Susceptibility Propagation for the Inverse Ising Model

CHARLES OLLION

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2010
Susceptibility Propagation for the Inverse Ising Model

CHARLES OLLION

Master’s Thesis in Biomedical Engineering (30 ECTS credits) at the School of Computer Science and Engineering
Royal Institute of Technology year 2010
Supervisor at CSC was Erik Aurell
Examiner was Anders Lansner

TRITA-CSC-E 2010:021
ISRN-KTH/CSC/E--10/021--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication
KTH CSC
SE-100 44 Stockholm, Sweden
URL: www.kth.se/csc
Abstract

In this paper we study the inverse Ising model, through one specific algorithm called Susceptibility Propagation. We show how this algorithm is derived and what are its theoretical results. We later apply it to simulated neural data and discuss the practical usefulness of the algorithm on it, compared to state-of-the-art methods.
## Contents

1 Introduction ................................................................. 1
   1.1 Introduction ......................................................... 1
   1.2 Motivations ............................................................. 2
   1.3 Outline ...................................................................... 2

2 The Ising Model ................................................................. 5
   2.1 Definition ................................................................. 5
      2.1.1 Maximum Entropy .................................................. 5
      2.1.2 Ising Model .......................................................... 6
   2.2 Present Methods .......................................................... 7
      2.2.1 Naive Mean Field reconstruction ............................... 7
      2.2.2 TAP reconstruction ............................................... 8
      2.2.3 Sessak-Monasson reconstruction ............................... 8
      2.2.4 Boltzmann Machine ............................................... 9

3 Message passing algorithm .................................................. 11
   3.1 Introduction ............................................................. 11
   3.2 The Sum-Product algorithm .......................................... 11
   3.3 Belief Propagation applied to Ising Model ......................... 14

4 Algorithm .......................................................................... 17
   4.1 Derivation ................................................................. 17
   4.2 Susceptibility Propagation Equations ................................. 18

5 Empirical results on abstract models .................................... 21
   5.1 Methods ..................................................................... 21
      5.1.1 General method ..................................................... 21
      5.1.2 Generation of $J_{ij}$ and $h_i$ ................................. 21
      5.1.3 Computation of $\chi_{ij}$ and $m_i$ .............................. 21
      5.1.4 Measure of the effectiveness of the algorithm ............. 22
   5.2 Reconstruction error ................................................... 24
      5.2.1 SK reconstruction .................................................. 24
      5.2.2 Performance depending on the distribution .................. 25
      5.2.3 Performance with Monte Carlo computed correlations .... 26
5.2.4 Performance with simulated noisy data .......................... 27
5.3 Study of the algorithm with more realistic data ..................... 28
  5.3.1 Performance for \( h_i \neq 0 \) .................................................. 28
  5.3.2 Performance at low temperature ....................................... 29
  5.3.3 Stopping procedure for the algorithm ............................... 30
  5.3.4 Performance for sparse graphs ....................................... 32

6 Performance on simulated neural data 35
  6.1 Settings of the experiment ................................................. 35
  6.2 Results ................................................................. 36

7 Discussion 39
  7.1 Further developments ..................................................... 39
  7.2 Conclusion ............................................................. 39

Bibliography 41

Appendices 43

A Demonstrations 43
  A.1 demonstrate \( h_{i \rightarrow j} = h_i + \sum_{k \in \partial \setminus j} u_{k \rightarrow i} \) ........................................... 43
  A.2 demonstrate \( \tanh u_{i \rightarrow j} = \tanh J_{ji} \tanh h_{i \rightarrow j} \) ......................................... 44
  A.3 demonstrate \( g_{i \rightarrow j,k} = \sum_{l \in \partial \setminus j} v_{l \rightarrow i,k} + \delta_{i,k} \) ........................................... 44
  A.4 demonstrate \( v_{i \rightarrow j,k} = g_{i \rightarrow j,k} \tanh J_{ij} \frac{1 - \tanh^2 h_{i \rightarrow j}}{1 - \tanh^2 u_{i \rightarrow j}} \) ............................... 45
  A.5 demonstrate \( m_i = \tanh (h_i + \sum_{j \in \partial} u_{j \rightarrow i}) \) .................................................. 45
  A.6 demonstrate \( \chi_{ij} = \frac{J_{ij} g_{j \rightarrow i,j} + g_{i \rightarrow j,i}(1 - m_i^2)}{C_{ij} - \tanh h_{i \rightarrow j} \tanh h_{j \rightarrow i}} \) ........................................... 46
  A.7 demonstrate \( \tanh J_{ij} = \frac{C_{ij} \tanh h_{i \rightarrow j} \tanh h_{j \rightarrow i}}{1 - C_{ij} \tanh h_{i \rightarrow j} \tanh h_{j \rightarrow i}} \) ........................................... 47
Chapter 1

Introduction

1.1 Introduction

How do we infer the most likely model given the knowledge of a set of data? From an Information theory point of view, this problem consists of finding a probability distribution that maximizes the entropy of the system, taking into account the data as constraints.

This Entropy theory, which was described by Shannon in [20], leads directly to a famous statistical physics problem, described as the Ising model. The Ising model is defined over a collection of variables that are driven by biases and have constraints, or couplings, between pairs. Computing the means and correlations of the variables (describing the data) from the biases and couplings (describing the model) is a NP-Hard problem. This problem is known as the Ising problem. We will focus on the inverse Ising problem: Given data, we want to infer the parameters of the Ising distribution with relative precision. This is also called an approximate reconstruction of the couplings.

The first approximate method uses the Mean Field Theory to approximate the couplings. This approach is applied to infer gene regulatory pairwise interaction from data consisting of gene expression level in [15]. Even if the results are quite promising, the Mean Field theory is often a poor approximation to strongly correlated systems.

Better approximation, such as Thouless-Anderson-Palmer or TAP approximation, or the Sessak-Monasson approximation were recently developed and shown to be more effective than the naive Mean Field Approximation. A survey of such approximations can be found in [11]. Even more recently, Mora and Mézard proposed a new algorithm in [2] called Susceptibility Propagation, that realizes the reconstruction.

This work focuses on testing this new algorithm. Due to its complicated nature, it has not been extensively tested and compared to the other approximations yet. The goal of this work is to determine whether it is useful for effective implementations or not. We will focus on two studies, one on the Sherrington-Kirkpatrick
model, which is a well-known environment for testing reconstruction methods. The second one will be one simulated neural data: From the study of spike trains of neurons, we want to identify functional connections between them, which is a relevant information for neural studies. We will focus on the data used to compare other methods, described in [10].

1.2 Motivations

While it is a challenge to find the best approximate to the inverse inference problem, there is also a real demand in biophysics for solving experimental problems. The Ising model in statistical physics is a model describing the states of interacting spins in a magnetic field, even if it can be extended to much different problems. In this section we will make a survey of applications of the inverse Ising problem.

Computation in the brain is performed by large populations of neurons which are highly connected and correlated. That is why the inference of the mechanisms is a difficult challenge, that only recent algorithms pretend to solve correctly. The analysis of neural networks has been connected to the inverse inference problem for a long time [5] [7] [8]. From local measures such as binned spike trains of neurons, we can define a spin variable for each neuron, and try to infer connections between them, which can be assimilated to the activity of synapses and give clues on their strength [10]. Knowing a model that generates neural-like spikes trains can also be useful for studying the distribution, looking for interesting behavior. Moreover, the model can then be used for generating more neural-like data.

Another application of the inverse Ising model in biophysics is the inference of interactions in gene regulation networks. Considering patterns of level of expression of genes, we want to reconstruct the graph of regulation of genes, and roughly the strength of these interactions. Such networks reconstructions have already been made using different Mean Field approximation in [15]. The resulting graph shows the pairwise interactions between genes, which can help identify protein interactions and regulations. The reverse-engineering of gene regulation patterns is still a difficult challenge and a lot of different methods are being developped to solve it [6].

The Inverse Ising model has also been used in [4] in structural biology, where one tries to infer structural parameters of a protein with the amino acid sequence. This list in not exhaustive, and the Ising model can probably be applied to many other fields.

