CHALMERS

Message-passing Methods on Loopy Factor Graphs Based on Homotopy and Reweighting

Master's Thesis in the Master Degree Programme, Electrical Engineering

KARL-MAGNUS DAHLÉN
CHRISTOPHER LINDBERG

Department of Signals and Systems
Division of Communication Systems and Information Theory
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 2012
Master's Thesis EX046/2012

Abstract

Belief propagation (BP) is an algorithm with a wide range of applications where it is used to solve inference problems defined on probabilistic graphical models, e.g. computing the marginal distributions of a distribution function defined on a factor graph. It belongs to a family of algorithms called message passing algorithms, since it is updated by passing messages over the edges of the graphical models on which it is defined. Problems arise when the graph contains cycles, or loops, since there is only a guarantee of convergence to the exact solution when the graph is of a tree structure. This thesis studies the convergence properties on Ising models of a proposed modification of the standard BP algorithm based on homotopy, where the factor graph is gradually changed from a solvable structure into a loopy graph. We also investigate how well it compares to the uniformly reweighted BP (URW-BP). The performance of the algorithms are tested by running Monte Carlo simulations and computing the Kullback-Leibler divergence (KLD) between the exact and the approximate solutions, and also by computing bit and word error rates. Additionally we transfer these methods to a tracking problem application over a sensor network. The Monte Carlo simulations on the Ising model indicate that the URW-BP algorithm exceeds the performance of the standard BP, and the homotopy methods seem to perform in parity with serial updating BP, but conserve the possibility of using distributed computations. Our results from the tracking problem simulations show that the URW-BP on a loopy graph performs equally well, or even better than the standard BP on a tree graph.

Acknowledgments

Firstly, we would like to thank our thesis advisor Assistant Professor Henk Wymeersch for the motivation and guidance he has given us during the work. The weekly meetings were always something to look forward to, to get some interesting comments and views on how our work was progressing. Henk's support made the work more interesting and rewarding than we ever thought it would be when we began writing this thesis, and Henk most certainly increased our fascination of science. We would also like to thank Lennart Svensson for initiating the contact with Henk, for giving advise and for interesting discussions about the research. Thanks also to everyone else in the Communication Systems group who has been involved in the research in any way, it has been a privilege to work with this group. Finally, we would like to thank our families and beloved ones for the unwavering love, support and encouragement you have given us. We would like to conclude with a quote from Shay Maymon's doctoral thesis: "The thesis ends; Research continues forever".

Contents

Chapter 1

Introduction

Belief propagation (BP) is used to calculate marginal distributions of an underlying joint density function defined on graphs such as factor graphs, Markov random fields and Bayesian networks [1]. It is actually true that a number of different methods are special cases of the BP algorithm. Such special cases include for example the Kalman filter, the forward-backward algorithm, the Viterbi algorithm and iterative decoding algorithms for Gallager codes [2], and turbo codes [3], (for further information, see [4], [5] and [6]). Due to the high complexity of computing these marginal distributions by using the definition of the marginals or other exact inference algorithms, the BP algorithm is used in a wide range of applications, from artificial intelligence [7], to various applications in the field of communication [1]. But the algorithm, which can also be seen as finding fixed points of the Bethe approximation to the free energy [8], is only exact in some special cases, i.e. when the algorithm finds the global minimum of the approximation to the free energy. This special case corresponds to a finite tree on a factor graph [9]. For a general graph structure, however, there is no guarantee for convergence to the exact solution [9]. In the alternative interpretation of an optimisation problem, this would mean that the problem may not be of a convex nature. Therefore, this thesis aims at exploring whether the performance of the BP algorithm on loopy factor graphs can be improved by applying some different homotopy and reweighted methods to an Ising model, which has been shown to improve convergence of BP [10]. Furthermore, the BP algorithm is applied to a tracking problem where a network of sensors producing Gaussian measurements is mapped on the factor graph and serves as the underlying model.

1.1 Purpose

The motivation for this thesis is the unreliability of BP when used on factor graphs with cycles. Thus, this thesis aims at exploring how the algorithm behaves on these loopy factor graphs and also how the performance can be improved by using variations of the original BP algorithm to solve the desired inference problem. We will investigate a homotopy approach to the edges of the factor graph, and also implement a URW-BP. The main contribution is in doing Monte Carlo simulations to test and verify the variations of BP on the Ising model, and applying the most promising modified algorithm on a sensor network. The results from the Monte Carlo simulations will then be compared with the exact marginal probabilities to provide a measure of performance, in several different ways. The thesis will hopefully increase our understanding of why the BP algorithm performs badly on these particular Ising models, and relate these results to the tracking

problem application. The aim of the study of the tracking problem application is to relate the tracking model to the Ising model and compare the results. The main goal is to compare the chosen modified algorithm and the original algorithm to see if the we can transfer the results from one model to the other and to find out if the variational method can provide better solutions and in that case analyse at what cost.

1.2 Methodology

Given the theoretical statement of the belief propagation algorithm, we implemented it in a MATLAB environment. The code was implemented such that it allowed for great variation and freedom in constructing the Ising models, since this would facilitate the future modifications of the algorithm. To evaluate the performance of the different methods, some kind of measure was needed. Consequently, the Kullback-Leibler divergence was determined to be the appropriate measure for assessment of the algorithms, but we also looked at bit error rates. Following this, a number of tests were designed and run to test the algorithms' performances on the Ising model. The acquired results were then used in order to determine which modifications could be used in a real-world application of message-passing algorithms. Moving from the Ising models to the tracking problem application required modifications of the algorithms to make the variables take states on a grid rather than binary values. Again, the KL divergence measure was used in the evaluation process of the methods, but as this model was more complex, two other relevant measures were developed. To test how well the nodes converged to the same opinion, we computed the disagreement among the nodes. Also, the error in the expected value of the variables under the beliefs as the probability measure was considered. This was computed with respect to the expected value according to the true marginals, and served then as a complement to the probabilistic distance in the form of KL divergence.

1.3 Main findings

In this thesis we have studied the performance and convergence properties of a number of message-passing methods, including belief propagation, on loopy factor graphs through Monte Carlo simulations. We have found that particularly the URW-BP seems to be an appropriate choice of modification to the standard BP algorithm on Ising models in order to increase performance but at the same time preserve the important property of parallelism of the computations. This argument, it turns out, can also be applied to other models than Ising models. Experiments on a tracking problem, showed that the results improved the performance on this Gaussian model as well.

1.4 Thesis outline

The problem formulation in this thesis could be divided into two subproblems. The first problem concerns investigation of the Ising model on factor graphs. Here, we will apply the standard BP, as well as the homotopy approach and the URW-BP. The second part deals with the application of BP in a network of sensors. The thesis is divided in four chapters:

1. Chapter 1 introduces the topic and the relevance of our research. It also gives a brief background to the problem and shortly describes the algorithm used.

2. Chapter 2 gives a more elaborate description of the factor graph and the Ising model associated with the problem. The method of computing exact marginal distributions is also explained. We describe in detail the implementation of the message-passing algorithms in focus, and how they were derived from the traditional belief propagation algorithm.

3. Chapter 3 first explains how the Monte Carlo simulations were done using the algorithms on different sets of parameters in the Ising model. It also gives an interpretation to these cases in an informal and intuitive way. Furthermore, we give the results from the Monte Carlo simulations that were carried out. To make it easy to compare the performance of the different methods, the results are grouped according to which distribution the model parameters were generated from. The results are to a great extent supported by graphs and figures.

4. Chapter 4 formulates the tracking problem application and how belief propagation could be used to determine the position of an object on a grid. A detailed description of the implementation and the method is given and the different performance measures used to assess the message-passing methods are presented. We also briefly give an example of how prediction can be used to improve the performance. Lastly, the results are given with support from figures and plots along with a discussion.

5. Chapter 5 concludes this thesis by summing up our results and stating what we achieved during the work, and how it has improved our knowledge about message-passing algorithms.

Chapter 2

Belief propagation on factor graphs

2.1 Basic factor graphs

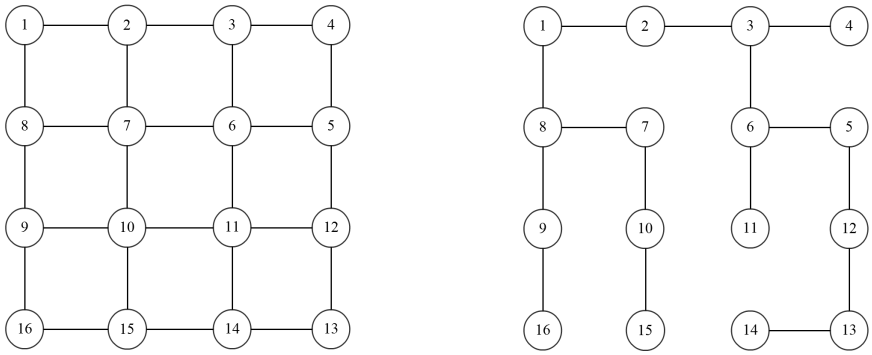


Figure 2.1: (2.1a) shows a simplified factor graph with cycles for an Ising model with $N = 16$, while (2.1b) shows a simplified factor graph with a tree structure for a similar model. All the variable vertices are shown as circles. Each vertex is associated with a local parameter θ_n and all edges with a bipartite parameter θ_{nm} describing the dependency relation between two variables.

A factor graph $\mathcal{G}$ consists of three main elements $(\mathcal{V},\mathcal{F},E)$. The set of variable vertices, or nodes, are denoted by $\mathcal{V}$ and the set of edges connecting the vertices E. There is also the set of factor vertices $\mathcal{F}$. We construct the graph with consideration to our model such that each vertex corresponds to a variable, the factor vertices describe the relation between two variables in an undirected way and the edges connect the variable vertices with the factor vertices. Given a probability density function $f(x_1,x_2,\ldots,x_N)$ which can be represented by a factorisation of the function in the following way

$$f(x_1,x_2,\ldots,x_N) = \prod_{i=1}^{N} \phi_i(x_i) \prod_{(m,n)\in E} \psi_{mn}(x_m,x_n) \tag{2.1}$$

This way each variable x_m will be associated with a local factor $\phi_m \in \mathcal{F}$ and each interacting pair of vertices x_m and x_n with a common factor $\psi_{mn} \in \mathcal{F}$, and thus this creates a probabilistic map onto our factor graph. Henceforth when investigating the Ising model, we will frequently work with the following notations: the set of variables $\mathbf{x} = \{x_i : i \in \mathcal{V}\}$ and the set of edges $E = \{(n,m) : \theta_{n,m} \neq 0\}$. For convenience we group

the factor vertices together with the edges and only specify the parameters θ_{mn}, not the whole mutual factor functions, and similarly we specify only the local parameters θ_n of the local factor functions. Now, if there is a path from one vertex back to itself over a set of edges, then that set forms what is called a cycle. When there exists no cycles on a graph, then the graph is said to have a tree structure. The set of neighbouring vertices to vertex m is called $\mathcal{N}_m$, i.e. this set contains all the vertices connected to vertex m with a nonzero parameter θ_{mn}.

