Gatsby Computational Neuroscience Unit
University College London
http://www.gatsby.ucl.ac.uk

17 Queen Square, London
WC1N 3AR, United Kingdom
+44 20 7679 1176

Funded in part by the Gatsby Charitable Foundation.

July 11, 2001

GCNU TR 2001–001

Passing And Bouncing Messages For Generalized Inference

Yee Whye Teh **Max Welling**

Gatsby Unit

1 Introduction

Inference on general loopy graphs is a NP hard problem. Many approximate methods, like Monte carlo sampling and variational approximations have become available over the last decades, each with its own advantages and disadvantages. However, when the graphical structure is a tree, there is an algorithm for doing inference that is only linear in the number of nodes in the graph. The algorithm, called belief propagation (BP), was independently discovered by [Pearl, 1988] and [Lauritzen and Spiegelhalter, 1988]. In this algorithm, messages are sent over the edges of the network in both directions, either in parallel or sequentially. It can be proven that if each message is sent only when all incoming messages which it requires have been received, then BP converges when all messages have been updated exactly once.

Recently, people have found that on some graphs, BP can also perform accurate (approximate) inference on graphs with cycles [Murphy et al., 1999]. Especially in the field of error-correcting codes, it was proven that the very successful decoding algorithms for Turbo codes and low density parity check codes can be viewed as versions of the loopy BP algorithm [McEliece et al., 1998], [MacKay and Neal, 1995].

The fact that loopy BP works at all is remarkable, since we know that evidence will be moving around in the cycles and will be overcounted as a result. However, especially when the cycles have a large circumference, and the interactions are not too strong, we may expect that the evidence from a particular observed node has “died out” when it feeds back into itself. An analysis in terms of unwrapped networks was performed by [Weiss, 2000] and revealed that in networks with a single loop, the posterior assignment will always be correct, even though the marginal probabilities are usually overly confident. However, the real breakthrough in our understanding of the nature of the approximation made when computing posterior marginals using BP on loopy graphs came with the identification of the Bethe free energy as the appropriate cost function which is being minimized[1]. This very fruitful marriage between machine learning and physics led to a class of generalized belief propagation algorithms which act as fixed point equations to minimize Kikuchi free energies, in which larger clusters are treated exactly and made consistent through propagation.

A seemingly unrelated problem is that of finding a probability distribution with prescribed marginals that is close to a prior distribution in the sense of the KL-divergence. It is well known that this problem can be solved with the iterative proportional fitting (IPF) procedure [Deming and Stephan, 1940]. The algorithm computes the conditional probability table of all the nodes, given the node whose marginal we are going to update and multiplies that with the desired marginal. Although, this guarantees that that particular node now has the correct marginal, it also changes all the other marginals, possibly in the wrong direction. However, if one updates the marginals one by one, the algorithm is guaranteed to decrease the KL-divergence at every step and converge to the unique solution. Generalizations of this procedure (Generalized IPF), which can deal with more general constraints than marginalization constraints, and which can perform parallel updates have been analysed [Darroch and Ratcliff, 1972] (see also [Pietra et al., 1997] for the related improved iterative scaling algorithm). The most important drawback of IPF is the fact that one deals with full joint probability tables, which become computationally infeasible, even for relatively small networks. In that respect, great advance was made by [Jiroušek and Přeučil, 1995], who have studied a “space saving implementation” of IPF, which basically uses Junction trees to represent the joint table. An IPF update is performed on the clique which contains the constrained marginal only, after which this information is propagated through the rest of the junction tree, before a new update is considered.

In this paper we will propose an algorithm which combines BP and IPF into one message

[1]More precisely, the fixed points of loopy BP are the stationary points of Bethe free energy. However, in practise it turns out that it converges to minima

passing algorithm on a tree. The new set of messages reduces to BP messages on the internal nodes of the tree. However, when a messages reaches a constrained marginal (other than a "hard evidence" node), it will be "bounced back", while being changed in the process. When the constrained marginal is a delta function (hard observation), the returned message is independent of the incoming message in the usual way. The algorithm requires a scheduling of messages such that the information from a bounced message has reached the node where the next bounce takes place. Unlike BP on a tree (but like IPF on a tree), the combined algorithm does not converge within a finite number of iterations.

We will argue that it is natural to view the problem being solved by this IPF-BP algorithm as one of generalized inference, where certain marginals can be constrained to marginal tables with values other than zeros and ones. Generalized inference is defined as finding the minimum divergence distribution relative to a prior distribution, when certain constraints are imposed on the marginals. When these constraints come in the form of hard evidence, this objective reduces to finding the posterior distribution. Since BP itself can be understood as minimizing divergences between marginals (and pairwise marginals) with their prior counterparts under the constraints that they have to be consistent and normalized, it may come as no surprise that one set of message updates can solve the combined problem.

Generalized inference on loopy graphs is intractible, even for a modest amount of nodes on the graph. However, also for this case we can define an extended Bethe free energy as an objective function. To minimize this objective we can run loopy BP internally until (approximate) convergence after which we perform an IPF update (a bounce) at a softly constrained node. This procedure is however not guaranteed to converge, since loopy BP is not. We therefore propose to segment the loopy graph into a set of overlapping trees, while clamping the boundaries to their current estimated marginals (or observed marginals). We can now run IPF-BP on each tree and cycle through them until convergence. If every hidden node is unclamped at least once in every cycle we can proof that the algorithm decreases the objective at every step, and moreover only terminates at a stationary point. As a special case, this induces a stable alternative for loopy BP on graphs containing only hard evidence (and hidden nodes). In previous work we have called this kind of algorithm belief optimization [Welling and Teh, 2001].

2 Iterative Proportional Fitting as Belief Propagation

Consider a discrete, undirected, tree-structured graphical model T with pairwise potentials. Let i, j, k denote vertices, and (ij) denote an edge connecting nodes i and j. Each node i is associated with a discrete variable X_i, and let x_i, x_i' be some states of X_i. If S is a subset of nodes, or a subgraph of T, let $X_S = \{X_i\}_{i\in S}$ and $x_S = \{x_i\}_{i\in S}$. Let $X = X_T$ and $x = x_T$. Each node i has a marginal potential $\Psi_i(x_i)$ and each edge has a pairwise potential $\Psi_{ij}(x_i, x_j)$. The distribution over X is defined by

$$P(X = x) \propto \prod_{(ij)} \Psi_{ij}(x_i, x_j) \prod_i \Psi_i(x_i) \tag{1}$$

2.1 Belief propagation

Suppose observations $X_i = \hat{x}_i$ were made at evidence nodes $i \in V$. Belief propagation (BP) is the standard procedure for inferring the posterior distribution over the unobserved nodes H given the observations [Pearl, 1988]. The posterior distribution is fully described by the marginal and pairwise marginal distributions over nodes in H. BP is an iterative procedure where at each iteration a message is passed from one node to a neighboring node based on the incoming messages to the first node.