1.3 Outline

We will first redefine the mathematical context of Susceptibility Propagation, which is based on the Inverse Ising model and a belief propagation algorithm. The Ising model will be covered in Chapter 2 and belief propagation in Chapter 3. In Chapter 4, we will redemonstrate and comment all the equations of the algorithm. Chapter 5 will cover the empirical performance of the algorithm on generated data,
1.3. OUTLINE

while Chapter 6 will describe the performance on simulated neural data, as used in [10]. Finally, Chapter 7 is dedicated to further developments and the conclusion.
Chapter 2

The Ising Model

2.1 Definition

2.1.1 Maximum Entropy

The entropy of a system as defined by Shannon in 1948 in [20] for a discrete probability \( p(\sigma) \) over a set of variables \( \sigma \) is written:

\[
S = \sum_{\sigma} -P(\sigma) \log(P(\sigma))
\]  
(2.1)

To create a model that describes the probability distribution using the knowledge of data such as means \( m_i \) and two-point correlations \( \chi_{ij} \) of the variables, we maximize the entropy under constraint. The term to be maximized is:

\[
\sum_{\sigma} -P(\sigma) \log(P(\sigma)) + \sum_i h_i (\sum_{\sigma} \sigma_i P(\sigma) - m_i) \\
+ \sum_{i,j} J_{ij} (\sum_{\sigma} \sigma_i \sigma_j P(\sigma) - (\chi_{ij} + m_i m_j)) \\
+ \lambda (\sum_{\sigma} P(\sigma) - 1)
\]  
(2.2)

Here \( h_i, J_{ij} \) and \( \lambda \) are the Lagrange multipliers for the maximization under constraint. By expending this relation with a small variation \( (P(\sigma) \leftarrow P(\sigma) + \partial P(\sigma)) \) and considering only the variational terms we have the following equation:

\[
\sum_{\sigma} \partial P(\sigma)(- \log P(\sigma) - 1 + h_i \sigma_i + J_{ij} \sigma_i \sigma_j + \lambda) + O(\partial^2 P(\sigma)) = 0
\]  
(2.3)

The maximized terms then corresponds to a probability distribution such as:

\[
P(\sigma) = \exp \left( -1 + \sum_i h_i \sigma_i + \sum_{i,j} J_{ij} \sigma_i \sigma_j + \lambda \right)
\]  
(2.4)
2.1.2 Ising Model

We will see that the model (2.4) is derived from the Maximum Entropy principle is the Ising model of statistical physics. The Ising problem is characterized by the following Energy function:

$$E(\sigma) = -\sum_i h_i \sigma_i - \sum_{i<j} J_{ij} \sigma_i \sigma_j$$  \hspace{1cm} (2.5)

In this equation \( h_i \) are the external biases and \( J_{ij} \) are the couplings between variables \( i \) and \( j \).

The equilibrium distribution at temperature \( T \) is:

$$p(\sigma) = \frac{1}{Z(T)} \exp \left( -\frac{E(\sigma)}{k_B T} \right)$$  \hspace{1cm} (2.6)

Where \( k_B \) is the Boltzmann constant and \( Z \) a normalization factor defined as followed:

$$Z(T) = \sum_\sigma \exp \left( -\frac{E(\sigma)}{k_B T} \right)$$  \hspace{1cm} (2.7)

It is then seen that the equation (2.4) is this equilibrium distribution at \( k_B T = 1 \).

The Ising Model is a mathematical model consisting of variables, that can take two different values. Considering the pair-wise Ising model, there is a magnetic interaction between pairs of variables. We will consider the mean field Ising model, which means that each variable is connected to all other variables (full graph of interaction).

The Ising model has been studied as a model for magnetism, and the variables can therefore be called "spins" \( \sigma_i = \pm 1 \). In straightforward physical model the couplings \( J_{ij} \) are non-zero only between spins that are close in space. In max-entropy model there is no such restriction. A model that we will use is the Sherrington-Kirkpatrick spin glass model, where all the \( J_{ij} \) are independent and Gaussian random variables.

In the studied model \( T \) only plays the role of setting a common scale of all the \( J_{ij} \) and \( h_i \). We will either consider the system at different explicit temperatures \( T \), or set \( T = 1 \) and modify all the \( J_{ij} \) in accordance to a parameter \( \tilde{J} \).

The Boltzmann distribution is linked to the means of the spins \( m_i \) and the two point correlations between the spins \( \chi_{ij} \) by the following formulae:

$$m_i = \frac{1}{Z} \sum_\sigma \sigma_i p(\sigma)$$  \hspace{1cm} (2.8)

$$\chi_{ij} = \frac{1}{Z} \sum_\sigma \sigma_i \sigma_j p(\sigma) - m_i m_j$$  \hspace{1cm} (2.9)
2.2. PRESENT METHODS

The Ising problem is defined as following: Knowing the parameters of the system \((h_i, J_{ij})\), we want to infer the behavior of the spins, which can for instance be described by \((m_i, \chi_{ij})\). In physics, other quantities are often of interest, which will not be considered here. This problem is difficult, because in order to get the exact correlation and means, you must sum over all possible spin states. For a system with \(N\) spins, this means a calculation over \(2^N\) states, which becomes large when \(N > 20\).

The pairwise Ising model is one of the simplest statistical model presenting a phase transition: If strength of the coupling \(J_{ij}\) are too high, (or the temperature \(T\) is too low), the spins become correlated at long distances, in the same way as magnetic materials at low enough temperatures. In the Sherrington-Kirkpatrick model, the low temperature (ordered) phase is a so-called spin-glass with complex properties. The phase transition in such a model occurs at \(T = 1\) and all \(h_i = 0\).

We are interested in the inverse problem: Reconstructing the couplings and external fields \((J_{ij}, h_i)\) from the experimental parameters \((m_i, \chi_{ij})\). It is easy to get the means and correlation from a dataset with \(M\) samples:

\[
m_i = \frac{1}{M} \sum_m \sigma_i^m \quad (2.10)
\]
\[
\chi_{ij} = \frac{1}{M-1} \sum_m \sigma_i^m \sigma_j^m - m_i m_j \quad (2.11)
\]

From this data, we will try to reconstruct the couplings and fields. In the following examples, we are only interested in couplings \(J_{ij}\) for the reconstruction. We will see that the difficulty of the reconstruction depends on the strength of the couplings: when approaching the transition phase, the reconstruction typically fails.

## 2.2 Present Methods

The reconstruction of the couplings on a Sherrington-Kirkpatrick Ising model has recently been studied by several authors. The different method that we will introduce here are studied theoretically in [11], and their effectiveness is also compared in [10].

### 2.2.1 Naive Mean Field reconstruction

The Mean Field theory consists of the following approximation:

\[
\sum_j J_{ij} \sigma_j \approx \sum_j J_{ij} m_j \quad (2.12)
\]

The first order approximation of mean field theory leads to the following equation:

\[
\tanh^{-1} m_i = h_i + \sum_{j \neq i} J_{ij} m_j \quad (2.13)
\]
By differentiating with respect to magnetization $m_i$, we get the following reconstruction rule:

$$ J_{ij} = -(\chi^{-1})_{ij} \quad (2.14) $$

This is the simplest rule, and it only requires to invert the correlation matrix (if invertible!). However this rule implies no magnetization which means it will be a poor reconstruction one for high magnetization systems.

### 2.2.2 TAP reconstruction

The Thouless-Anderson-Palmer or TAP approximation [19] is an extension of this model. It relies on the second order approximation of mean field theory:

$$ \tanh^{-1} m_i = h_i + \sum_{j \neq i} J_{ij} m_j - \sum_{j \neq i} J_{ij}^2 (1 - m_j^2) m_i \quad (2.15) $$

By differentiating with respect to magnetization $m_j$, we get the following new reconstruction rule (for $i \neq j$):

$$ J_{ij} + 2J_{ij}^2 m_j m_i + (\chi^{-1})_{ij} = 0 \quad (2.16) $$

This rule can also be derived from the Belief Propagation update rules [11]. This equation can be solved to infer the couplings values. The $h_i$ can later be inferred by solving equation (2.10) after substituting the resulting $J_{ij}$. When $(\chi^{-1})_{ij}$ is too negative, the second order equation only has complex solutions. In that case, Naive Mean Field can be used.