2.2 The Ising model

The model that nowadays is called the Ising model was first used by Ernst Ising in his doctoral thesis in 1924, after it was suggested by his doctoral advisor Wilhelm Lenz in 1920 [11]. What Ising did, was that he tried to explain observed data from ferromagnetic materials using this model. Hence, the term spin is used to describe the two states that a variable can assume in the Ising model. We introduce a probability space $(\Omega,\mathcal{F}_\sigma,P)$ to this model. Here, Ω is the set of all sequences of outcomes $\omega = (\omega_1,\omega_2,\ldots,\omega_N)$, $\mathcal{F}_\sigma$ is the σ-algebra of Ω and P is the probability measure assigned to Ω. Now, ω_m can have either the up $(+)$ or down $(-)$ spin. We can let the map $x_m : \omega_m \to \{-1,1\}$ be such that if $\omega_m = +$ then $x_m(\omega) = 1$, and if $\omega_m = -$ then $x_m(\omega) = -1$. Introduce, for each ω, the energy $U(\omega)$

$$U(\omega) = -J\sum_{m,n} x_m(\omega)x_n(\omega) - mH\sum_m x_m(\omega) \tag{2.2}$$

where m,n are neighbouring points. The assumption made here is that only neighbouring points affect each other, and therefore only neighbouring interactions need to be considered, i.e. $J = 0$ for (n,m) not neighbours. The constant J characterises the type of interaction that is in effect between the points. If $J > 0$ then two neighbouring points will tend to the same spin. This case is therefore called attractive, while the other case when $J < 0$ is called repulsive. The constants in the second term of equation (2.2) is due to effects from external magnetic fields. Ising then introduced the probability measure on Ω to be given by

$$p(\omega) = \frac{1}{Z}\exp\left(-\frac{1}{kT}U(\omega)\right) \tag{2.3}$$

From this we then get the model that we use in this thesis. Changing the notation a bit and introducing different interaction constants between all pairs as well as different external effects to all vertices yields the following expression

$$p(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_{n=1}^{N}\theta_n x_n + \sum_{(n,m)\in E}\theta_{nm}x_n x_m\right) \tag{2.4}$$

where θ_n,$\theta_{nm} \in \mathbb{R}$, $E = \{(n,m) : \theta_{n,m} \neq 0\}$ and $Z \in \mathbb{R}^+$ is a normalising constant. When testing the algorithm on this model, the parameters θ_n and θ_{nm} are assumed to be known, therefore they are determined prior to any calculations.

2.3 Belief propagation

To obtain the marginal distributions of a joint distribution $p(\mathbf{x})$, one can use the definition of the marginal distribution. It is done, by summing in the discrete case (if we don't want

to use the Stieltjes integral) or integrating in the continuous case, the joint probability distribution function over all possible values of the variables except the variable regarded in the marginal distribution. For the discrete Ising model this is done in the fallowing way

$$p_i(x_i) = \sum_{\mathbf{x}\setminus\{x_i\}} p(\mathbf{x}) \qquad \text{for } i = 1,2,\ldots,N \tag{2.5}$$

It can be shown, that the complexity of these calculations are proportional to $\mathcal{O}(2^N N^3)$, which makes this method impractical with just a few variables. The dominating term $\mathcal{O}(2^N)$ arises from the N number of variables and the two possible states that each variable can take. Comparing with a model also of N variables but with s number of states, the complexity of finding the marginals with this brute force approach would be dominated by a term $\mathcal{O}(s^N)$, which makes the approach intractable for many states.

The BP algorithm, which is a message passing algorithm, works instead by letting the variables, represented as vertices in the factor graph, communicate with each other by sending information in the form of messages, which we call $\mu_{m\to n}(x_n)$ where x_n is the variable associated with vertex n. These messages contain the information sent by the transmitting vertex m influencing the destination vertex n and they are sent over the edges of the factor graph. If there is a message $\mu_{m\to n}(x_n)$ there is also a message $\mu_{n\to m}(x_m)$. After a certain number of iterations, $t_{\max}$, the incoming information in each vertex is evaluated and then the beliefs, i.e. the approximate marginal distributions, are calculated based on this.

This method requires a factorisation of $p(\mathbf{x})$ so that the problem can be formulated on a factor graph. A nice property of BP is that the factorisation can be done on an unnormalised function $\tilde{p}(\mathbf{x})$ such that $\frac{\tilde{p}(\mathbf{x})}{Z} = p(\mathbf{x})$, where Z is the normalisation constant. BP is valid for a general joint distribution $p(\mathbf{x})$ of either continuous or discrete variables $\mathbf{x}$, and general factorisations. In this thesis we consider only mutual factors of two variables, and local factors of one variable. This is the case when dealing with the Ising model, and also for the extension to the tracking model. For further reading of belief propagation on more general factorisations and factor graphs, please read [1] and [5]. For a discrete factorised model with local factors $\phi_i(x_i)$ and mutual factors $\psi_{mn}(x_m,x_n)$, where $(m,n) \in E$ and $i \in V$ according to (2.1), the message in the factor graph will be updated at each iteration $t+1$, and the beliefs $b_n(x_n)$ will be computed, in the following way

$$\mu_{m\to n}^{(t+1)}(x_n) \propto \sum_{x_m} \left(\psi_{mn}(x_m,x_n)\phi_m(x_m) \prod_{l\in\mathcal{N}_m\setminus\{n\}} \mu_{l\to m}^{(t)}(x_m) \right) \tag{2.6}$$

$$b_n(x_n) \propto \prod_{l\in\mathcal{N}_n} \mu_{l\to n}^{(t_{\max})}(x_n)\phi_n(x_n) \tag{2.7}$$

In the continuous case the summation in (2.6) will be replaced by an integration. After factorisation of (2.4), the factors in the Ising model will be $\phi_i(x_i) = \exp(\theta_i x_i)$ and $\psi_{mn}(x_m,x_n) = \exp(\theta_{mn} x_m x_n)$, where $(m,n) \in E$ and $i \in V$. By calculating approximate marginals this way, we utilize a method whose complexity scales linearly as a function of the number of variables, [12], and quadratically with the number of states each variable can take, [13].

2.3.1 Implementation

For all directed pairs $(n,m) \in E$ a message $\mu_{m\to n}(x_n)$ is introduced. $\mu_{m\to n}(x_n)$ then denotes a messages from node variable x_m to node variable x_n. This means that the algorithm passes messages in each direction over the edges in the factor graph. These messages are functions such that $\mu : \{-1,1\} \to [0,1]$. The algorithm starts at iteration index $t = 0$ and then updates the messages in the graph $t_{\max}$ times as shown below in equation (2.8). Also, the beliefs are then proportional to the expression on the right hand side of equation (2.9).

$$\mu_{m\to n}^{(t+1)}(x_n) \propto \sum_{x_m} \left(\exp(\theta_m x_m + \theta_{nm} x_m x_n) \prod_{l \in \mathcal{N}_m \setminus \{n\}} \mu_{l\to m}^{(t)}(x_m) \right) \tag{2.8}$$

$$b_n(x_n) \propto \prod_{l \in \mathcal{N}_n} \mu_{l\to n}^{(t_{\max})}(x_n) \exp(\theta_n x_n) \tag{2.9}$$

Here, $\mu_{m\to n}^{(t)}$ denotes the message from x_m to x_n at iteration t and b_n denotes the belief calculated for vertex n in the Ising model. As initial condition we set $\mu_{m\to n}^{(0)}(x_n) = 1$ for all $(n,m) \in E$.

The updating according to equation (2.8) can be done in different ways. Either all pairs or just some pairs can be updated in each iteration, and also different pairs may be updated each iteration. In this thesis two different updating schemes are investigated. The first one is an updating scheme such that all the pairs in E are updated at each iteration t. We will refer to this scheme as the flooding updating scheme and in the results this will be denoted F. The benefit of the flooding updating scheme is that it can be implemented in a distributed fashion, where one computational node can compute one message, as the messages only depend on the existing information from the previous iteration. If the letter F is written after the name of a method we mean that the flooding updating scheme is used for the method. The other method of updating that is tested in this thesis is to only update one message each iteration, hence always using the newest information available at each iteration. In this case parallel computations cannot be implemented, as the computation of each message requires the incoming messages to contain information from the same iteration. Thus, this updating scheme is computed in a serial fashion instead. When testing this method, the messages are always updated equally many times. This method also gave the possibility to use different paths of updating through the model, which might affect the outcome of the results. Though, throughout our tests all the messages are updated in one cycle and in the same order, and we make no special investigation on certain updating paths. We will refer to this as the serial updating scheme and denote it S. If the letter S is written after the name of a method we mean that the serial updating scheme is used for the method.

Note that in the results from using serial updating the notation $t_{\max}$ will be slightly abused in the sense that $t_{\max}$ will denote the number of times each message has been updated. To make the beliefs and messages obey the second axiom of probability, they both have to sum up to 1 as explained below

$$\sum_{x_n} \mu_{m\to n}^{(t+1)}(x_n) = 1 \tag{2.10}$$

$$\sum_{x_n} b_n(x_n) = 1 \tag{2.11}$$

As stated before, the beliefs will only converge to the exact probabilities every time if the factor graph is a tree, i.e. the set E contains no cycles. For convenience, let $T \subset E$ denote the set of all (n,m) belonging to the tree structure.

2.3.2 Homotopy methods

The continuous maps $p_0, p_1 : X \to Y$ are called homotopic if there is a continuous map $F(x,\lambda) : X \times \Lambda \to Y$, where $x \in X$ and $\lambda \in \Lambda = [0,1]$, such that p_0 and p_1 are continuously connected through this map, i.e. $F(x,0) = p_0(x)$ and $F(x,1) = p_1(x)$ for all $x \in X$, [14]. The map F is called a homotopy from p_0 to p_1. For all $\lambda \in \Lambda$ there is a distribution function $p_\lambda(\mathbf{x}) = F(\mathbf{x},\lambda)$. To apply the homotopy approach to the Ising model, $p_0(\mathbf{x})$ is chosen such that it is possible to compute the exact marginal distribution $p_{0,n}(x_n)$ with BP. When using this as a starting problem in the algorithm and iteratively increasing λ, the problem converges to $p(\mathbf{x})$ as $\lambda \to 1$. The goal of utilising homotopy is, by introducing memory in the messages between the different p_λ's, to access more information about the model and this way decrease the error of the beliefs. Ideally, the BP algorithm will converge to the exact marginal distributions, i.e. $b_{\lambda,n}(x_n) \to p_{\lambda=1,n}(x_n)$ as $\lambda \to 1$.

Two different homotopies are investigated in this thesis. In both cases the function corresponding to $\lambda = 1$, i.e. $p_1(\mathbf{x})$, is equal to the function found in equation (2.4). In the first case, the factor graph is grown from a tree into a full graph with cycles as λ is increased from 0 to 1. The second case is initiated with all $\theta_{nm} = 0$, i.e. there are no connections at all, and then all edges are increased in a parallel sense. The corresponding p_0's equal the following functions

$$p_0(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_{n=1}^{N}\theta_n x_n + \sum_{(n,m)\in T}\theta_{nm}x_n x_m\right) \tag{2.12}$$

$$p_0(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_{n=1}^{N}\theta_n x_n\right) \tag{2.13}$$

Consequently, by combination of equation (2.4) and equations (2.12), (2.13) the following equation is acquired for the homotopy from $p_{\lambda=0}$ to $p_{\lambda=1}$, with λ increasing from 0 to 1.

$$p_\lambda(\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_{n=1}^{N}\theta_n x_n + \sum_{(n,m)\in T}\theta_{nm}x_n x_m + \lambda\sum_{(n,m)\in E\setminus T}\theta_{nm}x_n x_m\right) \tag{2.14}$$

The set T is the set of all (n,m) belonging to the tree of the factor graph in the first case, but in the second case there are no edges at all at the start of the algorithm, hence $T = \emptyset$. We will refer to the first homotopy as method T and the second as method H. In this thesis both methods are tested with the flooding and the single updating scheme. This yields four different homotopy methods, and in the figures in the results they will be referred to as HF, HS, TF and TS, where the letters F and S denote the type of updating scheme coupled with each method.