Let $i \in H$ be an unobserved node, and $j \in N(i)$ be a neighbor of i (j can be either observed or unobserved). Then the BP message $M_{ij}(x_j)$ from i to j is

$$M_{ij}(x_j) \propto \sum_{x_i} \Psi_{ij}(x_i, x_j)\Psi_i(x_i) \prod_{k\in N(i)\backslash j} M_{ki}(x_i) \tag{2}$$

where $N(i)\backslash j$ are all neighbors of i except j. If $i \in V$ is an observed node, then the outgoing message of i to $j \in N(i)$ is held fixed at the boundary conditions of $M_{ij}(x_j) \propto \Psi_{ij}(\hat{x}_i, x_j)\Psi_i(\hat{x}_i)$. For numerical stability, the messages are often normalized after each update. After BP has converged, the marginals and pairwise marginals are computed from the messages using

$$P(x_i) = b_i(x_i) \propto \Psi_i(x_i) \prod_{k\in N(i)} M_{ki}(x_i) \tag{3}$$

$$P(x_i, x_i) = b_{ij}(x_i, x_j) \propto \Psi_{ij}(x_i, x_j)\Psi_i(x_i)\Psi_j(x_j) \prod_{k\in N(i)\backslash j} M_{ki}(x_i) \prod_{l\in N(j)\backslash i} M_{lj}(x_j) \tag{4}$$

BP messages can be updated in series or in parallel. The parallel BP scheme is not practical on a normal computer, but is useful when one has a large graph and highly parallel hardware with one processor per node sending and receiving messages from its neighbors. Consider one such computer for node i. The computation of message M_{ij} from i to j requires the product of all incoming messages except M_{ji}. We can make BP more efficient by first computing the product of all incoming messages, and then dividing out M_{ji} to compute M_{ij} for each $j \in N(i)$.

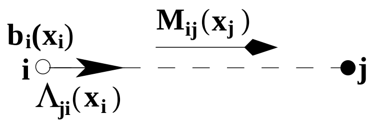

Figure 1: Two nodes and the messages between them.

That is, rewrite the BP update rule (2) as a sequence of three updates[2] :

$$b_i(x_i) \propto \Psi_i(x_i) \prod_{k \in N(i)} M_{ki}(x_i) \tag{5}$$

$$\Lambda_{ji}(x_i) = \frac{b_i(x_i)}{\Psi_i(x_i) M_{ji}(x_i)} \tag{6}$$

$$M_{ij}(x_j) = \sum_{x_i} \Psi_{ij}(x_i, x_j) \Psi_i(x_i) \Lambda_{ji}(x_i) \tag{7}$$

The updates (5) are used only for unobserved nodes $i \in H$. However if for each observed node $i \in V$ we clamp $b_i(x_i) = \delta_{x_i, \hat{x}_i}$ to have probability 1 on the observation $X_i = \hat{x}_i$, then the updates (6,7) can be defined for the observed nodes i too, and reduces to holding $M_{ij}(x_j) \propto \Psi_{ij}(\hat{x}_i, x_j)\Psi_i(\hat{x}_i)$ fixed.

The beliefs and messages relating nodes i and j are depicted in figure 1. The arrows for Λ_{ji} and M_{ij} depict the direction of information flow. By a filled circle, we will mean a node whose marginals are clamped to some fixed values, while an unfilled circle means an unclamped node.

2.2 A minimum divergence problem

We shall show that the same BP update rules (5,6,7) can be used to solve a minimum divergence problem which can be understood as a generalization of the inference problem in the previous section.

Let $i \in V$ be an observed node. Suppose instead of a "hard" observation where $X_i = \hat{x}_i$, we have a "soft" observation where we observe $X_i = x_i$ with probability $\hat{b}_i(x_i)$. Then our minimum divergence problem is to find the distribution $Q(x)$ minimizing $KL(Q\|P)$ while satisfying the constraints $Q(x_i) = \hat{b}_i(x_i)$. The following theorem shows that when we have hard evidence, i.e. $\hat{b}_i(x_i) = \delta_{x_i, \hat{x}_i}$, the minimum divergence problem reduces to inference.

Theorem 1 *Suppose we observed $X_i = \hat{x}_i$ at nodes $i \in V$. The distribution $Q(x)$ which minimizes $KL(Q\|P)$ and satisfies the constraints $Q(x_i) = \delta_{x_i, \hat{x}_i}$ for each $i \in V$ is given by $Q(x) = P(x_H|\hat{x}_V)\prod_{i \in V} \delta_{x_i, \hat{x}_i}$.*

Proof. As $Q(x)$ has to satisfy the constraints, we have $Q(x_V) = \prod_{i \in V} \delta_{x_i, \hat{x}_i}$ hence $Q(x)$ has to

[2] In practice, for more efficiency, we can use the following updates instead of (6,7) :

$$\Lambda'_{ji}(x_i) = \frac{b_i(x_i)}{M_{ji}(x_i)} \qquad\qquad M_{ij}(x_j) = \sum_{x_i} \Psi_{ij}(x_i, x_j) \Lambda'_{ji}(x_i)$$

We used (6,7) because the $\Lambda_{ji}(x_i)$'s are exponentials of Lagrange multipliers which we shall introduce in the next section.

have the form $Q(x) = Q(x_H|\hat{x}_V)\prod_{i\in V}\delta_{x_i,\hat{x}_i}$. Now

$$
\begin{aligned}
KL(Q\|P) &= \sum_x Q(x)\left(\log Q(x) - \log P(x)\right) \\
&= \sum_{x_H}\sum_{x_V} Q(x_H, x_V)\left(\log Q(x_H|\hat{x}_V) + \sum_{i\in V}\log\delta_{x_i,\hat{x}_i} - \log P(x)\right) \\
&= \sum_{x_H} Q(x_H|\hat{x}_V)\left(\log Q(x_H|\hat{x}_V) - \log P(x_H|x_V)\right) - \log P(\hat{x}_V) \qquad (8)
\end{aligned}
$$

and now minimizing (8) gives $Q(x_H|\hat{x}_V) = P(x_H|\hat{x}_V)$. □

$Q(x) = P(x_H|\hat{x}_V)\prod_{i\in V}\hat{b}_i(x_i)$ is a trivial extension of the posterior distribution $P(x_H|\hat{x}_V)$ to x_T. This justifies the minimum divergence problem as an extension of inference. We shall refer to the minimum divergence problem as generalized inference, and to the minimum divergence $Q(x)$ as the generalized posterior distribution.

Our minimum divergence problem is quite similar in flavor to using soft evidence. In soft evidence, instead of observing X_i, we observe another random variable $Z_i = \hat{z}_i$ where Z_i is related to X_i via a directed observation model $P(Z_i|X_i)$. With observations of Z_i, we then infer the posterior distribution over X. Soft evidence is similar to our minimum divergence problem in that the evidence on X_i in both are soft. However in soft evidence the posterior marginal distribution for X_i is not given and the evidence only acts as a bias affecting X_i. In our problem, the marginal distribution over X_i is given and is required to be satisfied. Ordinary BP can be used to find the posterior given soft evidence but unfortunately the minimum divergence problem cannot be solved so easily. In the next section, we introduce an algorithm to solve it which is very similar to BP.

