A Double-Loop Algorithm to Minimize the Bethe and Kikuchi Free Energies

A. L. Yuille
Smith-Kettlewell Eye Research Institute,
2318 Fillmore Street,
San Francisco, CA 94115, USA.
Tel. (415) 345-2144. Fax. (415) 345-8455.
Email yuille@ski.org

March 15, 2001

Abstract

Recent work (Yedidia, Freeman, Weiss [22]) has shown that stable points of belief propagation (BP) algorithms [12] for graphs with loops correspond to extrema of the Bethe free energy [3]. These BP algorithms have been used to obtain good solutions to problems for which alternative algorithms fail to work [4], [5], [10] [11]. In this paper we first obtain the dual energy of the Bethe free energy which throws light on the BP algorithm. Next we introduce a discrete iterative algorithm which we prove is guaranteed to converge to a minimum of the Bethe free energy. We call this the double-loop algorithm because it contains an inner and an outer loop. It extends a class of mean field theory algorithms developed by [7],[8] and, in particular, [13]. Moreover, the double-loop algorithm is formally very similar to BP which may help understand when BP converges. Finally, we extend all our results to the Kikuchi approximation which includes the Bethe free energy as a special case [3]. (Yedidia *et al* [22] showed that a "generalized belief propagation" algorithm also has its fixed points at extrema of the Kikuchi free energy). We are able both to obtain a dual formulation for Kikuchi but also obtain a double-loop discrete iterative algorithm that is guaranteed to converge to a minimum of the Kikuchi free energy. It is anticipated that these double-loop algorithms will be useful for solving optimization problems in computer vision and other applications.

Draft submitted to Neural Computation

1 Introduction

Local belief propagation (BP) algorithms [12] have long been known to converge to the correct marginal probabilities for tree-like graphical models. Recently, several researchers have empirically demonstrated that BP algorithms often perform surprisingly well for graphs with loops [4], [5], [10] [11] and, in particular, for practical applications such as learning low level vision [4] and turbo decoding [10]. (See [11], however, for examples where BP algorithms fail). It is important to understand when and why BP, and related algorithms, perform well on graphs with loops.

More recently Yedidia, Freeman and Weiss [22] proved that the stable fixed points of BP algorithms correspond to extrema of the Bethe free energy [3]. Therefore if a BP algorithm converges it must go to an extremum (i.e. maximum, minimum, or saddle point) of the Bethe free energy (and empirically this extremum seems always to a minimum [22]). Yedidia *et al* describe how these results can be generalized to Kikuchi approximations [3] which include the Bethe free energy as a special case. Overall, Yedidia *et al*'s work gives an exciting link between belief propagation and inference algorithms based on statistical physics.

This paper develops the connections between BP, Bethe free energy, the Kikuchi approximation, and statistical physics algorithms. We first show that the BP algorithm (as formulated by Yedidia *et al*) can be reformulated simply in the dual space of the free energy (which throws light on the algorithm). Our next, and main, result is a novel discrete iterative algorithm, called the *double-loop algorithm* which is guaranteed to converge to a minimum of the Bethe free energy. The double-loop algorithm is similar to BP because it also proceeds by passing "messages" between nodes of the graph (and there are interesting formal similarities between the two algorithms). The double-loop algorithm has its roots in statistical physics and, in particular, the use of mean field theory algorithms for solving optimization problems [7],[8],[13]. (Physicist readers should note that the "mean" is with respect to the Gibbs distribution but we do *not*, unlike most physics applications, assume that the "mean field" is spatially constant). Next, we extend the double-loop algorithm, and all the theoretical results, to the general Kikuchi approximation.

Finally, in Section (7.1), we place this work in context of mean field theory approaches to optimization (see [7],[8],[13] and chapters by Peterson and Yuille in [2]). This material is essentially a review (and some readers might prefer to read it before the rest of the paper).

Section (2) describes the Bethe free energy and BP algorithms. It also gives a dual formulation of the Bethe free energy which shows how the "messages" can be derived from Lagrange multipliers (which may motivate other algorithms). Section (3) gives the two basic design principles of our double-loop algorithm: (i) showing how to construct discrete iterative algorithms to minimize energy functions which are sums of a concave and a convex term, (ii) designing an iterative update algorithm guaranteed to enforce linear constraints. In Section (4) we apply these principles to the Bethe free energy and show that the specific nature of the constraints means that the iterative algorithm to solve the constraints takes a particularly simple form. Section (5) summarizes the double-loop algorithm and discusses formal similarities to BP. In Section (6) we show that all our results can be extended to the Kikuchi approximation and obtain both a dual energy function and a double-loop algorithm that is guaranteed to converge to a minimum of the Kikuchi free energy. We conclude in Section (7.1) by briefly discussing how the scheme in this paper can be extended to obtain different statistical estimators, how temperature annealing can be done, and who the approach relates directly to mean field theory algorithms.

2 The Bethe Free Energy, Duality, and the BP algorithms

In the first subsection we describe the Bethe free energy and the BP algorithm. In the next subsection we show that the Bethe free energy can be reformulated in dual space which throws some light on the BP dynamics.

2.1 The Bethe Free Energy and BP algorithms

This section sets up the problem following the formulation of Yedidia *et al* [22].