When the magnetizations are all zero, TAP is equivalent to Naive Mean Field. TAP is one of the most effective methods used for solving the Inverse Ising model and will be used as reference in this report.

### 2.2.3 Sessak-Monasson reconstruction

More recently, Sessak and Monasson [14] have developed a different expansion of the fields and couplings of the Ising distribution. The Sessak Monasson approximation can be written as:

$$ J_{ij} = -C_{ij}^{-1} + J_{ij}^{IP} - \frac{C_{ij}}{(1 - m_i^2)(1 - m_j^2) - C_{ij}^2} \quad (2.17) $$

Here $J_{ij}^{IP}$ is the reconstructed couplings with the Independant Pair approximation: one considers that the pairwise couplings are independant and can then reconstruct them easily. This approximation itself is very bad, but summed with the other terms of the Sessak Monasson approximation, it is one of the best reconstruction methods, that can sometimes outperform TAP.
2.2. PRESENT METHODS

2.2.4 Boltzmann Machine

The last reconstruction method that we will present is a bit different. It does not rely on an approximation, but on a learning rule called Boltzmann machine in Neuroscience:

\[ \partial h_i = \eta(\langle \sigma_i \rangle_{Data} - \langle \sigma_i \rangle_{Ising}) \tag{2.18} \]
\[ \partial J_{ij} = \eta(\langle \sigma_i \sigma_j \rangle_{Data} - \langle \sigma_i \sigma_j \rangle_{Ising}) \tag{2.19} \]

To get the \( \langle \sigma_i \rangle_{Ising} \), one has to perform Monte Carlo runs using for instance Metropolis-Hastings algorithm. If N is large, these runs can be extremely long, and one run has to be performed for each step of the learning rule. This is why this iterative method is much slower than the other methods. However, this method is guarantted to converge towards a solution that is in principle exact, and thus can be used as a reference for comparing other methods.
Chapter 3

Message passing algorithm

3.1 Introduction

This chapter will cover the Belief Propagation algorithm, from which Susceptibility Propagation can later be derived.

Many scientific and technological problems today rely on computing a complicated global function of many variables. The most common ones are optimization problems of a cost function defined on a discrete set of variables, such as the previously defined Ising model. As computing a summation over all variables would become quickly computationally prohibitive even for rather small problems, we are interested in a solution that takes advantage of 'local' intermediates, even if that solution is only an approximation.

3.2 The Sum-Product algorithm

The Sum-Product algorithm, as described in by Kschischang in [1], is designed to compute the local marginals of a global probability function.

\[ p_i(x_i) = \sum_{x' \in \{x \setminus x_i\}} p(x') \] (3.1)

The marginal functions \( p_i(x_i) \) need be factorized in several smaller so-called local functions, which can be computed easily.

The Sum-Product algorithm takes advantage of a factor graph - a graph that represents the factorization of the global function into smaller ones - to compute efficiently the marginals.

\[ p(x) = \prod_j p_j(x_j) \] (3.2)

where \( x_j \) are sets of variables usually much smaller than the set of all variables \( x \).

For example let us assume that the probability function can be factorized in the following way:

\[ p(x_1, x_2, x_3, x_4, x_5) = f_1(x_1, x_3)f_2(x_1, x_2, x_5)f_3(x_2, x_4)f_4(x_3), \] (3.3)
we will define the factor graph that describes the factorization as:

Each node is either a variable node or a factor node. Associated to a factor node is the function of its related variables. This graph is bipartite, which means that there are no links between variable nodes or function nodes. There is a link between a variable node and function node if and only if that variable is an argument of that function.

When the factor graph is a tree, the Sum-Product algorithm computes all the marginals at the same time. This can be seen as passing messages from the leaves of a tree to the root, for all different rootings, with all the messages partially stored before rooting. We differentiate two types of messages: those from variables \( x \) to factor \( f \), and the other way.

\[
p_{x_i \rightarrow f_j}(x_i) = \frac{1}{Z_{x_i \rightarrow f_j}} \prod_{f_k \in \partial x_i \setminus f_j} q_{f_k \rightarrow x_i}(x_i) \quad (3.4)
\]

\[
q_{f_j \rightarrow x_i}(x_i) = \frac{1}{Z_{f_j \rightarrow x_i}} \sum_{x' \in \{x_j \setminus x_i\}} \left( f_j(x', x_i) \prod_{x_k \in \partial f_j \setminus x_i} p_{x_k \rightarrow f_j}(x'_k) \right) \quad (3.5)
\]

where \( \partial x_i \) are the factor nodes in the neighbourhood of \( x_i \), and \( \partial f_j \) are the variable nodes in the neighbourhood of \( f_j \). \( Z_{x_i \rightarrow f_j} \) and \( Z_{f_j \rightarrow x_i} \) are normalization
3.2. THE SUM-PRODUCT ALGORITHM

Constants that can sometimes be omitted in implementations. Details of computation can be found in [1]. After exchanging messages, the marginals are expressed that way:

\[ p_i(x_i) = \frac{1}{Z_{i}} \prod_{f_k \in \partial x_i} q_{f_k \rightarrow x_i}(x_i) \] (3.6)

When the graph is not a tree, but is locally a tree, the message passing algorithm can still work by making the messages update many times and converge to the solution after sufficiently many iterations. The graph is considered locally a tree if it is not too connected (particularly the presence of small loops can decrease the efficiency of the algorithm), or if the factors in the loops do not imply too strong constraints. The algorithm used on any graph is well known as Belief Propagation.

This algorithm which is only based on a mathematical factorisation principle, can be used and derived in many different applications. It is widely used in coding and error correction: the low density parity checks (LDPC) can be seen as factor graphs and belief propagation proves to be one of the most efficient method to decode messages. There are other applications like SAT problems, the inference of Bayesian networks and many more, as discussed in [1] and [3].
3.3 Belief Propagation applied to Ising Model

In this section we will show how Belief Propagation is written for the pairwise Ising model.

The function we are trying to factorize here is the probability distribution over the spin states of an Ising model:

\[ p(\sigma_i) = \frac{1}{Z} \sum_{\sigma' \in \{\sigma \setminus \sigma_i\}} e^{-E(\sigma')} \quad (3.7) \]

The probability function is an exponential of a sum, thus it can be factorized into a products of simple exponentials, each depending on only one or two spins:

\[ p(\sigma) \propto \prod_i e^{-h_i \sigma_i} \prod_{i < j} e^{-J_{ij} \sigma_i \sigma_j} \quad (3.8) \]

The corresponding factor graph for an Ising model for \( N = 3 \) can look like:

As each variable node has one factor corresponding to the external field and that the factors corresponding to couplings only link two variables (pairwise couplings), we can simplify the model by only considering the variable nodes, and making the factor nodes implicit. The messages are then passed from one (variable) node to another as:

\[ p_{i \rightarrow j}(\sigma_i) \leftrightarrow p_{\sigma_i \rightarrow f_j}(\sigma_i) \]
\[ q_{j \rightarrow i}(\sigma_i) \leftrightarrow q_{f_j \rightarrow \sigma_i}(\sigma_i) \]
3.3. BELIEF PROPAGATION APPLIED TO ISING MODEL

Where $f_j$ is a factor corresponding to $e^{-J_{ij}\sigma_i\sigma_j}$. Due to the particular structure of the graph, we can simplify the message updating rules. First we can include the factors corresponding to the external fields in the $p$-messages:

$$p_{i\rightarrow j}(\sigma_i) = e^{-h_i\sigma_i} \prod_{f_k \in \partial_i \setminus f_j} q_{k\rightarrow i}(\sigma_i)$$ (3.9)

where $\partial i$ are the nodes in the neighbourhood of $i$.

Then we can simplify the $q$ messages by seeing that the partial sum is a sum over only one variable because the factor node is involved with only two variables.

$$q_{j\rightarrow i}(\sigma_i) = \sum_{\sigma_j = \pm 1} e^{J_{ij}\sigma_i\sigma_j} p_{j\rightarrow i}(\sigma_j)$$ (3.10)

And to finally retrieve the marginals:

$$p_i(\sigma_i) = e^{-h_i\sigma_i} \prod_{f_k \in \partial i} q_{k\rightarrow i}(\sigma_i)$$ (3.11)

These equations are the canonical update equations of Belief Propagation applied to the pairwise Ising model.