The implementation is done by partitioning λ into K parts on the interval [0,1] such that $0 = \lambda_0 < \lambda_1 < \lambda_2 < , \ldots , < \lambda_{K-1} = 1$ and $\lambda_k = k/(K-1)$, $k = 0, \ldots, K-1$. To each λ_k we then have a corresponding probability mass function $p_{\lambda_k}(\mathbf{x})$. BP is then run for each $p_{\lambda_k}(\mathbf{x})$, hence $K \cdot t_{\max}$ steps of BP are performed with $t_{\max}$ iterations for each λ_k. This yields messages $\mu^{(t)}_{m\to n,\lambda_k}(x_n)$ for all $(n,m) \in E$ and for these messages we set the initial condition $\mu^{(0)}_{m\to n,\lambda_0}(x_n) = 1$. Equation (2.15) shows how the messages are computed using the homotopy approach, and equation (2.16) shows what the corresponding beliefs to each λ_k will be according to the homotopy-based BP.

$$\mu^{(t+1)}_{m\to n,\lambda_k}(x_n) \propto \sum_{x_m} \left(\exp(\theta_m x_m + \tilde{\theta}_{nm,k} x_m x_n) \prod_{l \in \mathcal{N}_m \backslash \{n\}} \mu^{(t)}_{l\to m,\lambda_k}(x_m) \right) \tag{2.15}$$

$$b_{n,\lambda_k}(x_n) \propto \prod_{l \in \mathcal{N}_n} \mu^{(t_{\max})}_{l\to n,\lambda_k}(x_n) \exp(\theta_n x_n) \tag{2.16}$$

where

$$\tilde{\theta}_{nm,k} = \begin{cases} \theta_{nm} & , \text{if} (n,m) \in T \\ \lambda_k \theta_{nm} & , \text{if} (n,m) \in E \backslash T \end{cases} \tag{2.17}$$

When increasing λ the messages are set such that $\mu^{(0)}_{m\to n,\lambda_{k+1}}(x_n) = \mu^{(t_{\max})}_{m\to n,\lambda_k}(x_n)$. As mentioned before, messages are only introduced between nodes where there is a nonzero connection, i.e. a nonzero θ_{mn} and in this case a nonzero $\tilde{\theta}_{mn,k}$. When implementing the homotopy method there is no need to introduce messages for the pairs $(n,m) \notin T$ during iterations on λ_0, since the constants $\theta_{nm,k}$ would then be zero for the pairs $(n,m) \in E \backslash T$ according to equation (2.17). The implementation could then be done such that messages are introduced only for the pairs $(n,m) \in T$ when running the algorithm for λ_0 and as λ is gradually increased to λ_1, new messages for the pairs $(n,m) \in E \backslash T$ with initial condition $\mu^{(0)}_{m\to n,\lambda_1}(x_n) = 1$ are introduced. But the same result is also achieved by introducing messages for all pairs $(n,m) \in E$ even for λ_0 with initial condition $\mu^{(0)}_{m\to n,\lambda_0}(x_n) = 1$.

2.3.3 Reweighted methods

Another variational method to compute the marginals is to use a family of reweighted message-passing algorithms. These algorithms might solve some of the convexity issues that arise when a graph contains cycles, and have been shown to have better convergence properties in some cases, [10]. We acquire this family of tree-reweighted algorithms by introducing weights $\boldsymbol{\rho} = \{\rho_0, \rho_1, \ldots, \rho_M\}$, where M is the number of nonzero interactions in the graph, and each ρ_i is paired with one of the interaction parameters θ_{mn}. We also divide the incoming messages to vertex m by $\left(\mu^{(t)}_{n\to m}(x_m)\right)^{1-\rho}$, [15]. But, to avoid the high-dimensional optimisation problem over $\boldsymbol{\rho}$ we impose a restriction to a fixed $\rho_i = \rho \; \forall i$, as suggested in [16]. Hence, this message passing algorithm is called uniformly reweighted belief propagation (URW-BP).

The implementation of URW-BP requires some modifications of the original BP algorithm. The inference calculations are done instead according to the following updating

equations for the messages and the beliefs.

$$\mu_{m\to n}^{(t+1)}(x_n) \propto \sum_{x_m} \left(\exp(\theta_m x_m + \frac{1}{\rho}\theta_{nm} x_m x_n)) \prod_{l\in\mathcal{N}_m\setminus\{n\}} \frac{\left(\mu_{l\to m}^{(t)}(x_m)\right)^{\rho}}{\left(\mu_{n\to m}^{(t)}(x_m)\right)^{1-\rho}} \right) \tag{2.18}$$

$$b_n(x_n) \propto \prod_{l\in\mathcal{N}_n} \left(\mu_{l\to n}^{(t_{\max})}(x_n)\right)^{\rho} \exp(\theta_n x_n) \tag{2.19}$$

Here ρ is a constant that should be chosen between 0 and 1. Note that $\rho = 1$ corresponds to the original BP algorithm. The theory says that there exists a value of $\rho < 1$ that gives a better approximation of the marginals than the original algorithm. This optimal ρ can be approximated as $\rho^* \approx \frac{|V|-1}{|E|}$, where $|V|$ is the number of the vertices in the graph and $|E|$ is the number of edges in the graph, [15].

Chapter 3

Monte Carlo simulations on Ising models

BP was implemented to calculate the beliefs of the Ising models on a factor graph structure using MATLAB. The implementation of the homotopy method was done by creating a vector of values in MATLAB to represent a partition of λ, and a similar approach was done to implement the URW-BP. For generation of tree graphs in the homotopy approach, an algorithm generating random tree graphs was created. To test the performance of the algorithms, a sequence of Monte Carlo simulations were done on the two model parameters θ_n and θ_{mn}. Tests were made in both "normal-" and log-domain, to ensure that the numerical issues that could arise from the software would not affect the results. When generating random sets of the parameters θ_{mn} and θ_n, MATLAB's random number generator `rand()` was used. The plots depicting the results provided by the computations were all produced by the plot functions `plot()`, `semilogy()` and `contourf()` embedded in MATLAB.

3.1 Measure of performance

To measure performance of the computed approximate marginal distributions in relation to the true marginals the Kullback-Leibler divergence was used. This is a probability distance from the distribution $b(x)$ to the distribution $p(x)$ defined by the following expression

$$D_{\mathrm{KL}}(b||p) = \sum_x b(x) log \frac{b(x)}{p(x)} \tag{3.1}$$

where $b(x)$ and $p(x)$ are probability mass functions. With this measure the distance in probability from the beliefs to the exact marginals can be computed. This measure was visualized by averaging over all nodes, the KL divergence from each node to its respective true marginal.

Other measures used to test the performance include the bit error- and the word error rates. From both the true marginals and the beliefs the most likely state of a variable can be computed. If this state is not the same in the true marginals as in the beliefs for a variable this gives a bit error. The number of incorrect bits divided by the total number of bits in a sequence is then defined as the bit error rate. If one or more bit errors are detected, then the method used generated a word error, i.e. the word error is one if at

least one state for a variable is different in the beliefs and the true marginals, and zero if all the states are the same. The number of trials producing word errors divided by the total number of trials is then the word error rate. Both the KL divergence, the bit error rate, and the word error rate were measured on all the methods in the simulations. For the computation of the KL divergence, the logarithmic operator with the base 2 was used.

3.2 Simulations

To generate the results Monte Carlo simulations of the model were performed. In the simulations Ising models of 9 variables were used to make it easy to compute the exact marginals with the brute force approach, and hence made it possible to compute the error of the beliefs with respect to the exact marginal distributions. The simulations were performed with all pairwise connections nonzero, i.e. the set E contained all pairs in the square Ising model.

The goal of the simulations was to find out for which types of Ising models a certain method performed well and evaluate which method that gave the best results according to bit error, word error and KL divergence measures. For the homotopy methods the goal was to find out how to choose K, the number of partitions of λ, and $t_{\max}$, the number of iterations to run the BP algorithm for each λ, to achieve the best results. For the URW-BP the goal was to find the ρ which gave the best result and compare this to the theory. Also common for the modifications to BP, we wanted to find cases where they outperformed the original algorthm.

To investigate how the algorithms would behave in different cases, four different types of distributions were distinguished between, from which the parameters θ_{mn} and θ_n were generated.

- *Distribution 1*
  Both θ_{nm} and θ_n were chosen uniformly from 40 different equally spaced values in the interval [-4,4], which then yielded a total of 1600 trials. Distribution 1 yielded the most simple set of probability distributions from which the parameters θ_n and θ_{nm} were drawn in our tests. These tests served as a first hint of how the relation between the strength (magnitude) of the local information θ_n and the mutual information θ_{nm} affected the error of the beliefs with respect to the true marginals.

- *Distribution 2*
  $\theta_{nm} = 10$, $\forall$ $(n,m) \in E$, while each $\theta_n = \pm 10$ with $+10$ and -10 appearing with equal probability. A total of 1500 trials were made using this distribution. The second distribution from which the model parameters were generated, can be described in an intuitive and slightly naïve way as a case when some of the variables are very confident in being -1 and some being very confident in being $+1$, but the mutual parameters θ_{nm} are trying to force everyone to the same value (be it -1 or $+1$). This effect is achieved by letting the magnitude of all values, i.e. $|\theta_n|$, of the parameters be equal to a large number, in our case we choose 10, but with different signs. Then, we let θ_{nm} be of equal magnitude, but always positive, to make it enforce uniform values over the whole model. The purpose with tests from this

distribution was to generate cases when the BP algorithm almost always failed, and see if the homotopy or reweighted methods could improve the performance.

- *Distribution 3*
  $\theta_n = 10,\ \forall\ n$, while each $\theta_{nm} = \pm 10,\ \forall\ (n,m) \in E$, with $+10$ and -10 appearing with equal probability. A total of 1500 trials were made using this distribution. To interpret Distribution 3 in the same sense as we did Distribution 2, this means that the variables had very strong opinions about the local value being $+1$, but also a strong opinion about the neighboring node variables since $|\theta_{nm}|$ was quite large for all interactions $(n,m) \in E$. The mutual parameters θ_{nm} were either $+10$ or -10, which means that some parameters tried to enforce opposite signs while others tried to enforce same signs between two nodes. As with the tests on distribution 2, these tests aimed at finding scenarios where the homotopy or reweighting methods could exceed the performance of the standard BP algorithm.

- *Distribution 4*
  Each $\theta_{nm} = Z$, $\theta_n = Z$ where Z is a random variable $Z = XY$, and where in turn X was a continuous random variable such that $X \in \mathcal{U}(3,4)$ and Y was a discrete random variable of the Rademacher distribution. 1500 trials of this distribution were tested. The fourth and last case which was investigated was also the most general case studied. The variables could have any local information about the sign, and also different magnitude, while the mutual information could be either repulsive or attractive with varying magnitude.

3.3 Results

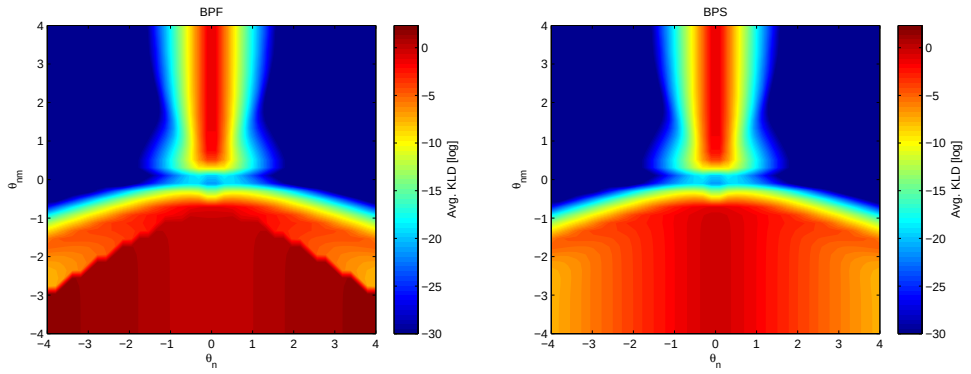


Figure 3.1: Plots of the average KLD in $\log_e$ scale for the belief propagation algorithm with the flooding updating scheme in figure (a) and the serial updating scheme in figure (b). Here $t_{\max} = 100$ for both the flooding and serial updating schemes. The data to generate the contour plots originates from computations on generated values from distribution 1.