2.3 IPF-BP

Generalized iterative proportional fitting (GIPF) is a procedure for solving general constrained minimum divergence problems [Darroch and Ratcliff, 1972]. It can be used to solve our generalized inference problem as well. Since our distributions are defined over a tree, BP is required as an inner loop to compute marginal distributions. In this section we describe the relationship between the GIPF outer loop and the BP inner loop and introduce a unifying algorithm which we call IPF-BP to solve the generalized inference problem. IPF-BP consists of the same BP updates (5,6,7) performed in a particular order to ensure convergence.

Theorem 2 *Suppose we made observations $Q(x_i) = \hat{b}_i(x_i)$ for $i \in V$. There is an ordering of the IPF-BP updates (5,6,7), with $b_i(x_i)$ clamped at $\hat{b}_i(x_i)$ for each $i \in V$, which guarantees that IPF-BP will converge. Further, the beliefs given by (3,4) are the marginal and pairwise marginal distributions of $Q(x)$, where $Q(x)$ minimizes $KL(Q\|P)$ while satisfying the constraints $Q(x_i) = \hat{b}_i(x_i)$ for each $i \in V$. $Q(x)$ itself is given by*

$$Q(x) \propto P(x)\prod_{i\in V}\left(\left(\frac{\Psi_i(x_i)}{\hat{b}_i(x_i)}\right)^{n_i-1}\prod_{j\in N(i)}\Lambda_{ji}(x_i)\right) \qquad (9)$$

where n_i is the number of neighbors of i.

Proof. We can solve for the posterior distribution $Q(x)$ using Lagrange multipliers. The Lagrangian is

$$L = \sum_x Q(x)\left(\log Q(x) - \log P(x)\right) - \sum_{i\in V}\sum_{x_i}\lambda_i(x_i)(Q(x_i) - \hat{b}_x(x_i)) \qquad (10)$$

Solving for $Q(x)$, we get

$$Q(x) \propto P(x) \prod_{i \in V} e^{\lambda_i(x_i)} \propto \prod_{(ij)} \Psi_{ij}(x_i, x_j) \prod_{i \in H} \Psi_i(x_i) \prod_{i \in V} \left(\Psi_i(x_i) e^{\lambda_i(x_i)} \right) \tag{11}$$

where $\lambda_i(x_i)$'s are chosen to satisfy the constraints.

We shall first prove the result when V consists of only leaf nodes (i.e. nodes with only one neighbor). Then we shall show the general case by applying the algorithm and proof to each segment of the tree separated by V, and patching the segments together using the Markov property of trees. The BP algorithm remains unchanged.

First suppose that V consists of only leaf nodes. The Lagrange multipliers in (11) can be solved using GIPF, which at each iteration updates a subset $U \subset V$ of the Lagrange multipliers with the update rule

$$e^{\lambda_i(x_i)} \leftarrow e^{\lambda_i(x_i)} \left(\frac{\hat{b}_i(x_i)}{Q(x_i)} \right)^{1/|U|} \tag{12}$$

where $Q(x)$ is given by (11) with the current setting of the Lagrange multipliers. We shall in particular use the serial version of GIPF where U consists of a single node i. The update rule is thus

$$e^{\lambda_i(x_i)} \leftarrow e^{\lambda_i(x_i)} \frac{\hat{b}_i(x_i)}{Q(x_i)} \tag{13}$$

$Q(x_i)$ can be computed in an inner loop using the usual BP procedure with no evidence nodes and $Q(x)$ given by (11). From (3), it is given by

$$Q(x_i) \propto \left(\Psi_i(x_i) e^{\lambda_i(x_i)} \right) M_{j_i i}(x_i) \tag{14}$$

where j_i is the unique neighbor of node i. Substituting (14) into (13), the serial GIPF update rule then reduces to

$$e^{\lambda_i(x_i)} \leftarrow \frac{\hat{b}_i(x_i)}{\Psi_i(x_i) M_{j_i i}(x_i)} \tag{15}$$

which is exactly the same as the BP update for $\Lambda_{j_i i}(x_i)$ given in (6). Identifying $e^{\lambda_i(x_i)} = \Lambda_{j_i i}(x_i)$, we see that the two loop serial IPF algorithm can be reduced to a single run of the BP procedure given by (5,6,7) with $b_i(x_i)$ fixed at $\hat{b}_i(x_i)$ for each $i \in V$. We shall call this single run BP procedure IPF-BP. IPF-BP updates of $\Lambda_{ji}(x_i)$ where i is a clamped node will be called IPF updates, other updates will be called internal BP updates. Note that our derivation of IPF-BP assumes a particular ordering of the updates : after each IPF update, we have to perform a batch of internal BP updates until the internal BP messages have converged. In section 2.4 we show that more efficient update orderings are possible.

Now suppose that V contains internal nodes of the tree T. The minimum divergence distribution $Q(x)$ is the unique distribution of the form (11) which satisfies the constraints. We will construct a distribution satisfying the above properties, which by uniqueness, must be $Q(x)$.

First we partition the tree T into a number of segments as follows. Each node $i \in V$ is replicated n_i times and each replica of i is connected to a neighbor $j \in N(i)$ and no other nodes. A segment is defined to be a connected component of the resulting graph. Aside from being connected to different neighbors of i, there is no difference between the replicas of i, and we will in general make no distinction between the individual replicas and the node i itself, and will identify each segment with the corresponding subtree of T. The same segmentation scheme can be applied to any graph G with a number of clamped nodes. This is depicted in figure 2. Note that a single edge can form a segment (bottom right of graph), and a segment can be singly connected even though the original subgraph is not (top right).

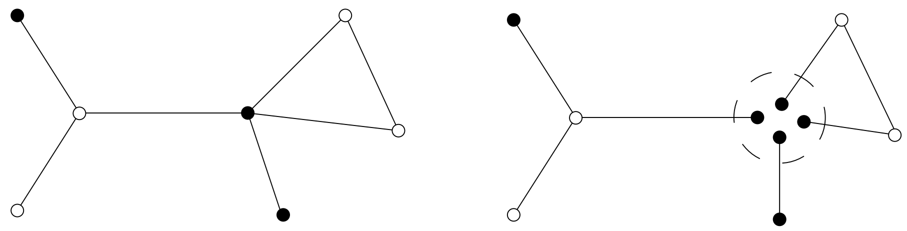

Figure 2: Segmenting a graph. On the left is the original graph, with filled circles being clamped nodes, and unfilled circles being unclamped nodes. The clamped nodes are replicated on the right and each connected component forms a segment of the original graph.