A convenient way to work on these equations is to use the log-likelihood notation of the messages and marginals.

$$h_{i\rightarrow j} = \frac{1}{2} \log \frac{p_{i\rightarrow j}(+1)}{p_{i\rightarrow j}(-1)}$$ (3.12)

$$u_{i\rightarrow j} = \frac{1}{2} \log \frac{q_{i\rightarrow j}(+1)}{q_{i\rightarrow j}(-1)}$$ (3.13)

The messages can then be related to each other in an easier way:

$$h_{i\rightarrow j} = h_i + \sum_{k \in \partial i \setminus j} u_{k\rightarrow i}$$ (3.14)

$$\tanh u_{i\rightarrow j} = \tanh J_{ji} \tanh h_{i\rightarrow j}$$ (3.15)

Furthermore, we have the effective fields $H_i = \frac{1}{2} \log \frac{p_i(+1)}{p_i(-1)}$ which are related to $m_i$ and $\chi_{ij}$ in the following way:

$$m_i = \tanh H_i$$ (3.16)

$$\chi_{ij} = \frac{\partial \tanh H_i}{\partial h_j}$$ (3.17)

Then the means and correlations are expressed in the following way:

$$m_i = \tanh (h_i + \sum_{j \in \partial i} u_{j\rightarrow i})$$ (3.18)

Details of the log-likelihood messages can be found in the appendix A.
Chapter 4

Algorithm

4.1 Derivation

In this section we will derive and describe the susceptibility propagation algorithm as introduced by T. Mora and M. Mézard [2]. This algorithm is derived from belief propagation algorithm applied to Ising model that we introduced in the previous section.

The aim of this inverse algorithm is to find coupling variables $J_{ij}$ and exterior fields $h_i$ from two-points correlation functions $\chi_{ij}$ and magnetizations $m_i$. As a starting point, we will use the log-likelihood notation and their derivatives as defined by:

\[
\begin{align*}
  h_{i \rightarrow j} &= \frac{1}{2} \log \frac{p_{i \rightarrow j}(+1)}{p_{i \rightarrow j}(-1)}, \\
  u_{i \rightarrow j} &= \frac{1}{2} \log \frac{q_{i \rightarrow j}(+1)}{q_{i \rightarrow j}(-1)}, \\
  g_{i \rightarrow j,k} &= \frac{\partial h_{i \rightarrow j}}{\partial h_k}, \\
  v_{i \rightarrow j,k} &= \frac{\partial u_{i \rightarrow j}}{\partial h_k}
\end{align*}
\]  

(4.1)

The next equations are the steps that lead to the Susceptibility Propagation algorithm. These steps are taken from [2] and each of them is demonstrated in appendix A.

First, here are the messages updates as well as their derivative using the log-likelihood notation:

\[
\begin{align*}
  h_{i \rightarrow j} &= h_i + \sum_{k \in \partial i \setminus j} u_{k \rightarrow i}, \\
  \tanh u_{i \rightarrow j} &= \tanh J_{ji} \tanh h_{i \rightarrow j} \\
  g_{i \rightarrow j,k} &= \sum_{l \in \partial i \setminus j} v_{l \rightarrow i,k} + \delta_{i,k}, \\
  v_{i \rightarrow j,k} &= g_{i \rightarrow j,k} \tanh J_{ij} \frac{1 - \tanh^2 h_{i \rightarrow j}}{1 - \tanh^2 u_{i \rightarrow j}}
\end{align*}
\]  

(4.3)

(4.4)
Then the means and correlations are expressed in that way:

\[ m_i = \tanh (h_i + \sum_{j \in \partial i} u_{j \rightarrow i}) \quad (4.5) \]

\[ \chi_{ij} = \chi_{ij} g_{j \rightarrow i, j} + g_{i \rightarrow j, j} (1 - m_i^2) \quad (4.6) \]

where:

\[ \overline{\chi_{ij}} = \frac{\tanh J_{ji} + \tanh h_{i \rightarrow j} \tanh h_{j \rightarrow i}}{1 + \tanh J_{ji} \tanh h_{i \rightarrow j} \tanh h_{j \rightarrow i}} - m_i m_j \quad (4.7) \]

This can be used to solve the normal Ising problem, and can also be reversed to express \( J_{ij} \) as a function of \( \chi_{ij} \) and the messages.
4.2 Susceptibility Propagation Equations

The inversion algorithm can be understood as follows:

In a forward step, the values of the messages $h_{i\rightarrow j}$ and $g_{i\rightarrow j,k}$ are updated. It is a fact that the correlation functions $\chi_{ij}$ can be expressed in only these Belief-Propagation variables, as well as the magnetization $m_i$ and the couplings $J_{ij}$. These are the twin equations above (4.3) and (4.4), which can be inverted to express $J_{ij}$ in terms of $\chi_{DATA}^{ij}, m_i^{DATA}, h_{BPvariable}^{i\rightarrow j}$ and $g_{BPvariable}^{i\rightarrow j,k}$.

In a second forward step, the values of $u_{i\rightarrow j}$ and $v_{i\rightarrow j,k}$ are updated. Here is the final expression of the algorithm as implemented in this work, and the order in which the variables are updated:

$$h_{i\rightarrow j} \leftarrow \text{arctanh} \ m_i - u_{j\rightarrow i}$$  \hspace{1cm} (4.8)

$$g_{i\rightarrow j,k} \leftarrow \sum_{l \in \partial i \setminus j} v_{l\rightarrow i,k} + \delta_{i,k}$$  \hspace{1cm} (4.9)

$$C_{ij} \leftarrow \frac{\chi_{ij} - g_{i\rightarrow j,j}(1 - m_i^2) + m_i m_j}{g_{j\rightarrow i,j}}$$  \hspace{1cm} (4.10)

$$\tanh J_{ij} \leftarrow \frac{C_{ij} - \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i}}{1 - C_{ij} \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i}}$$  \hspace{1cm} (4.11)

$$\tanh u_{i\rightarrow j} \leftarrow \tanh J_{ij} \tanh h_{i\rightarrow j}$$  \hspace{1cm} (4.12)

$$v_{i\rightarrow j,k} \leftarrow g_{i\rightarrow j,k} \tanh J_{ij} \frac{1 - \tanh^2 h_{i\rightarrow j}}{1 - \tanh^2 u_{i\rightarrow j}}$$  \hspace{1cm} (4.13)

After convergence,

$$h_i \leftarrow \text{arctanh} \ m_i - \sum_{j \in \partial i} u_{j\rightarrow i}$$  \hspace{1cm} (4.14)

This algorithm is basically the same as described in [13], for $T=1$. Having explicitly $T$ in the equations of the algorithm can be good for theoretical experiments, but is of no use for real data when the temperature is not known a priori. There are a few tricks that can help the algorithm converge, that we will describe in the following section.

To initialize the algorithm, one has to pick random messages $u_{i\rightarrow j}$ and initialize all other parameters to 0.
We notice that another algorithm can also be considered by starting at equation (4.12). The variables are then updated in that way: (4.12), (4.13), (4.8), (4.9), and the expression for the correlation function that go into (4.10) then account for computing $C_{ij}^{BP,J} = \langle \sigma_i \sigma_j \rangle_{BP,J} - \langle \sigma_i \rangle_{BP,J} \langle \sigma_j \rangle_{BP,J}$, where the expansion values are with respect to the Belief Propagation approximation of the probability distribution, and the current $J_{ij}$. A Boltzmann machine update of the $J_{ij}$ could then be:

$$J_{ij}^{\text{new}} = J_{ij}^{\text{old}} + \eta (-C_{ij}^{BP} + C_{ij}^{DATA})$$

However, Susceptibility Propagation does not use this learning rule, and directly updates a single coupling $J_{ij}$ in a different iterative process.

The computational complexity of the algorithm is $\Omega(N^3\text{Iteration})$, and it requires the memorization of $2N^3$ real variables. This is to be compared to other methods such as TAP which only requires $\Omega(N^2)$ to perform a matrix inversion and is not an iterative process. On a modern workstation, Susceptibility Propagation could be run up to $N=500$, while TAP could be run up to $N=3000$. 