A smaller study of the convergence rates of the BP and its variations for the Ising model was made. We found that in the convergent cases of the BP algorithm, the serial updating scheme seemed to converge faster than the flooding. In these cases less than

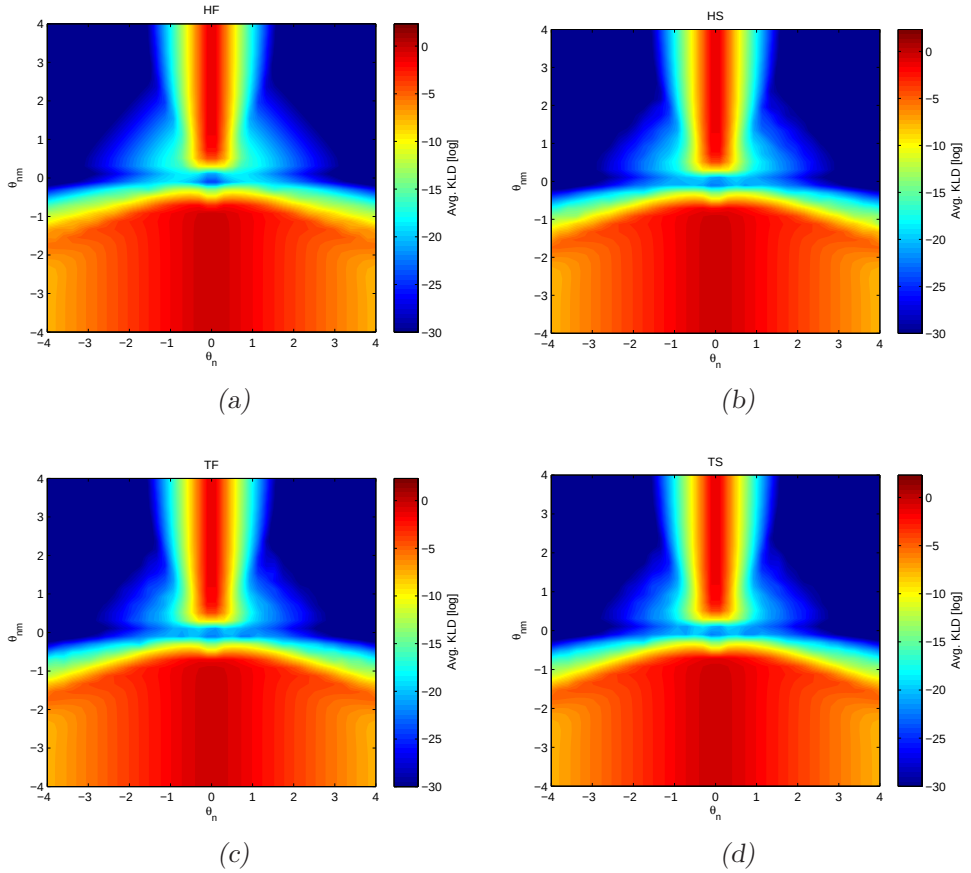


Figure 3.2: Figures (a), (b), (c) and (d) show plots of the average KLD in $\log_e$ scale for all the homotopy methods with the flooding and the serial updating scheme. $K = 100$ and $t_{\max} = 1$ for all homotopy variations of BP. These figures shows computations for distribution 1.

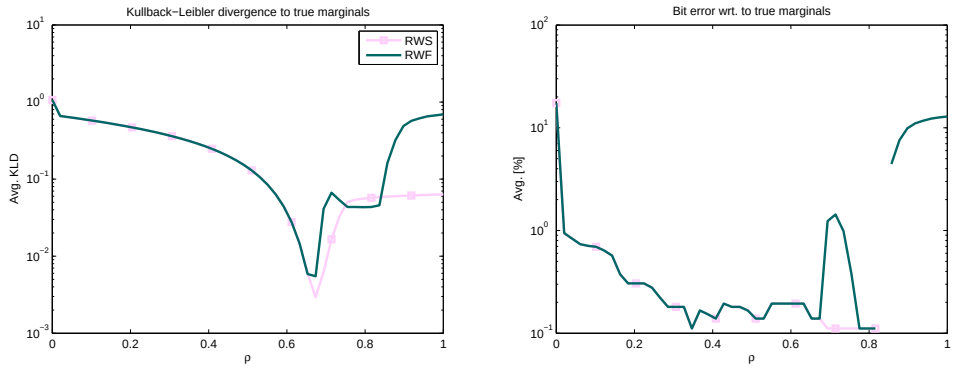
$t_{\max} = 100$ iterations were required for BP with flooding updating while only around 50 iterations were required for the serial updating to obtain convergence in the beliefs. The URW-BP was in these cases also shown to converge faster than the standard BP. The choice of K, the partitions in λ for the homotopy methods, and the balance between K and $t_{\max}$, seemed to have quite a low dependence on the results. In the convergent case of BP, as for the standard BP a total of around 100 iterations were needed for convergence of the homotopy variations of BP. Thus the only requirement for convergence in the beliefs was to choose K and $t_{\max}$ such that $K \cdot t_{\max} \gtrsim 100$. For the cases of convergence to wrong solution and periodic beliefs, no combination of K and $t_{\max}$ seemed to be able to fix this problem or improve the performance. For simplicity, all methods were simulated with 100 iterations in total.

All in all, when using modifications of the BP algorithm, the message updating method (flooding or serial) seems to have less importance than in the standard BP algorithm. This was true for all distributions in the simulations on the Ising models. For the homotopy modification, it seems that the choice of starting structure of the factor graph is influencing the result in a more distinguishable way. The curves plotted in (3.4a), (3.5a) and (3.6a) show quite different characteristics of the HS, HF curves and the TS, TF curves.

Note that even though word error rates are mentioned in the text, the plots of this particular performance measure look almost the same as those for the bit error rate, apart from the scale. Therefore, we choose to exclude these plots from the report, but nonetheless tests comparing the word error rates were performed and are thus mentioned in the text.

3.3.1 Results from simulations on Distribution 1

As can be seen below in figures (3.1a), (3.1b) and (3.2a)-(3.2d), the attractive case of θ_{nm} gives good results, since it agrees well with what the local parameters θ_n say about the variables. It can also be established that when θ_n approaches 0, which means little or no local information about the variables, this yields almost always a bad result from the algorithm. A third important observation that can be made is that as θ_{nm} gets more repulsive, this increases the disagreement with the local parameter θ_n, and hence the performance of the BP algorithm decreases. This happens since θ_n in this case is uniform, and considers all variables to be either -1 or 1, which contradicts the repulsive information in θ_{nm}. What we can see regarding the different methods, is that the original BP coupled with a flooding updating scheme is the method that performs worst. The original BP with serial updating and all the four different homotopy methods perform almost equally well. From figure (3.3a) we can see that the URW-BP seems to improve the correctness of the beliefs where the other methods fail. Note also that the transition between the good and the bad cases in figure (3.1a) and (3.1b) are quite sharp, hence we can really make a distinction between the two. Worth mentioning for this distribution is that all the homotopy methods with both updating schemes and the standard BP with the serial updating scheme give zero bit error in all trials, while the standard BP with the flooding updating gives an average bit error of 16% over all trials.



(a) *(b)*

Figure 3.3: Figures (a) and (b) show the KL divergence in $\log_{10}$ and the bit error rate in [%] respectively, for the URW-BP with respect to the parameter ρ. The figures also show results for both updating schemes with $t_{\max} = 100$. These are average plots over all 1600 trials from distribution 1.

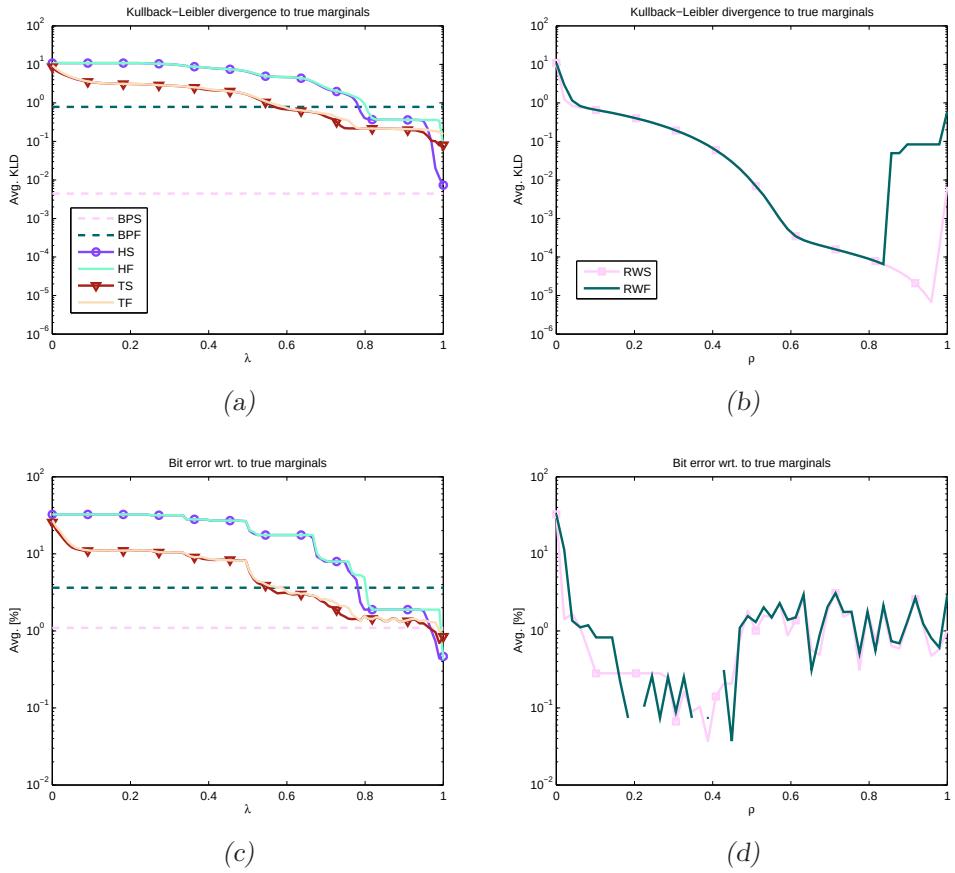


Figure 3.4: Figures (a) and (c) show the average KL divergence in $\log_{10}$ scale and the average bit error rate in [%] respectively, for all the homotopy methods with both the flooding and the serial updating scheme. The performance measures are plotted with respect to the homotopic parameter λ . Figure (b) and (d) also show the KL divergence in $\log_{10}$ and the bit error rate in [%] respectively, but for the URW-BP with respect to the parameter ρ. These figures also show results for both updating schemes. Here $t_{\max} = 100$ for both the standard BP and the URW-BP with flooding and serial updating, while $K = 100$ and $t_{\max} = 1$ for all the homotopy variations of BP. These are average plots over 1500 trials from distribution 2.

3.3.2 Results from simulations on Distribution 2

As figure (3.4a) shows, the serial updating scheme outperforms the flooding updating scheme for the original BP in terms of KL divergence again, as in the trials from distribution 1. Regarding the four different homotopy methods, none of the implemented homotopy algorithms performs better than the original BP with serial updating in terms of KL divergence. But, as we can see in figure (3.4c), the bit error rates improve with all the homotopy methods compared to the standard BP algorithms, although only marginally. When testing the URW-BP on this distribution, the results are by far the best in the sense of KL divergence. Though, in figure (3.4d) depicting the bit error rates for the reweighted methods, we notice that a lower average KLD does not always correspond to a lower average bit error rate.