Since the clamped nodes of T appear only as leaf nodes of segments of T, we can run the previously described IPF-BP algorithm independently on each segment S of T. The resulting posterior distribution on S is

$$Q_S(x_S) \propto \prod_{(ij)\in S} \Psi_{ij}(x_i, x_j) \prod_{i\in S} \Psi_i(x_i) \prod_{i\in V\cap S} \Lambda_{j_i^S i}(x_i) \tag{16}$$

where j_i^S is the unique neighbor of i in S. Define

$$Q(x) = \frac{\prod_S Q_S(x_S)}{\prod_{i\in V} \hat{b}_i(x_i)^{n_i-1}} \tag{17}$$

where we take $0/0 = 0$. Because T is a tree and we have $Q_S(x_i) = \hat{b}_i(x_i)$ for each S adjacent to $i \in V$, it can easily be shown that $\sum_x Q(x) = 1$ hence $Q(x)$ is a distribution. Further, for each $i \in V$, $Q(x_i) = \hat{b}_i(x_i)$ satisfies the constraint. Now expanding each Q_S in (17) using (16), we see that

$$Q(x) \propto P(x) \prod_{i\in V} \left(\left(\frac{\Psi_i(x_i)}{\hat{b}_i(x_i)} \right)^{n_i-1} \prod_{j\in N(i)} \Lambda_{ji}(x_i) \right) \tag{18}$$

has the same form as (11). By uniqueness, $Q(x)$ must be the sought after generalized posterior distribution. □

2.4 Scheduling of IPF-BP messages

There are a few differences between IPF-BP and ordinary BP. Firstly, ordinary BP can converge with only one update to each message on the tree. This is achieved with the following scheme : update a message M_{ij} only if each incoming message M_{ki} for $k \in N(i)\backslash j$ has already been updated or is a fixed message coming in from an observed node. IPF-BP, on the other hand, cannot in general converge exactly to the correct solution with a finite number of steps, since simple GIPF itself cannot do so either.

Secondly, the updates of ordinary BP can be performed in parallel, while the IPF updates in IPF-BP cannot be parallelized unless one takes small steps, i.e. use $U = V$ in the GIPF update (12). Ordinary BP can be parallelized because the value of each M_{ij} never directly or indirectly affect the value of M_{ji}. With IPF-BP, a message going into a clamped node will "bounce" off the node and affect the message going out of the node. As a result every message is dependent on every other message which implies that IPF-BP cannot be parallelized.

Thirdly, the proof of theorem 2 assumes a particular ordering of message updates where all internal BP messages are updated before each IPF update. Since the convergence of IPF-BP is determined by the number of IPF updates performed, this is very inefficient since it requires a full sweep of internal BP updates for each IPF update. However, it turns out that we do not actually need to update all internal BP messages in between two consecutive IPF updates.

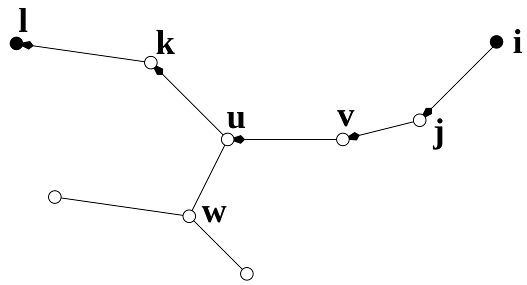

Figure 3: The internal BP update path from i to l is given by the arrowed edges.

Suppose the first IPF update is for $\Lambda_{ji}(x_i)$ and the second update is for $\Lambda_{kl}(x_l)$. See figure 3. Then we only need to update the messages along the path from i to l. For each unclamped node v traversed on the path from i to l, we update $b_v(x_v)$ and for edge $v \to u$ traversed we first update $\Lambda_{uv}(x_v)$ then update $M_{vu}(x_u)$. The reason these internal BP updates are sufficient is because updating $\Lambda_{kl}(x_l)$ requires only $M_{kl}(x_l)$ which in turn only requires messages pointing toward l to be computed. Among these messages, only those laying along the path from i to l are affected by the first IPF update of $\Lambda_{ji}(x_i)$ and hence need to be recomputed. Messages like $M_{wu}(x_u)$ in figure 3 remain unchanged after the first IPF update.

The above describes the minimum set of internal BP updates required between two IPF updates. We still have freedom in deciding how to schedule the IPF updates. For efficiency, it is desirable to schedule the IPF updates such that the number of internal BP updates per IPF update is minimized. This can be achieved by minimizing the length of the path traversed from one IPF update site to the next.

3 Loopy Extension to IPF-BP

Recently, a number of groups have shown that BP works surprisingly well in many practical problems where the graphical model involved contains loops [Frey and MacKay, 1997, McEliece et al., 1998, Freeman and Pasztor, 1998, Murphy et al., 1999]. Efforts to understand how and why loopy BP works were spear headed by Weiss [Weiss, 2000], but the breakthrough came when Yedidia *et al* showed that the fixed points of loopy BP correspond exactly to the stationary points of the Bethe free energy [Yedidia et al., 2000]. The Bethe free energy is an approximation to the true free energy, and in practice the BP fixed points are found to be minima of the Bethe free energy. This motivates exploring the use of other algorithms to minimize the Bethe free energy. In this section we shall derive an algorithm based on IPF-BP to directly minimize the Bethe free energy. We refer to this algorithm as IPF-BO, where BO stands for belief optimization [Welling and Teh, 2001].

Suppose we have a second order, discrete, undirected graphical model G. Let i, j, k denote nodes of the model, and (ij) denote an edge connecting nodes i and j. Each node i is associated with a variable X_i, and let x_i be some states of X_i. Let $X = \{X_i\}$ and $x = \{x_i\}$. The distribution over X is defined by

$$P(X = x) \propto \prod_{(ij)} \Psi_{ij}(x_i, x_j) \prod_i \Psi_i(x_i) \tag{19}$$

where $\Psi_{ij}(x_i, x_j)$ denotes a pairwise potential between node i and j, while $\Psi_i(x_i)$ denotes the local potential for node i. Let the beliefs $b_i(x_i)$ be our estimate of $P(x_i)$, and $b_{ij}(x_i, x_j)$ be our estimate of $P(x_i, x_j)$. Define

$$\begin{aligned} E_{ij}(x_i, x_j) &= -\log\left[\Psi_{ij}(x_i, x_j)\Psi_i(x_i)\Psi_j(x_j)\right] \\ E_i(x_i) &= -\log\left[\Psi_i(x_i)\right]. \end{aligned} \tag{20}$$

The Bethe free energy is defined as

$$\begin{aligned} F_{bethe} =& \sum_{(ij)} \sum_{x_i, x_j} b_{ij}(x_i, x_j)\Big(E_{ij}(x_i, x_j) + \log b_{ij}(x_i, x_j)\Big) \\ &+ \sum_i (1 - n_i) \sum_{x_i} b_i(x_i)\Big(E_i(x_i) + \log b_i(x_i)\Big) \end{aligned} \tag{21}$$

We obtain estimates b_i, b_{ij} by minimizing F_{bethe} subject to constraints that every b_{ij} marginalizes down to b_i and b_j, and that every b_i sums to 1. We can enforce these constraints by adding Lagrange multipliers to the free energy as follows[3],

$$\begin{aligned} L =& F_{bethe} - \sum_{(ij)} \sum_{x_i} \lambda_{ji}(x_i)\Big(\sum_{x_j} b_{ij}(x_i, x_j) - b_i(x_i)\Big) \\ &- \sum_{(ij)} \sum_{x_j} \lambda_{ij}(x_j)\Big(\sum_{x_i} b_{ij}(x_i, x_j) - b_j(x_j)\Big) - \sum_i \gamma_i\Big(\sum_{x_i} b_i(x_i) - 1\Big) \end{aligned} \tag{22}$$

It was shown in [Yedidia et al., 2000], that the stationary points of this cost function correspond to the fixed points of loopy BP.