Chapter 5

Empirical results on abstract models

5.1 Methods

In this section we will study the effectiveness of the Susceptibility Propagation algorithm compared to other state of art methods.

5.1.1 General method

To test the quality of the reconstruction, we generate an instance of $J_{ij}$ and $h_i$, compute correlations $\chi_{ij}$ and means $m_i$, and use these last two as the input for the Susceptibility Algorithm. The output of the algorithm will be $J'_{ij}$ and $h'_i$, which we can compare to the first $J_{ij}$ and $h_i$.

There are many ways to do each of these steps.

5.1.2 Generation of $J_{ij}$ and $h_i$.

The Susceptibility Propagation algorithm is designed for symmetric $J_{ij}$. The most common model we will use is the Sherrington-Kirkpatrick model, where $J_{ij}$ are chosen according to a normal law of mean 0 and variance $1/N$. $h_i$ can either be zero, chosen as a constant level of magnetization, or random.

We will also consider a binary model where $J_{ij} = \pm \frac{1}{\sqrt{N}}$, according to a uniform law.

And finally, we will consider a sparse model where most $J_{ij}$ are close to 0 and a few ones take random normal values as defined above. This will arguably be the model which is the closest to real data.

5.1.3 Computation of $\chi_{ij}$ and $m_i$

The Ising problem consists on finding $\chi_{ij}$ and $m_i$ from $J_{ij}$ and $h_i$ and is generally a computationally hard problem. To compute exact correlation and means from $J_{ij}$ and $h_i$ by going exhaustively to each spin state, the computational complexity is about $\Omega(2^N N^2)$. Because of that, we can only compute exact correlation and
means for $N \leq 20$.

The second method we will be using is the Metropolis-Hastings algorithm. This is an iterative process based on Monte Carlo that converges to $\chi_{ij}$ and $m_i$, and proceeds as follows:

Generate a random configuration of spins. For each iteration, do

- pick a random spin and invert it
- compute the energy difference $\Delta E$ between the new and old configuration.
- if $\Delta E < 0$ keep the spin inversion
- if $\Delta E > 0$ keep the spin inversion with probability $p = e^{-\Delta E}$

Then compute correlation and means from all these configurations. A first 'burn-in' period before equilibrium is omitted.

This algorithm has a complexity of $\Omega(N \text{Iteration})$, but usually requires a huge number of iteration before giving good results. The precision of the results is majored by $\sqrt{\frac{N}{\text{Iteration}}}$. For $N=128$, you would need at least $10^7$ iterations to achieve a precision of $10^{-2}$.

This algorithm has also another drawback, which is that it converges even slower when $T$ increases. This is explained by the fact that when $T$ is big, $\Delta E$ will be smaller, and the probability of spin inversion will become very close to 1. In order to get good statistics, the number of iterations must be increased according to this also.

The Metropolis–Hastings algorithm will often require $10^9$ iterations to get the required precision, which is clearly too slow to be effective. There are a few ways to improve this algorithm, by doing clustering [16], but while it enables to reduce the effect of $N$ on the precision, we still have trouble at large $T$.

5.1.4 Measure of the effectiveness of the algorithm

The usual way to evaluate how good is the reconstruction is to compare the couplings. The inference of the fields is often neglected. The most systematic measure of the distance between the original couplings $J_{ij}$ and reconstructed couplings $J'_{ij}$ is the arguably Kullback–Leibler distance, but requires too much computation for large systems (exhaustive computation on all spin states). Instead of this distance, we will use the average square difference between couplings, as used in [2]:

$$
\Delta = \sqrt{\frac{\sum_{i<j}(J'_{ij} - J_{ij})^2}{N(N-1)/2}} = \sqrt{\frac{\sum_{i<j}(J'_{ij} - J_{ij})^2}{(N-1)/2}}
$$  \hspace{1cm} (5.1)
5.1. METHODS

\( \sigma \) is here the variance of the distribution, and will be set to \( \tilde{J}/\sqrt{N} \) in the Sherrington-Kirkpatrick model. One can either consider the distribution of the couplings to be modified by a temperature \( T \), or \( \tilde{J} \). The only important factor that will affect the behavior of the different methods for reconstruction is \( \tilde{J}/T \). For easier comparison with other studies, we will modify the parameter \( T \) while keeping \( \tilde{J} \) to 1.

The algorithm is compared to other methods that rely on the inversion of the correlation matrix. These methods are described in Chapter 2 and tested in [11].

- Naive mean-field inversion: \((C^{-1})_{ij} = -J_{ij}\)
- TAP approximation: \((C^{-1})_{ij} = -J_{ij} - 2m_im_jJ_{ij}^2\)
- Sessak-Monasson approximation (SM): \(J_{ij} = -(C_{ij})^{-1} + J_{ij}^P - \frac{C_{ij}}{(1-m_i^2)(1-m_j^2)-C_{ij}}\)
CHAPTER 5. EMPIRICAL RESULTS ON ABSTRACT MODELS

5.2 Reconstruction error

5.2.1 SK reconstruction

In this section, the external fields $h_i$ are zero, and the couplings $J_{ij}$ follow a SK model.

Here is a first display of the error $\Delta$ at different temperatures, compared to the Naive mean-field inversion for the same data (which is equivalent to TAP when all $h_i$ are zero). The use of small $N$ makes the exact computation of correlations possible. In this graph we only consider $T > 5$ because the algorithm can enter into non-convergent behaviors under $T = 5$. We will discuss later on the convergence at low temperatures.

For this theoretical data, Susceptibility Propagation greatly outmatches the basic matrix inversion.
5.2. RECONSTRUCTION ERROR

5.2.2 Performance depending on the distribution

The reconstruction process could be used to reconstruct graphs with couplings that
do not follow a normal law. Here we study the reconstruction of a binary graph:
All the couplings are $J_{ij} = \pm 1/\sqrt{N}$.

This graph shows that the nature of the distribution hardly affects the precision
of the reconstruction. This is a good news for the reconstruction of graphs in
biological systems that often have patterns that differentiate them from gaussian
distributed graphs.

We will only study normal distributed couplings for the next performance tests.
5.2.3 Performance with Monte Carlo computed correlations

The means and correlations are now generated with the Monte Carlo method as described in section 5.1.3. The set of data generated is more relevant than exact computation, because it is closer to experimental data, with statistical errors. A huge set of Monte Carlo steps is required to achieve a good precision of means and correlations. Here is the error reconstruction depending on the number of Monte Carlo Steps. All the results below are obtained for $N = 128$. The error $\Delta$ decrease

![Figure 5.1: Reconstruction error $\Delta$ depending on the number of Monte Carlo Steps](image)

when the number of Monte Carlo Steps increase, which means the precision of the convergence and means are the limiting factor for the reconstruction. Due to the huge time necessary to compute $10^{10}$ Monte Carlo Steps, we couldn’t go further to see when the reconstruction would be limited by the Susceptibility algorithm rather than by the precision of the correlations. However for $T = 5$ we can see that the limiting factor becomes the Susceptibility Propagation precision when the number of steps is big enough.

The slope of the precision increase also confirms that the precision of the correlations is related to $\sqrt{\text{Steps}}$. 

26
5.2. RECONSTRUCTION ERROR

5.2.4 Performance with simulated noisy data

In order to study deeper the loss of precision due to noise, the algorithm is now given exact correlations with controlled noise level. The most relevant way to add noise to correlations is to add a small random gaussian value to the correlations and see how the reconstruction is affected.

![Figure 5.2: Reconstruction error $\Delta$ for different Noise levels](image)

This graph shows that even a quite low level of noise quickly reduces the precision of the reconstruction. The noise level can easily be the bottleneck to the reconstruction.

These empirical tests have already been presented in the very recently published paper [13], and had the same conclusion.
5.3 Study of the algorithm with more realistic data

5.3.1 Performance for $h_i \neq 0$

The level of the fields also modifies the reconstruction quality of the couplings $J_{ij}$. Here we plot the error distance $\Delta$ as a function of the level $h$ of all $h_i$.