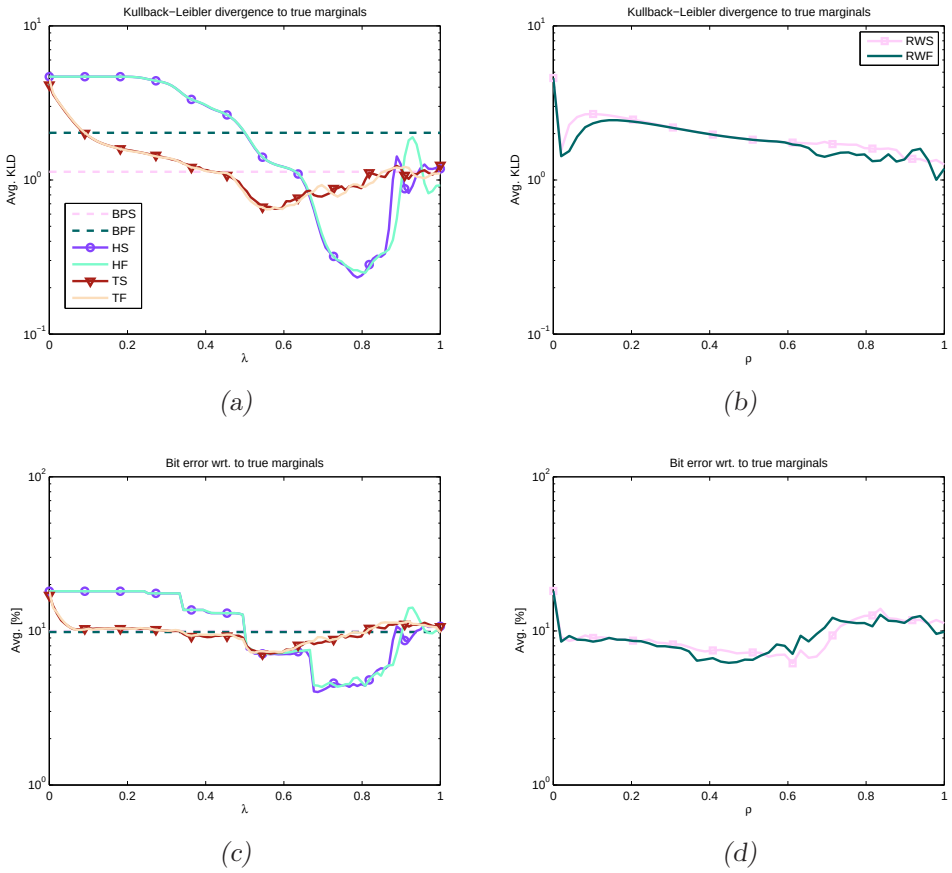


Figure 3.5: Figures (a) and (c) show the average KL divergence in $\log_{10}$ scale and the average bit error rate in [%] respectively, for all the homotopy methods with both the flooding and the serial updating scheme. The performance measures are plotted with respect to the homotopic parameter λ . Figure (b) and (d) also show the KL divergence in $\log_{10}$ and the bit error rate in [%] respectively, but for the URW-BP with respect to the parameter ρ. These figures also show results for both updating schemes. Here $t_{\max} = 100$ for both the standard BP and the URW-BP with flooding and serial updating, while $K = 100$ and $t_{\max} = 1$ for all the homotopy variations of BP. These are average plots over 1500 trials from distribution 3.

3.3.3 Results from simulations on Distribution 3

The results obtained from the simulations (figures (3.5a)-(3.5d)) suggest that the standard BP with flooding updating is worse than all the homotopy methods and BP with serial updating in KL divergence. The four homotopy methods seem to tend towards the same error as the standard BP algorithm with serial updating, but especially the HF and the HS methods have a distinguishable minimum somewhere in a neighbourhood of $\lambda = 0.75$ which outperforms the serial BP algorithm. From the results we observe that the URW-BP cannot handle these kinds of Ising models, as it provides no distinguishable better solutions in any of the performance measures. Overall, none of the methods perform particularly well for these Ising models. In addition, the methods do not only converge to the wrong solution, but in some cases they fail to converge all together, as we observed several configurations which resulted in periodic beliefs with respect to the iteration index.

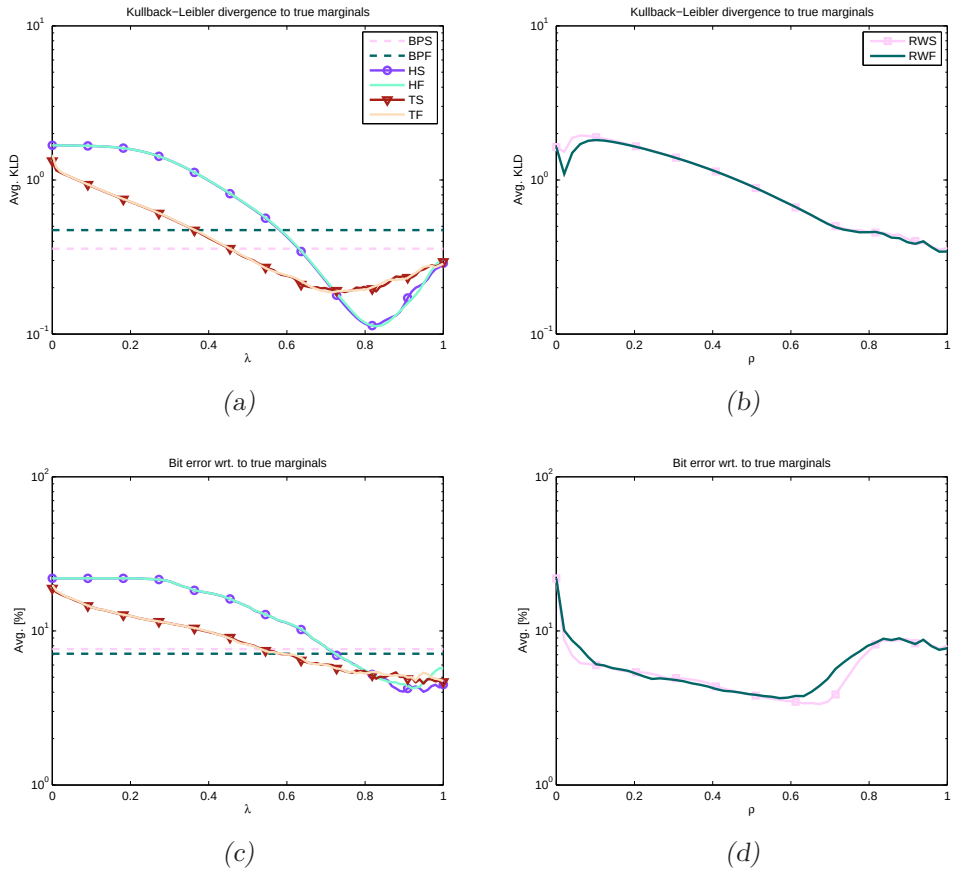


Figure 3.6: Figures (a) and (c) show the average KL divergence in $\log_{10}$ scale and the average bit error rate in [%] respectively, for all the homotopy methods with both the flooding and the serial updating scheme. The performance measures are plotted with respect to the homotopic parameter λ . Figure (b) and (d) also show the KL divergence in $\log_{10}$ and the bit error rate in [%] respectively, but for the URW-BP with respect to the parameter ρ. These figures also show results for both updating schemes. Here $t_{\max} = 100$ for both the standard BP and the URW-BP with flooding and serial updating, while $K = 100$ and $t_{\max} = 1$ for all the homotopy variations of BP. These are average plots over 1500 trials from distribution 4.

3.3.4 Results from simulations on Distribution 4

In figures (3.6a)-(3.6d), we can see the results for this more general set of Ising models. The observed solutions of the homotopy methods produce a smoother error function with respect to λ. In this set up, all the proposed homotopy variations to the standard BP algorithm perform slightly better than the original algorithm. The URW-BP performs better in the sense of bit error but provides no better solutions than the original algorithm in KL divergence. Similarly to distribution 3, the URW-BP has trouble of handling these Ising models. Worth noting is that we again observe minima for the homotopy methods in the interval $\lambda \in (0,1)$. Also similarly to the simulations on distribution 3, we again observe the periodic beliefs for several cases of the values of the model parameters.

3.4 Discussion

In many applications where the BP algorithm is used, it is important to be able to implement the algorithm in a distributed fashion. To do that, the flooding updating scheme has to be used. This seems to restrict the performance of such a system, as we have seen that the serial updating scheme performs better in most cases compared to a BP algorithm with flooding updating. The fix might be to implement either homotopy or reweighting modifications to the algorithm. They perform equally well, and sometimes even better than the BP algorithm with serial updating, and they have the advantage that they can be constructed as distributed algorithms. But, as we have seen it would be preferred to use the URW-BP method, as it seems to be the best performing method for distribution 1 and 2 among those tested in this thesis. Even though the homotopy methods showed to slightly improve the performance for distribution 3 and 4, no method suggested in this thesis seemed to improve the performance in a distinguishable way when comparing to the standard BP for these distributions.

Another issue that originates from applications concerns the choice of ρ for the URW-BP method. It is not feasible to search for the optimal ρ as was done in the simulations, since this would increase the complexity a lot and it would also require the exact solution. Instead the approximation presented in Section (2.3.3) should be used in practical problems utilising URW-BP. According to this equation the optimal value of ρ for the Ising model tested in this thesis would be $\rho_{\text{opt}} \approx \frac{|V|-1}{|E|} = \frac{9-1}{12} \approx 0.667$. From the simulations on distribution 1 and 2 we can see that this value of ρ gives a very good result in all measures, and for distribution 1 we notice from the simulations that the optimal value of ρ in KL divergence is somewhere around 0.67. This value of ρ also gives a good result in the bit error for distribution 1, so the test seems to confirm the theoretical derivation.

It should also be considered that the KL divergence does not always fit all applications' need for a good performance measure. For applications using binary models like the Ising model, it would perhaps be more interesting to look at bit error rates and word error rates. An interesting observation that we made was that a lower KL divergence does not always correspond to a lower bit error or word error rate. This could probably be explained by the decisional nature of the measure when the approximations and the true marginals lie close to a probability of 0.5 for both states. This could cause bit errors, but the probability distance would not be large between the two. On the other hand, assume that one of the approximated bits is very far off, say probability almost 1 for one state, when the true bit is almost surely the other state. Then the probability distance between the approximation and the exact solution would be quite large but result in just one bit error anyhow. Future work in investigating belief propagation and other message passing methods on the Ising model should include tests for larger networks and also other structures than a square, e.g. toroidal structures. Also, the periodicity of the solutions to some models should be explored. Perhaps some kind of damping is needed in the system to evade the instability issues in the estimation.

3.5 Conclusion

We have seen that it is possible to improve the performance of the standard BP algorithm by means of the investigated modifications. Especially interesting is the results from

distribution 1, where the URW-BP seems to be an appropriate choice of modification. It is interesting because this distribution could be directly related to the tracking model, and hence, these alternative reweighted algorithms could hopefully improve the results of BP also in the tracking problem application. The homotopy-based message-passing algorithms on the other hand did not perform as well as we hoped. The only case where they performed best was distribution 4 but the difference was almost insignificant. There is a downside however with this case, since there are cases where the solution is periodic and little could then be said about the quality of the solution. Also, the best solution was not obtained by letting λ increase to 1, but there were minima for other values less than 1, which also makes determining the quality of the solution more difficult. Future work concerning the homotopy methods should include investigation on finding the optimal λ, as in the case with the URW-BP where there is an optimal value of ρ. Success in finding some sort of theoretical derivation of this could improve the performance of the HF and HS methods for Ising models of distribution 3 and 4, to make them outperform standard BP. Perhaps the homotopy algorithms could even be modified to make the decreasing trend in the error continue until $\lambda = 1$, and not stop and turn upwards again as it does in our case, and thus make them perform even better.

Chapter 4

The tracking problem

4.1 Problem description

Suppose that there is a joint probability mass function $p(\mathbf{x}|\mathbf{z})$ with a possible factorisation according to the following equation

$$p(\mathbf{x}|\mathbf{z}) \propto \prod_{(n,m)\in E} p(x_n|x_m) \prod_{i=1}^{N} p_i(z_i|x_i)p_i(x_i) \tag{4.1}$$

where $\mathbf{x} = (x_1,x_2,\ldots,x_N)$, $\mathbf{z} = (z_1,z_2,\ldots,z_N)$ and E is the set of pairs (n,m) with a nonzero mutual factor $p(x_n|x_m)$. This factorisiation can be interprated as a factor graph with $\mathbf{x}$ representing the variable vertices, $\mathbf{z}$ representing the local parameters and the set E corresponding to edges of the factor graph.

Let now each sensor in a network be represented by a vertex m in the factor graph and let the Gaussian measurement of a target position from said sensor be represented by the corresponding z_m. Also consider the communication routes over the network as the edges E. We define the domain Ω as the area covered by the sensor network and let the position of the target be a specific point in Ω. By letting the sensors take noisy distance measurements to the target and apply message-passing between each other, beliefs for each sensor of where the target is can be computed by belief propagation. We define $p_m(z_m|x_m)$ as the unnormalised likelihood function of the measurement z_m from sensor m which yields that

$$p_m(z_m|x_m) = \exp(-\frac{1}{2\sigma^2}(\|x_m - s_m\| - z_m)^2)$$

Here s_m is the known position of sensor m and σ^2 is the variance of the noise in the sensor measurements. $p_m(x_m)$ is the prior distribution of x_m, i.e. it is the prior knowledge of where the target is located according to sensor m. We also let $p(x_n|x_m)$ be an unnormalised Gaussian distribution of the variable x_n given x_m such that

$$p(x_n|x_m) = \exp\left(-\alpha\|x_n - x_m\|^2\right)$$

This corresponds to a Gaussian distribution $\mathcal{N}(x_n;\Sigma,\mu)$ with covariance matrix $\Sigma = \begin{bmatrix} 1/2\alpha & 0 \\ 0 & 1/2\alpha \end{bmatrix}$ and mean $\mu = x_m$, up to some normalisation constant.