In the following we will describe an iterative procedure to decrease the Bethe free energy at every iteration and show that the procedure stops only when a stationary point has been found. At each iteration a number of nodes are chosen to have their marginals clamped at the current values. These clamped nodes define a segmentation of the graph as in figure 2. The nodes are chosen such that each resulting segment forms a tree, and IPF-BP is applied to

[3] Notice that unlike [Yedidia et al., 2000], we do not include Lagrange multipliers which directly enforce the pairwise marginals to integrate to one, which are redundant.

each segment with the aforementioned nodes clamped to their current marginals. After IPF-BP converges, a new iteration starts with a new set of nodes chosen to have their marginals clamped. We will show that each iteration of this algorithm decreases the Bethe free energy, hence is guaranteed to converge since the Bethe free energy is bounded below for discrete networks. So long as no node stays clamped at every iteration, we shall show furthermore that the algorithm will converge to a stationary point of the Bethe free energy (it can be a saddle point or local minimum).

Let C be a set of nodes and consider a segment T of the graph after we clamp the nodes of C to their marginals. T is assumed to be singly-connected. We wish to minimize F_{bethe} with respect to the beliefs corresponding to the edges and unclamped nodes of T. The relevant terms of L are

$$\begin{aligned} L_T = & \sum_{(ij)\in T}\sum_{x_i,x_j} b_{ij}(x_i,x_j)\Big(E_{ij}(x_i,x_j)+\log b_{ij}(x_i,x_j)\Big) \\ & +\sum_{i\in T}(1-n_i^T)\sum_{x_i} b_i(x_i)\Big(E_i(x_i)+\log b_i(x_i)\Big) \\ & -\sum_{(ij)\in T}\sum_{x_i}\lambda_{ji}(x_i)\Big(\sum_{x_j} b_{ij}(x_i,x_j)-b_i(x_i)\Big) \\ & -\sum_{(ij)\in T}\sum_{x_j}\lambda_{ij}(x_j)\Big(\sum_{x_i} b_{ij}(x_i,x_j)-b_j(x_j)\Big) \\ & -\sum_{i\in T}\gamma_i\Big(\sum_{x_i} b_i(x_i)-1\Big) \end{aligned} \tag{23}$$

where n_i^T is the number of neighbors of node i in T. We can minimize L with respect to the unclamped beliefs of T by minimizing L_T. Because T is a tree, if each b_{ij} marginalizes down to b_i and b_j, we can define a distribution over T by

$$b_T(x_T) = \frac{\prod_{(ij)\in T} b_{ij}(x_i,x_j)}{\prod_{i\in T} b_i(x_i)^{n_i^T-1}} \tag{24}$$

The problem of the constrained minimization of L_T with respect to the unclamped beliefs of T is equivalent to the problem of minimizing

$$\sum_{x_T} b_T(x_T)\left(\log b_T(x_T)+\sum_{(ij)\in T}E_{ij}(x_i,x_j)+\sum_{i\in T}(1-n_i^T)E_i(x_i)\right) \tag{25}$$

subject to the constraints that b_T has to be of the form given in (24) and that b_T has to marginalize down to $b_i(x_i)$ for each clamped node $i\in C\cap T$. Define a distribution over T by

$$P_T(x_T)\propto \exp\left(-\sum_{(ij)\in T}E_{ij}(x_i,x_j)-\sum_{i\in T}(1-n_i^T)E_i(x_i)\right) \tag{26}$$

By theorem 2, the the minimum divergence distribution Q_T from P_T with constraints $Q_T(x_i) = b_i(x_i)$ for each $i\in C\cap T$ is

$$Q_T(x_T)\propto P_T(x_T)\prod_{i\in C\cap T}\exp\left(\gamma_i(x_i)\right) \tag{27}$$

where $\gamma_i(x_i)$ are chosen to satisfy the constraints. Because T is a tree, a Q_T given by (27) can also be expressed in the form

$$Q_T(x_T) = \frac{\prod_{(ij)\in T} Q_T(x_i,x_j)}{\prod_{i\in T} Q_T(x_i)^{n_i^T-1}} \tag{28}$$

Now (25) is, up to an additive constant, $KL(b_T \| P_T)$, hence Q_T is the solution to the constrained minimization of (25).

By theorem 2, we can find Q_T using IPF-BP, and so IPF-BP can be used to decrease F_{bethe} at each iteration. Now we will show that the overall algorithm converges to a stationary point of F_{bethe} if no node is left clamped in every iteration. Suppose that the algorithm has already converged. Let i be a node and j be a neighbor of i, and consider an iteration where i is unclamped. Let T be the segment containing i and consider the IPF-BP updates of T. Since the overall algorithm has converged, the update rules (5,6,7) do not change b_i, Λ_{ij} or M_{ji}. Consider b_{ij} as given by (4). We have

$$
\begin{aligned}
\sum_{x_j} b_{ij}(x_i, x_j) &\propto \sum_{x_j} \Psi_{ij}(x_i, x_j)\Psi_i(x_i)\Psi_j(x_j) \prod_{k \in N(i)\backslash j} M_{ki}(x_i) \prod_{l \in N(j)\backslash i} M_{lj}(x_j) \\
&= \sum_{x_j} \Psi_{ij}(x_i, x_j)\Psi_i(x_i)\Psi_j(x_j)\Lambda_{ij}(x_j)\Lambda_{ji}(x_i) && \text{by (5,6)} \\
&= \Psi_i(x_i)\Lambda_{ji}(x_i)M_{ji}(x_i) && \text{by (7)} \\
&= \Psi_i(x_i) \prod_{k \in N(i)} M_{ki}(x_i) && \text{by (5,6)} \\
&= b_i(x_i) && \text{by (3)}
\end{aligned}
\tag{29}
$$

So the beliefs satisfy the marginalization and normalization constraints. Since a set of beliefs b_i and b_{ij} are at the stationary point of F_{bethe} if they are given by (3) and (4), and they satisfy the marginalization and normalization constraints [Yedidia et al., 2000], our overall algorithm has converged to a stationary point of F_{bethe}.

4 Comparing Loopy BP and IPF-BO

We investigated the practicality of minimizing the Bethe free energy using IPF-BO by comparing the accuracy of its estimated marginal and pairwise marginal distributions with loopy BP.

We ran all our experiments on a 10 by 10 square lattice with binary $(0, 1)$-valued units. This is because it is still feasible to compute the exact marginal and pairwise marginal distributions, against which we can compare the estimates obtained using loopy BP and IPF-BO. The exact marginal and pairwise marginal distributions are calculated using the junction tree algorithm where we cluster the nodes in each row into a super-node. We sampled each weight W_{ij} independently from a zero mean Gaussian with standard deviation s_w. Then we sampled each bias b_i independently from a Gaussian with mean $-\frac{1}{2}\sum_{k \in N(i)} W_{ik}$ and standard deviation s_b. The means of the biases are offset from 0 so that networks with biases deviating only slightly from their means have complex multi-modal distributions where the mean value of each X_i is approximately 0.5, while if the biases deviate more from their means the distribution will tend to be peaked around a single mode with the mean value of X_i polarized at either 0 or 1. That is, if s_b is small the resulting distribution will be qualitatively more similar to a prior distribution before observations were made, and if s_b is large the resulting distribution will be qualitatively more similar to the posterior after observing some nodes. The standard deviations s_w and s_b are chosen from $\{0.1, 1, 3, 6, 10\}$ separately. For each setting of s_w and s_b, 20 networks are generated and used to compare loopy BP and IPF-BO. For large weights $s_w \geq 6$ and small biases $s_b \leq 3$, we generated and used 40 networks instead, as loopy BP does not converge all the time.