![Influence of $h$ over the precision of reconstruction](image)

This graph shows that having a high $h$ decreases the precision of the reconstruction. However, even with large values, for instance $h = \pm 1$, the precision is only decreased by a factor of 10, at any temperature. This graph has been made with $N = 20$, but we obtain the same with $N = 10$, and in general $N$ does not seem to modify the performances.

At low temperatures, this can lead to strong damage on the reconstruction, because it might lead the algorithm to a non convergent phase: If the algorithm is close to critical temperature for $h = 0$, then adding a level $h$ will make it diverge.
5.3. STUDY OF THE ALGORITHM WITH MORE REALISTIC DATA

5.3.2 Performance at low temperature

At low temperatures, the right term in the update of $J$ (equation 4.15) can become too big, and eventually have an absolute value bigger than one. Then the algorithm cannot evaluate the hyperbolic arctangent, and it will fail. The first way to lessen this problem is to introduce a damping factor $\epsilon$ by modifying the update rule in the following manner (as introduced in [13]):

$$
tanh J_{ij} \leftarrow \epsilon \frac{C_{ij} - \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i}}{1 - C_{ij} \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i}} + (1 - \epsilon)J_{ij} \quad (5.2)
$$

This term will make the update of the coupling slower, so that the algorithm will require more steps to converge, but can also avoid some fluctuation that would lead the algorithm to fail. With a very low $\epsilon$, the algorithm can be in 3 distinct phases depending on the temperature:

- At fairly high temperature, $T > 5$ the algorithm converges.
- At low temperature, $T > 3$ the algorithm is in a divergent state, but the $\epsilon$ term makes it diverge very slowly, and eventually the terms that starts the divergence become so small that they are under the floating point precision, and the algorithm stops diverging.
- At very low temperature $T < 3$, the algorithm is in a divergent state, even if $\epsilon$ is very low.

Here is a plot of the convergence at low temperature, using $\epsilon = 0.01$, waiting 10000 Iteration of Susceptibility Propagation for an eventual divergence. This graphs shows a quite clear transition that becomes sharper when $N$ increases. This is probably due to the fact that small dimension systems are more likely to have instances with a very low $J_{ij}$ distribution, which would result in a convergent behavior.

For each instance there is presumably a critical temperature $T_c$ above which the algorithm almost surely converges and below which it almost surely diverges.
5.3.3 Stopping procedure for the algorithm

At low temperatures, the Susceptibility Algorithm will almost surely diverge. However, before it diverges, it often comes really close to a good solution, which would give a reasonable approximation.

To prove that, we have selected a random instance of $J_{ij}^{true}$, for N=16, computed exhaustively the exact correlations for a given temperature $T$, run the Susceptibility algorithm and looked at the evolution of the $J_{ij}^{found}$. As we defined in previous section, there is a critical temperature $T_c$ that separate convergence and divergence for each instance of couplings.

In the following graph we show the evolution of the reconstruction of one coupling for one specific instance with $4 < T_c < 4.25$. The following graphs shows the reconstruction of $J_{01}$ for $T = 4$ and $T = 4.25$. 

Figure 5.3: Convergence percentage. N = 10, 15 with exact correlations; N = 30, 60 with Monte Carlo correlations
When the number of iterations is still small, it is clear that $J_{01}$ has the same behavior in both cases. But in the case of $T < T_c$ the algorithm will diverge after some iterations. This is why stopping the algorithm at Iteration=150 would give an approximation of $J_{ij}$ almost as good as if the algorithm had converged.

Here is the current stopping procedure:

$$if \forall i, j \ |J^t_{ij} - J^{t-1}_{ij}| - |J^{t-1}_{ij} - J^{t-2}_{ij}| > 0 then \ STOP$$

(5.3)

If the absolute difference between two consecutive $J_{ij}$ is increasing, then it means the algorithm is not slowly getting closer and closer to a solution, but rather diverging.

Without this procedure, the convergence is generally guaranteed for $T > 5$ (See previous section). This procedure guarantees good results for $T > 2$.

This procedure also stop the algorithm earlier, shortening the number of iterations and, by that way, the computation time. Note that a converging algorithm will stop a few iterations after convergence thanks to the stopping procedure.
5.3.4 Performance for sparse graphs

A SK model with Gaussian random couplings is not clearly always a good approximation to the coupling distribution in real systems. We are considering complete graphs of interaction, as opposed to the sparse graphs often encountered in e.g. biological systems. For instance in gene regulation networks, a very few connections are very strong, while a lot of others are weak or inexistant.

A better approximation is to work on sparse graphs, where only a fraction $s$ of the connections are strong, and the other are weakened. The coupling distribution is then:

- for $s\%$ of couplings, a normal distribution.

- for the other couplings, a normal distribution divided by a factor of 10.

The algorithm was then tested with very high couplings ($\tilde{J} = 1$) and strong magnetisation ($h_i = -0.9$). Here are the performance results for the exact computation for small $N$:

![Figure 5.4: Reconstruction error for different sparse factor](image)

In this graph one can see that at $s = 0\%$, the performance is equivalent to $\tilde{J} = 0.1$, and $s = 100\%$ equivalent to $\tilde{J} = 1$, where the algorithm is forced to stop very fast and give very bad results. Still, for $s \leq 30\%$, the results are good enough to be used.

With higher dimension $N$ and Monte Carlo simulation, the algorithm with a 10\% sparse factor is still very useful:
5.3. STUDY OF THE ALGORITHM WITH MORE REALISTIC DATA

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>MC 10^8</th>
<th>MC 10^9</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>64</td>
<td>0.016</td>
<td>0.0066</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>0.027</td>
<td>0.0097</td>
</tr>
</tbody>
</table>

Table 5.1: Reconstruction error for different N and Monte Carlo steps

The difference between $N = 64$ and $N = 128$ is rather small, and the Monte Carlo step is the limiting factor for the precision, at least for $10^8$ steps. Here are the couplings found plotted against the generated couplings, for $N=128$:

![Scatterplot of the couplings](image)

These are very good reconstruction values for such big $N$ and high $\tilde{J}$. This shows that the Susceptibility Propagation algorithm works best on sparse graphs. This was to be expected, as the algorithm is derived from Belief Propagation, which is not assumed to be converging when the graph contains small loops. The TAP approximation would not give any good results in these cases. The Sessak-Monasson algorithm is yet to be tested on such graphs.

The Susceptibility Algorithm will have trouble converging or giving a good approximation when there is such a loop, in other words when there are at least 3 strong couplings between 3 spins. This is much less likely to happen in sparse graph, and is the reason for these good results on sparse graphs.

In experimental data, one should expect sparse graphs, which is very good news for the quality of the reconstruction, but one can also expect small, strongly correlated loops. The reconstruction of a linear chain with strong correlations is typically easily performed by Susceptibility Propagation provided the loop is not too small [3].
Chapter 6

Performance on simulated neural data

6.1 Settings of the experiment

By using data generated for neural studies in [10], we can compare the results of susceptibility propagation on real experimental data to other methods described in [10] and [11].

The data is generated from a neural model described in figure 6.1: There are two population of neurons, one excitatory and one inhibitory, with random connections. The connectivity is $c = 0.1$. The simulation uses conductance based synapses; more details can be found in [10].

From this neuron model, the spiking activity is recorded. In order to have a static model that can be assimilated to an Ising model, one has to bin the recordings into time steps, and consider that the variable at a time bin $t$ is $+1$ if the neuron fired during the bin, or $-1$ if it didn’t fire.

The data was binned in 400000 small time steps, from which we can easily extract the means and correlation. Each of these steps corresponds to $N$ Monte Carlo steps in terms of precision for the means and correlations. This number is rather small compared to the numbers of steps used in the theoretical analysis, which means the reconstruction will already be limited by this factor.

From the 1000 neurons, one only selects a few ones to attempt a reconstruction. We remove neurons that have a too low firing rate ($m_i \leq -0.98$), and then select the first 20, 40 or 100 neurons. We attempt to reconstruct the couplings (and dismiss the external fields, which are not relevant) with different methods such as TAP, and Susceptibility Propagation (SusP).