Now let us introduce a grid on Ω. We then define $x_n = (x_{n,1}, x_{n,2})$ to take values on the grid points so that the belief $b_n(x_n = (y_1, y_2))$ is the approximated probability according to sensor n that the target is located on the grid point (y_1, y_2). This means that for R number of grid points the variables can assume R number of coordinate pairs. The message updating and the construction of the beliefs can then according to (2.6) and (2.7) be done as follows

$$\mu_{m\to n}^{(t+1)}(x_n) \propto \sum_{x_m} \left(\exp\left(-\alpha \|x_n - x_m\|^2\right) p_m(z_m|x_m) p_m(x_m) \prod_{l \in \mathcal{N}_m \setminus \{n\}} \mu_{l\to m}^{(t)}(x_m) \right) \tag{4.2}$$

$$b_n(x_n) \propto \prod_{l \in \mathcal{N}_n} \mu_{l\to n}^{(t_{\max})}(x_n) p_n(z_n|x_n) p_n(x_n) \tag{4.3}$$

In this model α can be interpreted as a constant that weights the interacting information. The message $\mu_{m\to n}(x_n)$ can be regarded as a probability distribution over x_n and the value of x_n that maximises this distribution is the most likely position of the target according to the information sent from node m to n. As the term $\exp(-\alpha\|x_n - x_m\|^2)$ is maximised when the variables x_n and x_m are the same, the variables are forced to be the same if α is positive, while a negative value will force them to be different and maximise the euclidean distance between them. A high positive value of α would force the nodes to agree more than a low value. If it is of high priority to have a strong agreement among the nodes α should be chosen to be equal to a large value. Similarly, the value of the noise or variance σ^2 weights the local information. As the term $\exp(-\frac{1}{2\sigma^2}(\|x_m - s_m\| - z_m)^2)$ is maximised when x_m takes values of the coordinates that lies on a circle with centre at position s_m and radius z_m, a positive value of σ^2 wants to push the variable x_m close to the measured radius around the sensor while a negative value will push the variable x_m away from the measured radius. A low positive value of σ^2 would force the variables to be close to the measured radius more than a high value. An interpretation of this is that a low positive value of σ^2 would make the algorithm depend more on the local information than a high value which will dampen this affect. Comparing with reality this says that the information from the measurement should be taken into account dependent on the noise and hence the reliability of the measurement.

A negative variance would not make a normalisation possible for the function $p_m(z_m|x_m)$ such that it sums up to one over all x_m if $x_m \in \mathbb{R}^2$ since $p_m(z_m|x_m)$ would not be bounded. $p_m(z_m|x_m)$ could then not be constructed to satisfy the definition of a probability distribution function. However, when a grid is considered, the finite number of grid points makes this possible and the implementation could theoretically be made. The same argument follows for $p(x_n|x_m)$ when considering a negative α. Negative values of α and σ^2 would only be of interest in a pedagogical point of view, but not in the sense of the tracking problem since the variance is defined to be positive and the idea is to force the nodes to agree and push the variables close to the measurements in an optimal balanced way.

Notable is that α can be related directly to the constant θ_{nm} in the Ising model while the term $1/\sigma^2$ relates to the constant θ_n. The relation between α and σ controls the proportions between the weighting of the local and interacting information in the tracking

model comparable to the relation between θ_n and θ_{nm} in the Ising model. Consider also the reweighted algorithm for the tracking problem

$$\mu_{m\to n}^{(t+1)}(x_n) \propto \sum_{x_m} \left(\exp\left(-\frac{\alpha}{\rho}\|x_n - x_m\|^2\right) p_m(z_m|x_m) \prod_{l\in\mathcal{N}_m\setminus\{n\}} \frac{\left(\mu_{l\to m}^{(t)}(x_m)\right)^{\rho}}{\left(\mu_{n\to m}^{(t)}(x_m)\right)^{1-\rho}} \right) \tag{4.4}$$

$$b_n(x_n) \propto \prod_{l\in\mathcal{N}_n} \left(\mu_{l\to n}^{(t_{\max})}(x_n)\right)^{\rho} p_n(z_n|x_n) \tag{4.5}$$

The tracking model, similarly to the Ising models, gives opportunities to test different approaches on the implementation of the belief propagation algorithm. Also different updating schemes and graph structures can be tested. Parameters to explore is ρ,α, σ and $t_{\max}$. As this method is implemented only in a theoretical environment there is an optimal solution to compare with. By multiplication of all the measurement likelihoods and all the priors from the nodes the optimal solution can be obtained. More mathematically this can be shown by first using Bayes' rule

$$p(x|z_1,z_2,\ldots,z_N) = \frac{p(z_1,z_2,\ldots,z_N|x)p(x)}{p(z_1,z_2,\ldots,z_N)} \propto p(z_1,z_2,\ldots,z_N|x)p(x) \tag{4.6}$$

where $p(x|z_1,z_2,\ldots,z_N)$ is the optimal belief which is the probability of the target position x given all the measurements z_i for $i = 1,\ldots,N$. Similarly $p(z_1,z_2,\ldots,z_N|x)$ is the likelihood of all measurements given the target position x and $p(x)$ is the prior distribution of x. Then by using the fact that the measurement likelihoods are independent, this can be rewritten as

$$p(x|z_1,z_2,\ldots,z_N) \propto p(x)\prod_{i=1}^{N} p(z_i|x) \tag{4.7}$$

Since the total prior $p(x)$ is the product of all priors $p_i(x)$ for the nodes $i = 1,\ldots,N$, i.e. $p(x) = \prod_{i=1}^{N} p_i(x)$, this expression can be further rewritten to finally obtain the desired result

$$p(x|z_1,z_2,\ldots,z_N) \propto \prod_{i=1}^{N} p(z_i|x)p_i(x) \tag{4.8}$$

In practice when considering large networks with constrained communication and uncertain position knowledge among nodes it would not be possible to estimate the position in this way since "someone" has to know all the positions and the measurements of the nodes to perform the operation in equation (4.8). However, since the tests in this thesis are done only theoretically all the node positions and measurements are known. Thus the optimal belief $p(x|z_1,z_2,\ldots,z_N)$ serves as the exact solution when evaluating the message passing methods.

4.2 Prediction

If no prior information is available then $p_m(x_m)$ will be a uniform distribution over all grid points represented by a constant that will fall out of the equations. Another possible

way of using the prior tested in this thesis is to assume a known model of the target's movement. Let us introduce the notation x_n^τ for the variable x_n at time stamp τ. Then the belief from the previous time stamp $\tau - 1$ can be used to predict the prior for the current time stamp τ according to the Markov property

$$p_n(x_n^\tau) = \sum_{x_n{}^{\tau-1}} p(x_n^\tau | x_n^{\tau-1}) p_n(x_n^{\tau-1})$$

Then as $p_n(x_n^{\tau-1})$ we will use $b_n(x_n^{\tau-1})$ and $p(x_n^\tau | x_n^{\tau-1})$ will be the assumed model of the movement. Thus we will obtain one prior $p_n(x_n)$ for each node n. This means that we will use the belief from each node to predict its individual prior in the next time stamp.

Let us introduce the index notation (i,j) for a pair of coordinates on the grid on Ω and assume R number of grid points. A simple random walk method is to only let the target move one grid point in vertical or horizontal direction with equal probability. Then the probability that the target moves from the coordinate (i,j) to either $(i,j \pm 1)$ or $(i \pm 1,j)$, if (i,j) is not on the boundary of Ω, is $P(X_{t+1} = (i,j \pm 1)) = 1/4$ or $P(X_{t+1} = (i \pm 1,j)) = 1/4$ respectively. Then it must also follow that $P(X_{t+1} \notin N_S) = 0$, when $N_S = \{(i,j \pm 1), (j,i \pm 1)\}$. With the same argument we can derive the probabilities for the case when (i,j) is on the boundary of Ω. A prior prediction distribution $p(x^\tau | x^{\tau-1})$ satisfying these properties is tested and evaluated in this thesis. $p(x^\tau | x^{\tau-1})$ is used to generate target positions in a series of movements and the priors $p_n(x_n^\tau)$ are used in the implementation to predict the movement.

4.3 Method

Investigations on the flooding and the serial updating schemes were made and the standard BP and the URW-PB were tested on a tree graph and and a loopy graph. The relation between the parameters σ and α was analysed and related to the parameters θ_{nm} and θ_n in the Ising model. The performance of the methods was evaluated for both updating schemes and both graph structures as a function of $t_{\max}$, α and ρ, to test convergence rates and model parameter dependences. To get an even more expositive analysis of the model, the methods were tested and evaluated both with a known movement model of the target and with no prior information of the location. By performing these tests, the hope was to find the set of parameters and method that optimised the performance and analyse the convergence of the standard BP and the URW-BP. Similarly to the simulations on the Ising model, the simulations for the tracking model were done by generating 10 measurements of the same target location and the performance measures were averaged over these trials. To assess the methods, three measures of performance were determined.

4.4 Measure of performance

- *Distance to optimal performance*
  By this we mean, for one node, the probability distance from its belief $b(x)$ to the optimal belief $p_{\text{opt}}(x)$. This measure was computed by the KL divergence as $D_{\text{KL}}(b||p_{\text{opt}})$. In the results we present the average over all nodes, the KL divergence from each belief to the optimal belief. The optimal belief was computed according to equation (4.8).

- *Disagreement of the nodes*
  This was evaluated by the average of the KL divergence between all nodes pairwise. The KL divergence between two nodes n and m was defined as $D_{\mathrm{KL}}(b_n||b_m) + D_{\mathrm{KL}}(b_m||b_n)$

- *Accuracy of the nodes*
  This was computed, for one node, by the euclidean distance between the expected value of the coordinate x according to its beliefs $b(x)$ and the expected value of x according to the optimal belief $p_{\mathrm{opt}}(x)$, i.e $\|\mathbb{E}_{p_{\mathrm{opt}}}[X] - \mathbb{E}_{b_n}[X]\|$, where $\mathbb{E}_{p_{\mathrm{opt}}}[x] = \sum_x p_{\mathrm{opt}}(x)x$ and $\mathbb{E}_{b_n}[x] = \sum_x b_n(x)x$. In the results we present the average over all nodes, the euclidean distance from each belief to the optimal belief. Also here, the optimal belief was computed according to equation (4.8)

4.5 Implementation

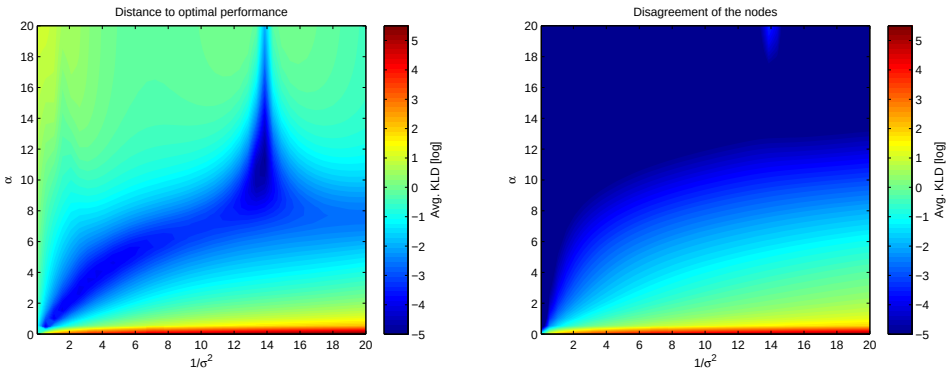
In the simulations, a square area of 10 by 10 meter is used as domain Ω, and as sensor network, one sensor is placed in each corner of the square. This yields a network of 4 sensors that can communicate through message passing. Ω is discretised into 400 equally spaced grid points which the variables can assume. The measurements z_m is generated as $z_m = \|s_m - p_{\mathrm{exact}}\| + \mathcal{N}(0,\sigma^2)$, where p_{exact} is the exact position of the target assumed unknown to the model. p_{exact} is placed on one of the grid points.