For IPF-BO, we iterate over the nodes i of the network, clamping the neighboring marginals $b_j(x_j), j \in N(i)$ to their current values and running IPF-BO on the star-shaped segment containing i. For loopy BP, we used a strong damping factor of 0.9 so that it has a higher chance of convergence. For both algorithms, the convergence criteria was for all the means $b_i(1)$ to be changed by less than 10^{-4} for twenty consecutive iterations. We stop if loopy BP has not converged by 10000 iterations.

For a given setting of s_w and s_b, the generated networks are separated into two sets : one in which loopy BP converged, and one in which it failed to converge. For each set separately, we compared IPF-BO and loopy BP using the mean error in the estimated marginals $b_i(1)$ averaged over all nodes and all networks in the set. We also compared the mean error in the estimated covariances $b_{ij}(1, 1) - b_i(1) - b_j(1)$, averaged over all neighboring pairs of nodes and all networks in each set. Accompanying each mean we also looked at the mean absolute deviation (MAD). The results are given in figure 4.

Each row of figure 4 corresponds to a setting of s_w, increasing from top to bottom. Within each row the left plot shows the errors in the estimated marginals, while the right plot shows the errors in the estimated covariances. In each plot there are five groups of four bars each. Each group corresponds to a setting of s_b, increasing from left to right. In each group, the first two bars show the errors using IPF-BO and loopy BP respectively, when loopy BP converged. The next two bars show the errors for both IPF-BO and loopy BP when loopy BP failed to converge in 10000 iterations. The number associated with each group indicates the percentage of runs that loopy BP failed to converge.

The qualitative behaviors of the errors of the marginals and covariances are the same. Hence we shall concentrate on the errors of the marginals. The general trends in figure 4 confirm our expectations. With increasing weights, both loopy BP and IPF-BO performed increasingly worse, as the distribution becomes more complicated and multi-modal. With increasing biases, both loopy BP and IPF-BO performed better, as the distribution tends toward a single mode.

For small weights or large biases ($s_w \leq 3$, or $s_b \geq 6$, or $s_w = 6$ and $s_b \geq 1$), loopy BP always

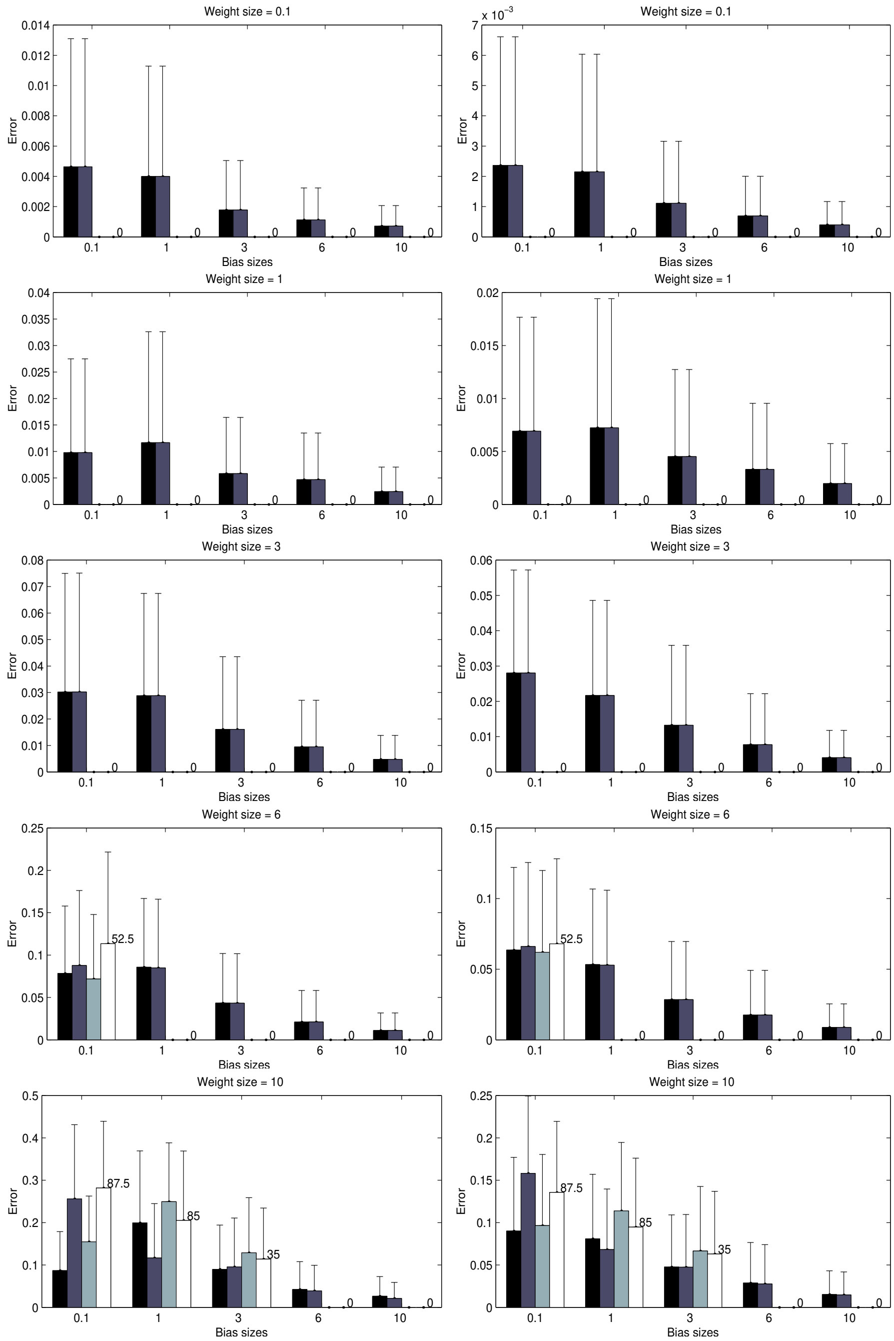

Figure 4: Errors in the estimated marginals (left) and the estimated covariances between neighboring nodes (right).