In order to compare the results of the different approximation methods, we plot each value of coupling against the couplings found by Boltzmann learning. This iterative method requires to do a Monte Carlo descent at each step, and thus is very time consuming. However, it is guaranteed to slowly converge to the exact value, and by waiting long enough (more that 3 days for larger $N$) we can achieve the required precision.
6.2 Results

We will consider $J_{ij}$ the couplings from Boltzmann learning, and $J'_{ij}$ the couplings from TAP or SusP, for calculating the $\Delta$ error. The temperature is not relevant and will always be set to 1.

<table>
<thead>
<tr>
<th>N</th>
<th>SusP</th>
<th>TAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.15</td>
<td>0.17</td>
</tr>
<tr>
<td>40</td>
<td>0.36</td>
<td>0.28</td>
</tr>
<tr>
<td>100</td>
<td>0.56</td>
<td>0.42</td>
</tr>
</tbody>
</table>

Table 6.1: Reconstruction error for SusP and TAP

This first table shows that the reconstruction is very hard, and the $\Delta$ is much higher than in the theoretical experiments. Susceptibility Propagation performs as well as TAP for $N=20$, but for larger dimension, TAP outperforms SusP.
6.2. RESULTS

These high error numbers ($\Delta = 0.2$) correspond to a poor reconstruction of couplings, where the sign of the interaction is usually reconstructed, but the strength can be severely damaged. For higher $\Delta$, the reconstruction can be meaningless.

In order to study more in details the reconstruction, we plot each value of coupling against the couplings found by Boltzmann learning. Here are the plots. For $N=20$

![Reconstruction error for $N=20$](image1)

![Reconstruction error for $N=40$](image2)

Figure 6.2: Reconstruction error for $N=20$

Figure 6.3: Reconstruction error for $N=40$

and 40, the SusP can compete with TAP, but for $N=100$, it is much worse: the reconstructed values are too low (in absolute value). This is due to the early stopping of the algorithm (the algorithm would diverge otherwise). The couplings of the systems are too high and the graph is too dense. There is a significant number of strongly connected loops that damage the reconstruction. TAP approximation is better than Susceptibility Propagation for this set of data.
CHAPTER 6. PERFORMANCE ON SIMULATED NEURAL DATA

Figure 6.4: Reconstruction error for N=100

(a) TAP

(b) SUSP
Chapter 7

Discussion

7.1 Further developments

The Susceptibility Propagation Algorithm is a fairly new algorithm which has not been used extensively on real data yet. Methods applied to Belief propagation algorithms, such as reinforcement and many other improvements [17] could probably applied to Susceptibility Propagation as well.

The algorithm was introduced in the context of an Ising model with spin variables. In order to apply it to other types of data, such as genetic data in which we take into account a level of expression instead of just a spin, one has to modify the whole algorithm, keeping the same principle. This modification is only extending an Ising model to another model and is in principle straightforward.

A way to improve efficiency of the algorithm would be to introduce higher order correlation functions. However, using only pairwise correlation is usually a good approximation, so the implementation of these higher order functions would only improve the efficiency of the algorithm in some rare cases. Moreover, it is not known whether the inversion of a model with higher order correlations would be possible, as the inversion with pairwise correlation was already a rather difficult trick.

Then, Susceptibility Propagation is only considering messages describing a log-likelihood of spins. For further description of the spin possibilities, one should consider the probability distribution instead of the log-likelihood. This means that twice as many messages are needed for spin variables, but could improve a lot the knowledge of the system when it is close to a spin glass phase, where both orientation of a group of spins are very likely or unlikely. This would give a Survey Propagation-like algorithm and could potentially be valuable [18].

7.2 Conclusion

In this report we described the recently introduced Susceptibility Propagation algorithm for statistical inference, which was seen to be more effective than state-of-the-art methods. Depending on the quality of the input data and the strength of the
couplings (or temperature), the inverse Ising problem can be solved with sometimes much better precision.

For generated data that is not too noisy and with a high enough temperature, the Susceptibility propagation algorithm will perform very well, even better than other algorithms like TAP or Sessak Monasson [14]. However, all of these algorithm perform well, and the precision in that case is not limited by these algorithms but rather by the quality of the data.

The interesting part is now at low temperatures where all the algorithms have trouble reconstructing the couplings when the system gets close to the spin-glass phase. It usually happens when spins have strong couplings between them, and can be seen in real systems.

The theoretical results are quite promising for this algorithm, as described in [13]. The algorithm seems to have a good behavior for small as well as large dimension N. However many factors can affect it: a small correlation error, the presence of external fields, strongly correlated loops. Unlike other algorithms, Susceptibility Propagation eventually fails (diverges), usually when the couplings are too high. To avoid that, we have shown that stopping the algorithm before it goes to a divergent state can give some good approximation to very low temperature reconstruction.

We have tested our algorithm on neural data which implies very strongly correlated neurons. The algorithm performs quite poorly when N is large, compared to the state-of-the-art approximation TAP. While preventing the algorithm from failing, the stopping procedure does not prevent the algorithm from giving bad approximations at very low temperatures. One should remember that the algorithm needs more computation than other approximation, and thus should give much better results to be competitive.

However, the study should not be limited to this only simulated neural data, and Susceptibility Propagation can perhaps be derived and apply to many other classes of problems, where its performances would eventually be much better.

Susceptibility Propagation is a very promising algorithm on sparse graphs, where we have shown it to reconstruct couplings much better than other approximation like TAP.

Even though the Susceptibility Propagation is a very interesting algorithm, it is not suited to deal with real neural data yet, and is very difficult to tweak due to the complicated nature of the messages. If applied to suitable sets of data, it can beat all other state-of-the-art methods.
Bibliography


Appendix A

Demonstrations

This appendix contains the demonstrations of the different equations used to derive the Susceptibility Propagation algorithm.

A.1 demonstrate $h_{i \rightarrow j} = h_i + \sum_{k \in \partial i \setminus j} u_{k \rightarrow i}$

\[
h_{i \rightarrow j} = \frac{1}{2} \log \frac{e^{h_i} \prod_{k \in \partial i \setminus j} q_{k \rightarrow i}(+1)}{e^{-h_i} \prod_{k \in \partial i \setminus j} q_{k \rightarrow j}(-1)} + \frac{1}{2} \log \frac{\sum_{k \in \partial i \setminus j} \log q_{k \rightarrow i}(1)}{\sum_{k \in \partial i \setminus j} \log q_{k \rightarrow j}(-1)}
\]

\[
h_{i \rightarrow j} = h_i + \sum_{k \in \partial i \setminus j} u_{k \rightarrow i}
\]
A.2 demonstrate \( u_{i \rightarrow j} = \tanh J_{ji} \tanh h_{i \rightarrow j} \)

\[
\begin{align*}
u_{i \rightarrow j} &= \frac{1}{2} \log \frac{e^{J_{ji}} p_{i \rightarrow j} + e^{-J_{ji}} p_{i \rightarrow j}}{p_{i \rightarrow j}} \\
\text{Hold } U &= \frac{e^{J_{ji}} p_{i \rightarrow j} + e^{-J_{ji}} p_{i \rightarrow j}}{p_{i \rightarrow j}} \left( \frac{p_{i \rightarrow j}}{p_{i \rightarrow j}} - \frac{p_{i \rightarrow j}(-1)}{p_{i \rightarrow j}(1)} \right) \\
\text{Then we have: } u_{i \rightarrow j} &= \frac{1}{2} \log \frac{1+U}{1-U} = \text{arctanh } U
\end{align*}
\]

\[
\begin{align*}
tanh u_{i \rightarrow j} &= U \\
&= \frac{e^{J_{ji}} - e^{-J_{ji}}}{e^{J_{ji}} + e^{-J_{ji}}} \times \frac{p_{i \rightarrow j}(1) - p_{i \rightarrow j}(-1)}{p_{i \rightarrow j}(1) + p_{i \rightarrow j}(-1)} \\
&= \tanh J_{ji} \sqrt{\frac{p_{i \rightarrow j}(1)}{p_{i \rightarrow j}(-1)}} \left( \sqrt{\frac{p_{i \rightarrow j}(1)}{p_{i \rightarrow j}(-1)}} - \sqrt{\frac{p_{i \rightarrow j}(-1)}{p_{i \rightarrow j}(1)}} \right) \\
&= \tanh J_{ji} \tanh \left( \log \sqrt{\frac{p_{i \rightarrow j}(1)}{p_{i \rightarrow j}(-1)}} \right) \\
tanh u_{i \rightarrow j} &= \tanh J_{ji} \tanh h_{i \rightarrow j}
\end{align*}
\]