Consider the network represented by a graph $G = (\mathcal{V},E)$, where $\mathcal{V}$ is the set of nodes and E is the set of edges. The edges E correspond to the possible communication links pairwise between the nodes $\mathcal{V}$. As mentioned above the set $\mathcal{V}$ contains 4 nodes. Let E be a set of edges that form a loop around Ω. Then the methods are tested for the graphs $G_1 = (\mathcal{V},E)$ and $G_2 = (\mathcal{V},T)$, where T is a subset of E such that one edge is removed and hence G_2 corresponds to a tree. We will refer to G_1 as the loopy graph and G_2 as the tree graph.

The implementation of the tracking model was done in MATLAB and all the plots depicting the results were produced by the plot functions `semilogy`() and `contourf`(). When generating target measurements with normal distributed noise, MATLAB's random number generator `randn`() combined with a seed function was used to generate the same normal distributed measurement noise for each set of model parameters. The computations were done in log-domain to make the implementation possible due to numerical issues. Vectors of the the model parameters α, ρ, σ and $t_{\max}$ were created to search for optimal performance and make it possible to evaluate and analyse the methods. The messages, the beliefs and the functions in the message updating and in the belief construction equations were implemented as matrices, where each element represented a position in the grid and hence a variable state. When multiplying these, pointwise matrix multiplication was used.

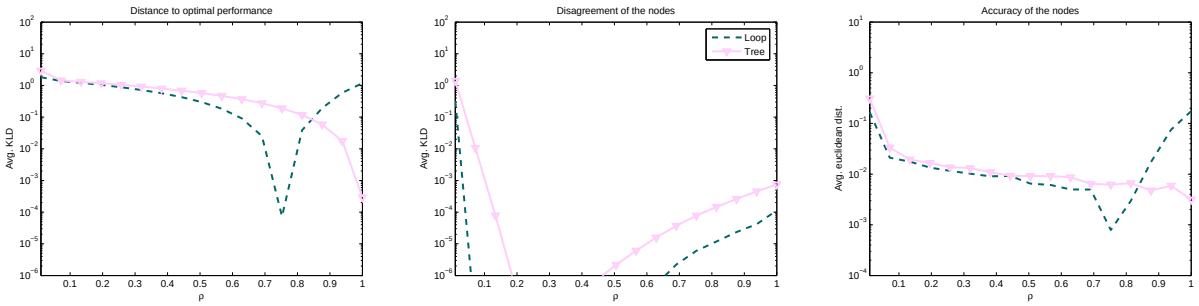
4.6 Results

From the simulations done on the tracking model, we conclude that both the standard BP and the URW-BP modification give the exact same results for the flooding updating



(a) *(b)*

Figure 4.1: These figures show the distance to the optimal performance and the disagreement of the nodes as a function of α and $1/\sigma^2$. Here, $t_{\max} = 60$. These are average plots of 10 target measurements.



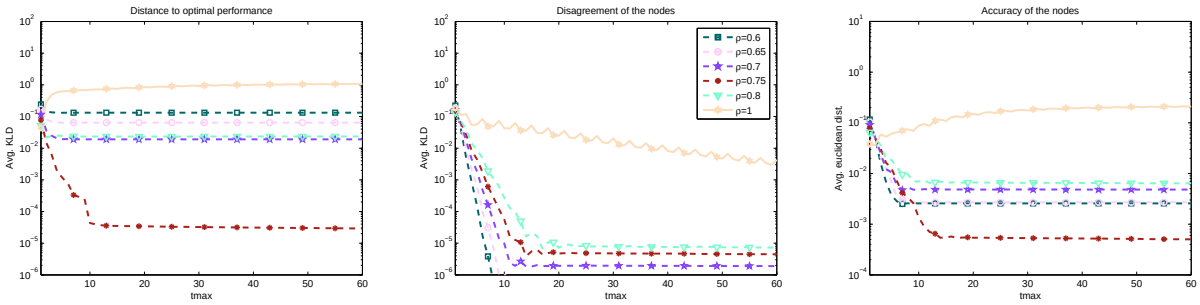
(a) *(b)* *(c)*

Figure 4.2: The above plots are all three performance measures as a function of ρ for the URW-BP algorithm. The standard BP, which is equal to to URW-BP for $\rho = 1$, can also be seen from these figures. $t_{\max} = 60$, $\alpha = 20$ and $\sigma = 0.5$. These are average plots of 10 target measurements.

scheme as well as the serial updating scheme. Therefore, no distinction will be made between the two, and there is no reason to present results from both nor to know which updating scheme the plots were generated from.

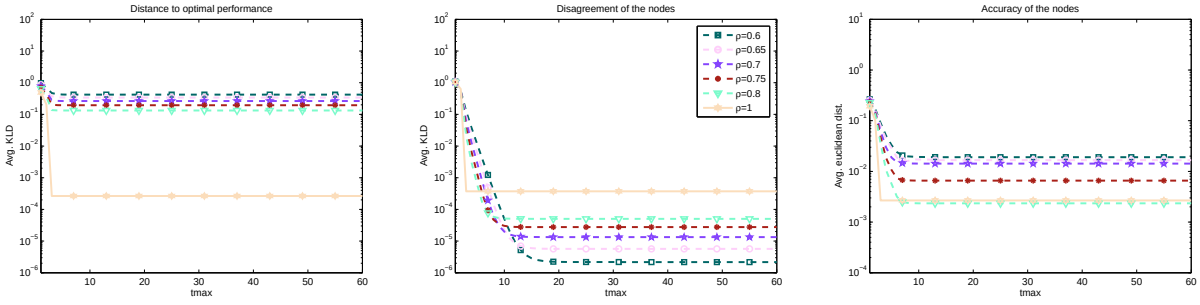
In figure (4.1a) and figures (3.1a)-(3.1b) we can clearly see how the parameter α and the term $1/\sigma^2$ in the tracking model relate to θ_{nm} and θ_n in the Ising model, due to the similarities in the figures. Figure (4.1a) would correspond to the upper right quadrant where $\theta_{nm} \geq 0$ and $\theta_n > 0$ in figures (3.1a) and (3.1b). The disagreement of the nodes can also be seen in figure (4.1b). Since it is desirable to make the nodes agree well and we would like a guarantee of convergence, the results presented here are done with $\alpha = 20$ and $t_{\max} = 60$ when these parameters are fixed. These settings have shown to meet this demand in the simulations.

Figures (4.2a)-(4.2c) show that the URW-BP on a loopy graph provides approximately



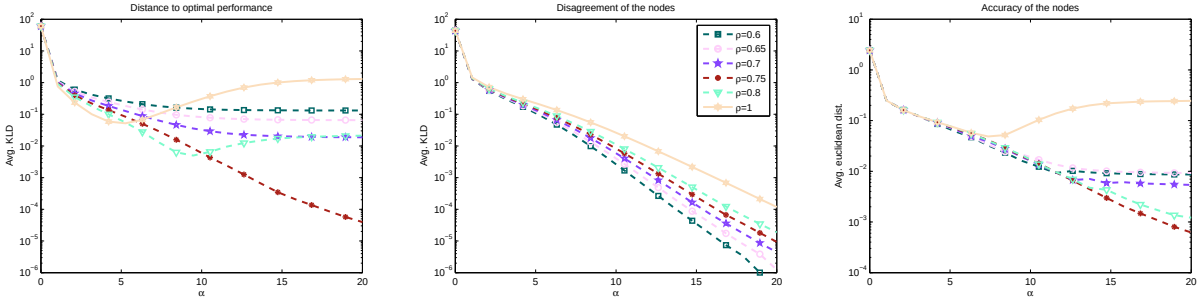
(a) *(b)* *(c)*

Figure 4.3: The above plots are all three performance measures as a function of $t_{\max}$ for the URW-BP algorithm and the standard BP on a loopy graph. $\alpha = 20$ and $\sigma = 0.5$. These are average plots of 10 target measurements.



(a) *(b)* *(c)*

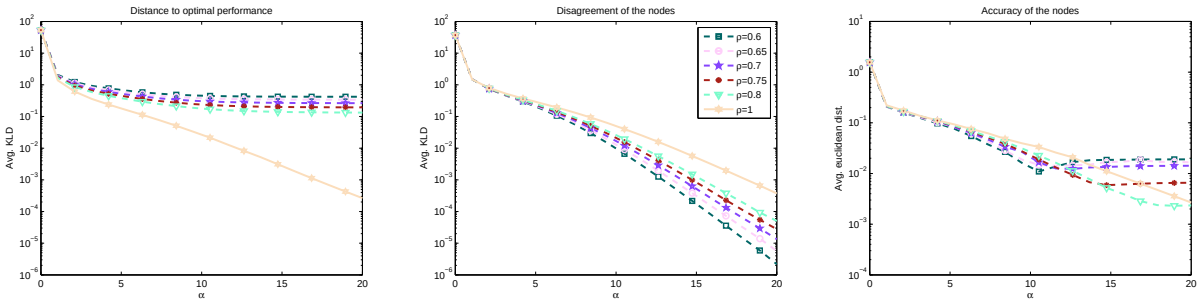
Figure 4.4: The above plots are all three performance measures as a function of $t_{\max}$ for the URW-BP algorithm and the standard BP on a tree graph. $\alpha = 20$ and $\sigma = 0.5$. These are average plots of 10 target measurements.



(a) *(b)* *(c)*

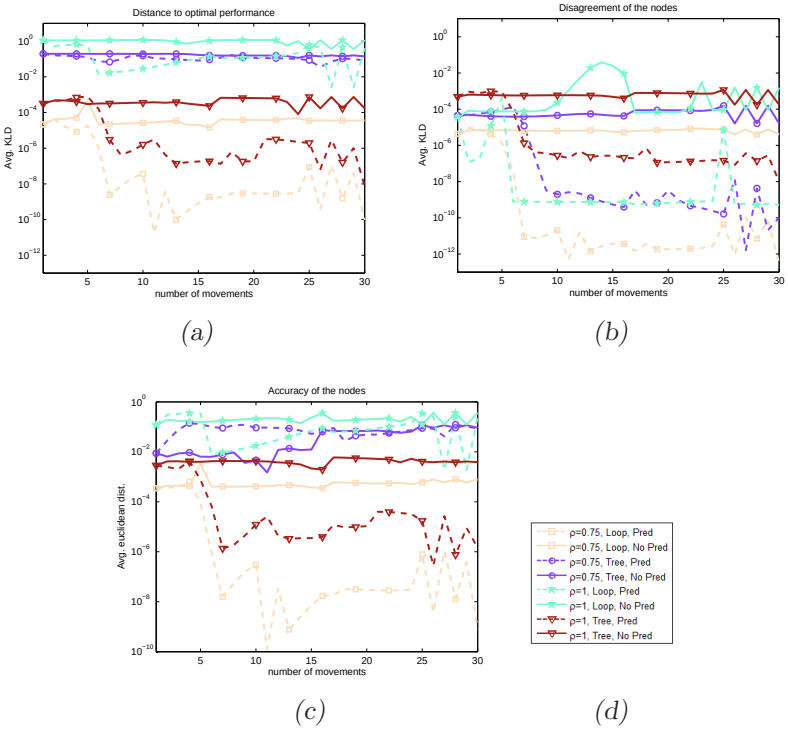
Figure 4.5: The above plots are all three performance measures as a function of α for the URW-BP algorithm and the standard BP on a loopy graph. $t_{\max} = 60$ and $\sigma = 0.5$. These are average plots of 10 target measurements.

the same minimum distance to the optimal performance as the standard BP on a tree graph. Important for the reader to know here is that the standard BP is the same as



(a) *(b)* *(c)*

Figure 4.6: The above plots are all three performance measures as a function of α for the URW-BP algorithm and the standard BP on a tree graph. $t_{\max} = 60$ and $\sigma = 0.5$. These are average plots of 10 target measurements.