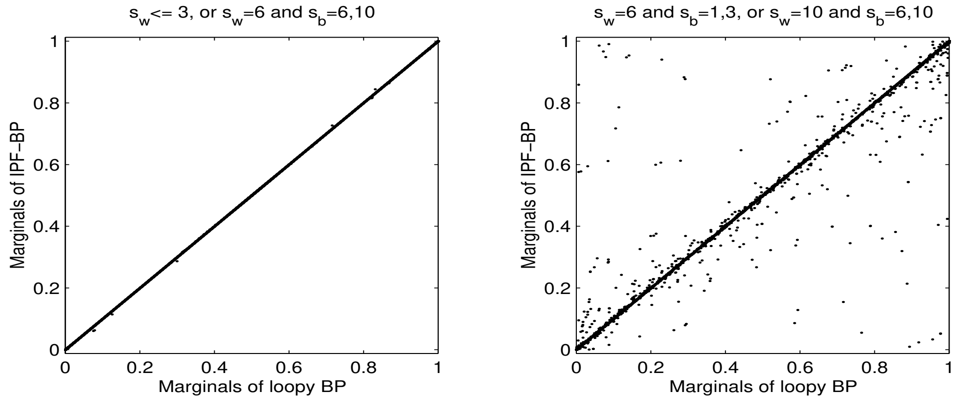

Figure 5: Scatter plots of the estimated marginals $b_i(1)$ for IPF-BO on the x-axis and for loopy BP on the y-axis. The left plot is for networks with $s_w \leq 3$ or $s_w = 6$ and $s_b = 6, 10$. The right plot is for networks with $s_w = 6$ and $s_b = 1, 3$, or $s_w = 10$ and $s_b = 6, 10$.

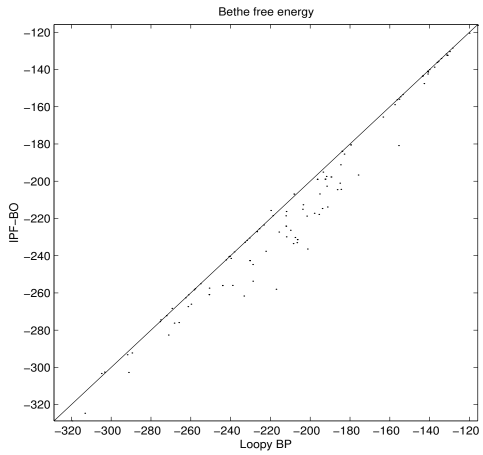

Figure 6: Scatter plot of Bethe free energy obtained using IPF-BO versus those obtained using loopy BP.

converged and both algorithm performed equally well. As a matter of fact, most of the time both algorithms converged to very similar solutions. This is shown in figure 5, where we plot the marginals obtained by IPF-BO versus those obtained by loopy BP. On the right plot, the algorithms sometimes get stuck in local minima or plateaus, resulting in a very small number of marginals being different : out of a total of 12000 points on the plot, only 372 lie outside the region $|x - y| \leq 0.01$.

The situation is more complicated for large weights and small biases ($s_w = 6$ and $s_b = 0.1$, or $s_w = 10$ and $s_b \leq 3$). In the regime where $s_b = 0.1$ and $s_w = 6, 10$, IPF-BO performed better than loopy BP, especially when $s_w = 10$. In the regime where $s_b \geq 1$ and $s_w = 10$, loopy BP amazingly performed better than IPF-BO even when loopy BP did not converge.

One possible explanation for this phenomenon is that IPF-BO is stuck in local minima or plateaus, in which case we can diagnose this by seeing if the Bethe free energy of the final beliefs obtained using IPF-BO is larger than the Bethe free energy of the beliefs obtained using loopy BP. This is shown in figure 6. We see that the reverse is true instead – IPF-BO always converges to a point where the Bethe free energy is lower than the Bethe free energy obtained with loopy BP. This shows that IPF-BO is not stuck in local minima and also shows that IPF-BO does what it was advertised to do – to decrease the Bethe free energy.

To understand why IPF-BO gives larger errors than loopy BP we look at how the marginals

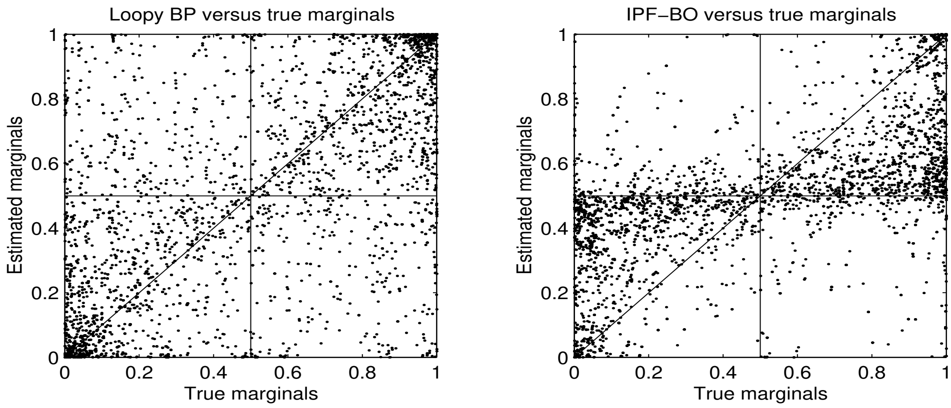

Figure 7: Scatter plot of loopy BP and IPF-BO marginals versus the true marginals for networks with $v_w = 10, w_b = 1$.

estimated by IPF-BO and loopy BP are related to the true marginals. This is shown in figure 7. We did not separate out cases where loopy BP converged from those where it did not because the analyses turn out to be similar anyway. Consider first the left plot of loopy BP marginals versus the true marginals. Most of the points are concentrated near the $(1, 1)$ and $(0, 0)$ corner. This means that if a true marginal is close to 0 or 1, loopy BP often converges to a limit cycle or stationary point close to the true marginal. Otherwise the loopy BP marginal can be totally unrelated to the true marginal, as seen by the uniform spread of the points on the plot away from $(0, 0)$ and $(1, 1)$. In summary, loopy BP often got the right marginal but sometimes got it totally wrong. Now consider the right plot of IPF-BO marginals versus the true marginals. Since there are not many points in the top left and bottom right quadrants, we see that the IPF-BO marginals are often on the same side of 0.5 as the true marginals. The problem lies with the (almost) horizontal ridge of points, where regardless of what the true marginal is, the IPF-BO estimate is often close to 0.5 (even though the IPF-BO estimate might lie on the same side of 0.5). This is true even when the true marginal is near to 0 or 1 (observe the two clumps of points, one near $(0, 0.5)$ and one near $(1, 0.5)$). It is the points near $(0, 0.5)$ and $(1, 0.5)$ which contributed to the high error as report in figure 4. IPF-BO prefers its marginals to be near 0.5 because they give a lower Bethe free energy, as seen in figure 6.

The same analysis shows why IPF-BO does better than loopy BP when $v_w = 6, 10$ and $v_b = 0.1$. The results are shown in figure 8 for $v_w = 10$. The results are similar for $v_w = 6$.