A.3 demonstrate \( g_{i \rightarrow j,k} = \sum_{l \in \partial i \setminus j} v_{l \rightarrow i,k} + \delta_{i,k} \)

\[
\begin{align*}
g_{i \rightarrow j,k} &= \frac{\partial h_{i \rightarrow j}}{\partial h_k} \\
&= \frac{\partial h_i}{\partial h_k} + \sum_{l \in \partial i \setminus j} \frac{\partial u_{l \rightarrow i}}{\partial h_k} \\
g_{i \rightarrow j,k} &= \sum_{l \in \partial i \setminus j} v_{l \rightarrow i,k} + \delta_{i,k}
\end{align*}
\]
A.4 demonstrate \( v_{i \to j,k} = g_{i \to j,k} \tanh J_{ij} \frac{1 - \tanh^2 h_{i \to j}}{1 - \tanh^2 u_{i \to j}} \)

\[
v_{i \to j,k} = \frac{\partial u_{i \to j}}{\partial h_k} = \frac{\partial (\arctanh (\tanh J_{ji} \tanh h_{i \to j}))}{\partial h_k} = \frac{\partial (\tanh J_{ji} \tanh h_{i \to j})}{\partial h_k} \times \frac{1}{1 - (\tanh J_{ji} \tanh h_{i \to j})^2} = \tanh J_{ji} (1 - \tanh^2 h_{i \to j}) \frac{\partial h_{i \to j}}{h_k} \times \frac{1}{1 - \tanh^2 u_{i \to j}}
\]

\[
v_{i \to j,k} = g_{i \to j,k} \tanh J_{ij} \frac{1 - \tanh^2 h_{i \to j}}{1 - \tanh^2 u_{i \to j}}
\]

A.5 demonstrate \( m_i = \tanh (h_i + \sum_{j \in \partial i} u_{j \to i}) \)

\[
m_i = \tanh \left( H_i = \tanh \frac{1}{2} \log \frac{e^{h_i} \prod_{j \in \partial i} q_{j \to i} (+1)}{e^{-h_i} \prod_{j \in \partial i} q_{j \to i} (-1)} \right) = \tanh \left( \frac{1}{2} 2 h_i + \frac{1}{2} \left( \sum_{j \in \partial i} \log q_{j \to i} (+1) - \sum_{j \in \partial i} \log q_{j \to i} (-1) \right) \right) = \tanh \left( h_i + \sum_{j \in \partial i} \frac{1}{2} \log \frac{q_{j \to i} (+1)}{q_{j \to i} (-1)} \right) = \tanh (h_i + \sum_{j \in \partial i} u_{j \to i})
\]
A.6 demonstrate \( \chi_{ij} = \chi_{ij} g_{j\rightarrow i,j} + g_{i\rightarrow j,j} (1 - m_i^2) \)

where \( \chi_{ij} = \frac{\partial \tanh H_i}{\partial h_j} = \frac{\partial m_i}{\partial h_j} \)

\[ = \frac{\partial (h_i + \sum_{j \in \partial i} u_{j\rightarrow i})}{\partial h_j} (1 - m_i^2) \]

\[ = \frac{\partial (h_{i\rightarrow j} + u_{j\rightarrow i})}{\partial h_j} (1 - m_i^2) \]

\[ = (v_{j\rightarrow i,j} + g_{i\rightarrow j,j}) (1 - m_i^2) \]

So we have to prove that:

\[ v_{j\rightarrow i,j} (1 - m_i^2) = \chi_{ij} g_{j\rightarrow i,j} . \]

\[ \Leftrightarrow \tanh J_{ij} \frac{1 - \tanh^2 h_{i\rightarrow j}}{1 - \tanh^2 u_{i\rightarrow j}} \times (1 - m_i^2) = \frac{\tanh J_{ij} + \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i} - m_i m_j}{1 + \tanh J_{ij} \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i}} \]

By assessing \( J_{ij} = J_{ji} \), we can prove this equality. To simplify let us set:

- \( a = \tanh J_{ij} \)
- \( b = \tanh h_{i\rightarrow j} \)
- \( c = \tanh h_{j\rightarrow i} \)

\[ m_i = \tanh (h_{i\rightarrow j} + u_{j\rightarrow i}) = \frac{\tanh h_{i\rightarrow j} + \tanh h_{j\rightarrow i} \tanh J_{ij}}{1 + \tanh h_{i\rightarrow j} \tanh h_{j\rightarrow i} \tanh J_{ij}} \]

\[ m_i = \frac{b + ac}{1 + abc} \]

\[ m_j = \frac{c + ab}{1 + abc} \]

\[ \tanh u_{j\rightarrow i} = ac \]

Then the equation becomes:
A.7. DEMONSTRATE TANH $J_{ij} = \frac{C_{ij} - \text{TANH} H_{i,i-j} \text{TANH} H_{j,i}}{1 - C_{ij} \text{TANH} H_{i,i-j} \text{TANH} H_{j,i}}$

\[
a \frac{1 - c^2}{1 - a^2 c^2} (1 - \left( \frac{b + ac}{1 + abc} \right)^2) = \frac{a + bc}{1 + abc} - \frac{c + ab}{1 + abc} \times \frac{b + ac}{1 + abc}
\]

\[
\Leftrightarrow a \frac{1 - c^2}{1 - a^2 c^2} ((1 + abc)^2 - (b + ac)^2) = (a + bc)(1 + abc) - (c + ab)(b + ac)
\]

\[
\Leftrightarrow a(1 - c^2)((1 + abc)^2 - (b + ac)^2) = (1 - a^2 c^2)((a + bc)(1 + abc) - (c + ab)(b + ac))
\]

\[
\Leftrightarrow a(1 - c^2)(1 + a^2 b^2 c^2 - b^2 - a^2 c^2) = (1 - a^2 c^2)(a + ab^2 c^2 - c^2 a - ab^2)
\]

\[
\Rightarrow 1 + a^2 b^2 c^2 - b^2 - a^2 c^2 - c^2 - a^2 b^2 c^2 + b^2 c^2 + a^2 c^4 = 1 + b^2 c^2 - c^2 - b^2 - a^2 c^2 - a^2 b^2 c^4 + a^2 c^4 + a^2 b^2 c^2
\]

\[
\Leftrightarrow 0 = 0
\]

We have proven that $v_{j \rightarrow i,j}(1 - m_i^2) = \chi_{ij} g_{j \rightarrow i,j}$, and so:

\[
\chi_{ij} = \chi_{ij} g_{j \rightarrow i,j} + g_{i \rightarrow j,i}(1 - m_i^2).
\]

A.7 demonstrate tanh $J_{ij} = \frac{C_{ij} - \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}}{1 - C_{ij} \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}}$

where $C_{ij} \rightarrow \frac{\chi_{ij} - g_{i,j,i}}{g_{j,i,j}} + m_i m_j$

\[
\chi_{ij} = [\text{tanh} J_{ji} + \text{tanh} h_{i,i-j} \text{tanh} h_{j,i} - \text{tanh} H_i \text{tanh} H_j] g_{j,i,j} + g_{i,j,i}(1 - \text{tanh}^2 H_i)
\]

\[
\text{tanh} J_{ji} + \text{tanh} h_{i,i-j} \text{tanh} h_{j,i} = \frac{\chi_{ij} - g_{i,j,i}(1 - \text{tanh}^2 H_i)}{g_{j,i,j}} + \text{tanh} H_i \text{tanh} H_j
\]

\[
\Leftrightarrow C_{ij}
\]

\[
\text{tanh} J_{ji}(1 - C_{ij} \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}) = C_{ij} - \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}
\]

\[
\text{tanh} J_{ji} = \frac{C_{ij} - \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}}{1 - C_{ij} \text{tanh} h_{i,i-j} \text{tanh} h_{j,i}}
\]