.21}$$

Pair trajectories in the SI model We show that any BP message in the SI model can be represented exactly by a MPS with matrices of size 6×6. Consider the BP equations for the SI model parametrized with infection times (see [87])

$$m_{i\to j}(t_i,t_j) \propto \sum_{t_{\partial i\setminus j}} \delta\left(t_i, \min_{k\in\partial i}\{t_k\}\right) \prod_{k\in\partial i\setminus j} m_{k\to i}(t_k,t_i) \tag{B.22}$$

$$\propto \mathbb{1}\left[t_i \le t_j\right] \prod_{k\in\partial i\setminus j} \sum_{t_k} \mathbb{1}\left[t_i \le t_k\right] m_{k\to i}(t_k,t_i) - \mathbb{1}\left[t_i < t_j\right] \prod_{k\in\partial i\setminus j} \sum_{t_k} \mathbb{1}\left[t_i < t_k\right] m_{k\to i}(t_k,t_i) \tag{B.23}$$

$$\propto \mathbb{1}\left[t_i \le t_j\right] a_{i\to j}(t_i) - \mathbb{1}\left[t_i < t_j\right] b_{i\to j}(t_i) \tag{B.24}$$

$$\propto \mathbb{1}\left[t_i \le t_j\right] c_{i\to j}(t_i) + \mathbb{1}\left[t_i = t_j\right] b_{i\to j}(t_i) \tag{B.25}$$

where we used $\delta(x,\min_{k\in S}\{x_k\}) = \prod_{k\in S}\mathbb{1}\left[x \le x_k\right] - \prod_{k\in S}\mathbb{1}\left[x < x_k\right]$ and defined $a_{i\to j}(t_i) = \prod_{k\in\partial i\setminus j}\sum_{t_k}\mathbb{1}\left[t_i \le t_k\right] m_{k\to i}(t_k,t_i)$, $b_{i\to j}(t_i) = \prod_{k\in\partial i\setminus j}\sum_{t_k}\mathbb{1}\left[t_i < t_k\right] m_{k\to i}(t_k,t_i)$, $c_{i\to j}(t_i) = a_{i\to j}(t_i) - b_{i\to j}(t_i)$.

Once normalized, both $c_{i\to j}$ and $b_{i\to j}$ are probability distributions for a single SI trajectory, hence they can be re-parametrized (with a slight abuse of notation) as MPSs $c_{i\to j}(x_i) = \prod_t \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \tilde{c}^t_{i\to j}(x_i^t)$, $b_{i\to j}(x_i) = \prod_t \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \tilde{b}^t_{i\to j}(x_i^t)$. Introducing the SI rule also for x_j,we get

$$\begin{aligned} m_{i\to j}(x_i,x_j) \propto & \prod_t \mathbb{1}\left[x_i^t = x_j^t\right] \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \tilde{b}_{i\to j}(x_i) + \\ & + \prod_t \mathbb{1}\left[x_i^t \le x_j^t\right] \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \mathbb{1}\left[x_j^{t+1} \ge x_j^t\right] \tilde{c}^t_{i\to j}(x_i^t). \end{aligned} \tag{B.26}$$

The first term is a chain-factorized distribution for, say, x_i times the constraint $x_j^t = x_i^t \forall t$, hence it can be represented as an MPS with 2×2 matrices. The second term is a chain of 4-state variables $\{(x_i^t, x_j^t)\}_{t=0:T}$, hence it can be represented as an MPS with 4×4 matrices. In full detail

$$\prod_t \mathbb{1}\left[x_i^t = x_j^t\right] \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \tilde{b}_{i\to j}(x_i) = \sum_{a_i^1,\dots,a_i^T} \prod_t \underbrace{\mathbb{1}\left[x_i^t = x_j^t\right] \delta(x_i^t, a_i^{t+1}) \mathbb{1}\left[a_i^t \le x_i^t\right] \tilde{b^{t-1}}_{i\to j}(a_i^t)}_{\left[B^t(x_i^t,x_j^t)\right]_{a_i^t,a_i^{t+1}}} \tag{B.27}$$

$$\prod_t \mathbb{1}\left[x_i^t \le x_j^t\right] \mathbb{1}\left[x_i^{t+1} \ge x_i^t\right] \mathbb{1}\left[x_j^{t+1} \ge x_j^t\right] \tilde{c}_{i\to j}^t(x_i^t) =$$

$$\sum_{\substack{a_i^1,\ldots,a_i^T \\ a_j^1,\ldots,a_j^T}} \prod_t \underbrace{\mathbb{1}\left[x_i^t \le x_j^t\right] \delta(x_i^t, a_i^{t+1}) \delta(x_j^t, a_j^{t+1}) \mathbb{1}\left[a_i^t \le x_i^t\right] \mathbb{1}\left[a_j^t \le x_j^t\right] \tilde{c}_{i\to j}^{t-1}(a_i^t)}_{[C^t(x^t)]_{(a_i^t, a_j^t),(a_i^{t+1}, a_j^{t+1})}} \tag{B.28}$$

Finally, since the mixture of two MPSs is itself an MPS (B.9), we get that $m_{i\to j}$ can be written as a MPS with matrices of size $2+4=6$.