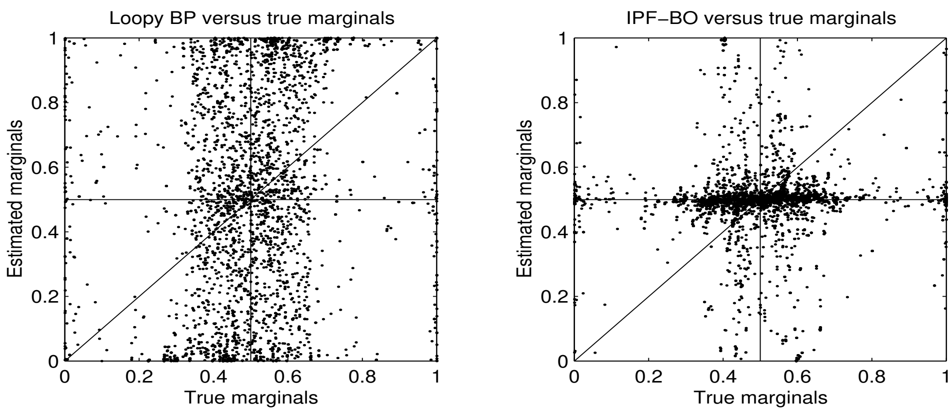

Figure 8: Scatter plot of loopy BP and IPF-BO marginals versus the true marginals for networks with $v_w = 10, w_b = 0.1$.

Again we did not distinguish between whether loopy BP converged or not as the analyses were similar. First of all note that because the biases are so small, the true marginals are mostly between 0.3 and 0.7. Loopy BP did not converge in 87.5% of the networks, and we can see from figure 8 that the marginals it estimated are essentially random. For IPF-BO, the points in the right plot of figure 8 can be approximately split into two strips : a horizontal strip from $(0, 0.5)$ to $(1, 0.5)$, and a less distinct vertical strip from $(0.5, 0)$ to $(0.5, 1)$. This means that IPF-BO marginals either close to 0.5 (horizontal strip), or are totally random (vertical strip). This should not be much better than what loopy BP did on the left plot. The reason the IPF-BO errors in figure 4 are so much smaller than the loopy BP errors is because the true marginals themselves are coincidentally often close to 0.5.

The above detailed analysis shows that loopy BP always converges when the Bethe approximation is good. Both IPF-BO and loopy BP will then converge to the same solutions in this case. If the Bethe approximation is bad, loopy BP often does not converge, but IPF-BO does not seem to do much better either. We can view this as an advantage for loopy BP – if loopy BP does not converge we can be certain that the Bethe approximation is not good to start with and perhaps we should look into more accurate approximations. The down side is that even if loopy BP converges we still cannot be sure if the Bethe approximation is good and the loopy BP estimates are good. However the stable nature of IPF-BO might make it more suitable as part of a network parameter learning scheme with approximate inference, as it is likely that we will encounter parameter settings for which loopy BP will not converge during the course of learning. Our conclusions are, however, limited by our use of synthetic randomly generated networks, and further experiments are necessary in understanding when and why direct minimization of the Bethe free energy might be better than loopy BP.

5 Discussion

In this work we have presented a unified view on inference and an algorithm to solve it efficiently on trees. We are currently investigating the possibility to perform exact generalized inference on loopy graphs which can be efficiently represented as junction trees, and its relation to Jirousek's and Preucil's algorithm.

Approximate generalized inference on loopy graphs is studied through the introduction of an extended Bethe free energy as an objective function to be minimized. An iterative algorithm which cycles through tree structured subgraphs while performing IPF-BP on each segmented tree was shown to minimize this objective efficiently. We briefly mentioned the possibility of segmenting small clusters of nodes containing loops and treating them exactly. Although initial experiments show promising results, we haven't yet found an appropriate costfunction which is being minimized. It it our hope that there is a close relationship with the Kikuchi free energy and the generalized BP algorithm [Yedidia et al., 2000].

In the experiment section we have tried to map out where the Bethe free energy is a good approximation. Theoretically, we know three regimes where it should be appropriate,

- On a tree, where it is exact.
- For small weights, since the correlations are expected to be short ranged, which is the appropriate regime for mean field type approximations.
- For very large weights if the interactions at most second order, since in this regime the entropy is negligable and the energy contribution is exact.

From an algorithmic perspective we have found good performance of both loopy BP and our belief optimization procedure for small weights (and on trees of course). However, when the weights grow too large and the biases remain small both loopy BP and BO procedures become inaccurate. In this regime, the lack of external evidence (in the form of small biases) will not produce a sharply peaked posterior distribution, but a distribution with many modes with polarized marginals. Since BP (when it converges) and BO will produce identical, but inaccurate estimates of the posterior marginals, we deduce, that the Bethe approximation must have broken down. We have found that this happens even before loopy BP fails to converge, making the use of minimization procedures like belief optimization not extremely useful for this application (one could even think of using the failure of BP to converge as a diagnostic tool for the accuracy of Bethe free energy). We have not found good performance in the regime of very large weights, irrespective of the biases (third regime), which may be explained by the rugged energy landscape of the posterior distribution.

Possible applications may involve learing where BO can be used as a partial E-step in an EM algorithm. It is our hope that other applications, possibly in the field of error correcting codes, will emerge in the future.

References

[Darroch and Ratcliff, 1972] Darroch, J. and Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480.

[Deming and Stephan, 1940] Deming, W. E. and Stephan, F. F. (1940). On a least square adjustment of a samplied frequency table when the expected marginal totals are known. *Annals of Mathematical Statistics*, 11:427–444.

[Freeman and Pasztor, 1998] Freeman, W. and Pasztor, E. (1998). Learning to estimate scenes from images. In *Advances in Neural Information Processing Systems*, volume 11.

[Frey and MacKay, 1997] Frey, B. and MacKay, D. (1997). A revolution: Belief propagation in graphs with cycles. In *Neural Information Processing Systems*, volume 10.

[Jiroušek and Přeučil, 1995] Jiroušek, R. and Přeučil, S. (1995). On the effective implementation of the iterative proportional fitting procedure. *Computational Statistics And Data Analysis*, 19:177–189.

[Lauritzen and Spiegelhalter, 1988] Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, 50.

[MacKay and Neal, 1995] MacKay, D. J. and Neal, R. M. (1995). Good codes based on very sparse matrices. In Boyd, C., editor, *Cryptography and Coding : 5th IAM Conference*, number 1025 in Lecture Notes in Computer Science, pages 100–111. Springer-Verlag.

[McEliece et al., 1998] McEliece, R., MacKay, D., and Cheng, J. (1998). Turbo decoding as an instance of pearl's belief propagation algorithm. *IEEE J. Selected Areas in Communication, 1997*, 16:140–152.

[Murphy et al., 1999] Murphy, K., Weiss, Y., and Jordan, M. (1999). Loopy belief propagation for approximate inference : An empirical study. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, volume 15. Morgan Kaufmann Publishers.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann Publishers, San Mateo CA.

[Pietra et al., 1997] Pietra, S. D., Pietra, V. D., and Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393.

[Weiss, 2000] Weiss, Y. (2000). Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12:1–41.

[Welling and Teh, 2001] Welling, M. and Teh, Y. W. (2001). Belief optimization for binary networks : A stable alternative to loopy belief propagation. In *Uncertainty in Artificial Intelligence*.

[Yedidia et al., 2000] Yedidia, J., Freeman, W., and Weiss, Y. (2000). Generalized belief propagation. In *Advances in Neural Information Processing Systems*, volume 13.