(a) *(b)* *(c)* *(d)*

Figure 4.7: These figures shows all the performance measures for the standard and URW-BP algorithm, with and without prediction, for both tree and loopy graph. $t_{\max} = 60$, $\alpha = 20$ and $\sigma = 0.5$.

the URW-BP for $\rho = 1$. However, this minimum is sharp around $\rho = 0.75$ and hence this performance measure is sensitive with respect to the parameter ρ. This indicates the importance of a correctly chosen ρ in the sense of minimising the distance to the optimal performance. In the accuracy of the nodes there is an interval of $0.7 \lesssim \rho \lesssim 0.85$ that gives better performance for the URW-BP on a loopy graph than the standard BP on a tree graph. This minimum, also centered around $\rho = 0.75$, is less sharp and the curve is more flat which means that the accuracy of the nodes is less sensitive with respect

to the parameter ρ when comparing to the distance to optimal performance. Though in the sense of the disagreement when comparing the URW-BP on a loopy graph and the standard BP on a tree graph, the only requirement to obtain a value better in the reweighted case would be to chose $\rho \gtrsim 0.05$. The value that optimises the performance in the distance to the optimal performance and the accuracy of the nodes, when the URW-BP on a loopy graph is considered, is $\rho = 0.75$. This value does not give the best possible result in the disagreement, nonetheless the results are still satisfying when comparing to the disagreement of the standard BP on a tree graph. Notable is that the URW-BP performs worse than the standard BP for almost all ρ with respect to the distance to the optimal performance and the accuracy of the nodes when a tree graph is considered for both, and better for almost all ρ when a loopy graph is considered for both. This makes sense as the URW-BP is constructed to improve performance on loopy graphs [17]. The disagreement, though, is better in the URW-BP for almost all ρ for both graph types.

Figures (4.3a)-(4.3c) and (4.4a)-(4.4c) support the statement that the URW-BP converges to equally or slightly better values on a loopy graph than the standard BP on a tree graph for $\rho = 0.75$. From these figures we can do a convergence study and conclude that the standard BP provides an extremely fast convergence rate in all measures when the tree graph is considered. For the loopy graph the URW-BP with $\rho = 0.75$ has a slightly slower convergence rate in the distance to optimal performance and in the accuracy of the nodes compared to the other reweighted methods which have approximately the same convergence rate. In the disagreement we notice higher convergence rate for higher values of ρ when the loopy graph is considered and the standard BP is obviously the slowest.

In figures (4.5a)-(4.5c) and (4.6a)-(4.6c) we can study the dependence of the performance with respect to α and notice similar results. The figures show how the URW-BP methods on a loopy graph improve the distance to the optimal performance and the accuracy of the nodes for higher values of α, and that the URW-BP always gives best result in the disagreement measure. Figures (4.7a)-(4.7c) also show that the URW-BP on a loopy graph and the standard BP on a tree graph give the best results when the prediction model outlined in the prediction section (4.2) is used.

4.7 Discussion

When standard BP is applied to both the tracking model and the Ising model, it performs best when applied to a tree graph. On a loopy graph the URW-BP is shown to improve the performance in most cases tested for the Ising model and in all cases tested for the tracking model. Ising models from distribution 1, which is the closest related Ising model to the tracking model, show the same improvements under the effect of reweighting as the tracking model. These similarities can also be seen in Ising models from distribution 2. The results also confirm a relation between the both model parameters that supports the theory outlined in the problem description section (4.1).

In the tracking model, a desirable property of the URW-BP on a loopy graph is that it allows for a high value of α without lack of performance in the distance to optimal performance and the accuracy of the nodes, in contrast to the standard BP. This enables strong agreement among nodes and in this thesis the URW-BP is shown to always give

the best agreement on both graph types. The simulations also show that the URW-BP on a loopy graph performs at least as good as the standard BP on a tree graph, when the small network tested in this thesis is considered and if ρ is set correctly. The theoretical value of ρ that minimises the error, $\rho_{\text{opt}} \approx \frac{|V|-1}{|E|} = \frac{4-1}{4} = 0.75$, as described in section (2.3.3) agrees very well with the observed optimal value which is also $\rho \approx 0.75$.

An important property of the tracking model is that the flooding and the serial updating schemes gave the same results, which means the tracking model in contrast to the Ising model needs no investigations in certain updating schemes. Since the message passing methods in practice often are implemented in a distributed computational manner, this is a valued result.

If this application of message passing were to be implemented in practice on a large network with access to a known tree graph and a known loopy graph that both connects all the nodes, the most convenient way would be to choose the tree graph and use the standard BP since it has an extremely fast convergence rate, the benefit that no estimations of the parameter ρ have to be made and the fact that both methods are shown to perform equally well (when ρ is set correctly). However, if no known tree graph exists, the task of finding a tree that connects all the nodes can be computationally complex and demanding for a large network and thus this becomes a problem. If the same results as for the tree graph can be obtained without any restrictions on the graph the communication can just be done by letting the nodes communicate unconstrained. This would facilitate the implementation substantially and it can be achieved by the URW-BP. The only problem would then be how to choose ρ properly. On the small network tested in this thesis we saw a quite sharp minimum in the distance to optimal performance with respect to ρ. The same behavior was also noticed for the accuracy of the nodes but to a smaller extent, while the dependency of ρ in the disagreement was almost negligible. The distance to optimal performance can be of importance in many cases, so if an accurate result in this measure is required, the biggest problem would probably be to determine ρ in the implementation. However, in the network in this thesis, the theoretical derivation and the simulations gave the same result so finding the optimal ρ in this case is not a problem. One should though not completely rely on this because the method needs to be tested on a much bigger network to really establish the sensitivity and the dependence with respect to ρ. Perhaps one would notice a more flat curve in all measures such that ρ belonging to a large continuous interval would satisfy the required results. Another problem in a practical implementation can be how to actually estimate ρ. If the number of edges (possible paired communication links) or the number of vertices is not known by any unit in the network the derivation done according to the equation mentioned earlier in this section could not be done exactly.

When considering all message passing algorithms and the implementations tested in this thesis, the biggest problem is that the complexity of the grid scales as R^2 for R number of grid points (see statement in section (2.3)). Hence this approach would be very impractical when accurate estimates on large areas are demanded. This is easily fixed by a parametric representation of the messages. Assuming that the messages can be represented by Gaussian distributions characterised by a mean vector of size 2×1 and a covariance matrix of size 2×2, the implementation would be easier and the algorithm would be a lot faster. Another benefit of this is that instead of sending messages with

R real numbers between the sensors the parametric representation would only require 6 real numbers (2 for the mean vector and 4 for covariance matrix). Important future work will be to implement the parametric representation of the messages and beliefs and test the algorithms on larger networks.

4.8 Conclusion

Since the URW-BP seemed to be the most promising modification to the standard BP after simulations on the Ising model, we chose to use this method in the tracking problem. The method was shown to improve the performance on a loopy factor graph of 4 nodes and provide slightly better result than the standard BP applied to a tree graph. Since it in most practical cases is favorable to use a loopy graph, the URW-BP may posses great potential in real world applications. The standard BP and the URW-BP were also shown to give the same results with both updating schemes in the simulations. To establish the URW-BP in an industrial application like the tracking problem, further research needs to be done. This includes testing the algorithm on larger networks and develop a method such that the implementation can be done without a grid to make it feasible for use in performance restricted systems. Similarities between the Ising model and the tracking model was discovered in the sense that some results where directly transferable from one model to another. This indicates that the Ising model is a great tool to easy test new variations of BP before applying them to problems of a more complex nature.

Chapter 5

Conclusions

After testing and evaluating the variations of BP on the square Ising model, we conclude that the homotopy methods do not over all give better results than the standard BP. However, there are classified cases where the URW-BP provides better solutions. In these classes of Ising models, the optimal weighting constant from the simulations agrees well with the derivation outlined in [15]. Since these classes are the ones most related to the tracking problem described in this thesis, the URW-BP was chosen as a variation of BP in this application. The URW-BP was, in the simulations on distributions 1 and 2 of the Ising models, shown to perform better than the standard BP on a loopy graph. In the tracking problem, the reweighed method successively allowed for stronger agreement among the nodes without lack of performance in the estimation of the target position, in contrast to the standard BP. It was also shown in the tracking problem, that when using the derivation of the optimal weighting constant proposed in [15], the URW-BP on a loopy graph performed slightly better than the standard BP on a tree graph. These results were established on a square factor graph of 4 nodes. The serial and the flooding message updating were tested for both the Ising model and the tracking model. In the Ising model the serial updating was shown to give a more accurate result and have a faster convergence rate than the flooding updating. The variations of BP though seemed to perform similarly with both updating schemes and the homotopy variations gave in total better performance than the standard BP with the flooding updating scheme on the Ising model. Though, in the tracking problem both updating schemes were shown to give the same result in all tests.

Bibliography

[1] F. R. Kschischang, B. J. Frey and H. A. Loeliger, "Factor graphs and the sum-product algorithm," in *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498-519, Feb. 2001.

[2] R. G. Gallager, *Low-Density Parity Check Codes.* Cambridge, MA: MIT Press, 1963.

[3] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," in *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 219-230, Jan. 1998.

[4] F. R. Kschischang, "Codes defined on graphs," in *IEEE Signal Processing Mag.*, vol. 41, pp. 118-125, Aug. 2003.

[5] H. A. Loeliger, "An introduction to factor graphs," in *IEEE Signal Processing Mag.*, vol. 21, pp. 28-41, Jan. 2004.

[6] A. S. Willsky, "Multiresolution Markov models for signal and image processing," in *Proceedings of the IEEE*, vol. 90, no. 8, pp. 1396-1458, Aug. 2002

[7] J. Pearl, *Probabilistic Reasoning in Intelligent Systems.* San Mateo, CA: Morgan Kaufmann, 1988.

[8] J. S. Yedidia, W. T. Freeman and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," in *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2282-2312, Jul. 2005.

[9] S. C. Tatikonda and M. I. Jordan, "Loopy belief propagation and Gibbs measures," in *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence*, pp. 493-500, 2002.

[10] T. G. Roosta, M. J. Wainwright and S. S. Sastry, "Convergence Analysis of Reweighted Sum-Product Algorithms," in *IEEE Transactions on Signal Processing*, vol. 56, no. 9, pp. 4293-4305, Sep. 2008.

[11] R. Kindermann and J. L. Snell, "The Ising model," in *Markov Random Fields and their Applications.* Providence, RI: American Mathematical Society, 1980, pp. 1-23.

[12] J. S. Yedidia, W. T. Freeman and Y. Weiss, "Understanding belief propagation and its generalizations," Mitsubishi Electric Research Laboratories, Tech. Rep. TR2001-22, Jan. 2002.

[13] N. Noorshams and M. J. Wainwright, "Stochastic belief propagation: low-complexity message-passing with guarantees," in *49th Annual Allerton Conference on Communication, Control and Computing*, pp.269-276, Sep. 2011.

[14] A. Katok and A. Sossinsky. (2006, Oct. 16) "Elementary Homotopy Theory," in *Introduction to Modern Topology and Geometry.*, pp. 41-74. [Online]. Available: `http://www.math.psu.edu/katok_a/TOPOLOGY/`

[15] M. J. Wainwright, T. S. Jaakkola and A. S. Willsky, "A New Class of Upper Bounds on the Log Partition Function," in *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 2313-2335, Jul. 2005.

[16] F. Penna, H. Wymeersch and V. Savić, "Uniformily reweighted belief propagation for distributed Bayesian hypothesis testing," in *Proceedings of the IEEE International Workshop on Statistical Signal Processing*, pp. 733-736, 2011.

[17] H. Wymeersch, F. Penna and V. Savić, "Uniformily reweighted belief propagation for estimation and detection in wireless networks," in *IEEE Transactions on Wireless Communications*, vol. 11, no. 4, pp. 1587-1595, Apr. 2011.