Consider a graph with nodes $i = 1, ..., N$. The state of a node is denoted by x_i (each x_i has M possible states). Each unobserved node is connected to an observed node y_i. The joint probability function is given by:

$$P(x_1, ..., x_N|y) = \frac{1}{Z} \prod_{i,j:i>j} \psi_{ij}(x_i, x_j) \prod_i \psi_i(x_i, y_i), \tag{1}$$

where $\psi_i(x_i, y_i)$ is the local "evidence" for node i, Z is a normalization constant, and $\psi_{ij}(x_i, x_j)$ is the (symmetric) compatibility matrix between nodes i and j. We use the convention $i > j$ to avoid double counting. To simplify notation, we write $\psi_i(x_i)$ as shorthand for $\psi_i(x_i, y_i)$. If nodes i, j are *not* connected then we set $\psi_{ij}(x_i, x_j) = 1$.

The goal is to determine estimate $\{b_i(x_i)\}$ of the marginal distributions $\{P(x_i|y)\}$. It is convenient also to make estimates $\{b_{ij}(x_i, x_j)\}$ of the joint distributions $\{P(x_i, x_j|y)\}$ of nodes i, j which are connected in the graph. (Again we use the convention $i > j$ to avoid double counting). The $\{b_i(x_i)\}$ can then be used to calculate approximations to the minimum variance (MV) estimators for the variables $\{x_i\}$, see section (7.1).

The BP algorithm introduces variables $m_{ij}(x_j)$ which correspond to "messages" that node i sends to node j (in the next subsection we will see how these messages arise as Lagrange multipliers). The BP algorithm is given by:

$$m_{ij}(x_j; t + 1) = c_{ij} \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_i(x_i) \prod_{k \neq j} m_{ki}(x_i; t), \tag{2}$$

where c_{ij} is a normalization constant (i.e. it is independent of x_j). We use the convention that $m_{ki}(x_i,t)=1$ if nodes i,k are *not* connected. Nodes are not connected to themselves (i.e $m_{ii}(x_i,t)=1,\ \forall\ i$).

The messages determine additional variables $b_i(x_i), b_{ij}(x_i,x_j)$ corresponding to the approximate marginal probabilities at node i and the approximate joint probabilities at nodes i,j (with convention $i>j$). These are given in terms of the messages by:

$$b_i(x_i;t) = \hat{c}_i \psi_i(x_i) \prod_k m_{ki}(x_i;t), \tag{3}$$

$$b_{ij}(x_i,x_j;t) = \bar{c}_{ij} \phi_{ij}(x_i,x_j) \prod_{k \neq j} m_{ki}(x_i;t) \prod_{l \neq i} m_{lj}(x_j;t), \tag{4}$$

where $\phi_{ij}(x_i,x_j)=\psi_{ij}(x_i,x_j)\psi_i(x_i)\psi_j(x_j)$ and $\hat{c}_i, \bar{c}_{ij}$ are normalization constants.

For a tree, the BP algorithm of equation (2) is guaranteed to converge and the resulting $\{b_i(x_i)\}$ will correspond to the posterior marginals [12].

The Bethe free energy of this system is written as [22]:

$$F_\beta(\{b_{ij},b_i\}) = \sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j)\log\frac{b_{ij}(x_i,x_j)}{\phi_{ij}(x_i,x_j)} - \sum_i (n_i-1)\sum_{x_i} b_i(x_i)\log\frac{b_i(x_i)}{\psi_i(x_i)}, \tag{5}$$

where n_i is the number of neighbours of node i.

Because the $\{b_i\}$ and $\{b_{ij}\}$ correspond to marginal and joint probability distributions they must satisfy *linear* consistency constraints:

$$\sum_{x_i,x_j} b_{ij}(x_i,x_j)=1,\ \forall\ i,j:\ i>j \quad \sum_{x_i} b_i(x_i)=1,\ \forall\ i,$$

$$\sum_{x_i} b_{ij}(x_i,x_j)=b_j(x_j),\ \forall\ j,x_j,\ \ \sum_{x_j} b_{ij}(x_i,x_j)=b_i(x_i),\ \forall\ i,x_i. \tag{6}$$

The Bethe free energy consist of two terms. The first is of the form of a Kullback-Leibler (K-L) divergence between $\{b_{ij}\}$ and $\{\phi_{ij}\}$ (but it is not actually a K-L divergence because $\{\phi_{ij}\}$ is not a normalized distribution). The second is *minus* the form of the Kullback-Leibler divergence between $\{b_i\}$ and $\{\psi_i\}$ (again $\{\psi_i\}$ is not a normalized probability distribution). It follows that the first term is *convex* in $\{b_{ij}\}$ and the second term is *concave* in $\{b_i\}$. This will be of importance for the derivation of our algorithm in section (3).

Yedidia *et al* [22] proved that the fixed points of the BP algorithm correspond to extrema of the Bethe free energy (with the linear constraints of equation (6)). The next subsection gives an alternative derivation of this result starting from a dual formulation of the Bethe free energy.

2.2 Duality and Dynamics of Belief Propagation

We now give a dual formulation of the Bethe free energy by solving for the primary variables $\{b_{ij},b_i\}$ in terms of the Lagrange multipliers used to impose the constraints, see equation (6). This dual formulation, see [18], gives an alternative way of thinking of the BP algorithm and demonstrates how the messages $\{m_{ij}\}$ arise from Lagrange multipliers.

We write the full Bethe free energy (including the linear constraints) as:

$$F(\{b_{ij}\},\{b_i\},\{\gamma_{ij}\},\{\lambda_{ij}\}) = \sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j)\log\frac{b_{ij}(x_i,x_j)}{\phi_{ij}(x_i,x_j)} - \sum_i (n_i-1)\sum_{x_i} b_i(x_i)\log\frac{b_i(x_i)}{\psi_i(x_i)}$$

$$+\sum_{ij:i>j}\gamma_{ij}\{\sum_{x_i,x_j} b_{ij}(x_i,x_j)-1\} + \sum_{i,j:i>j}\sum_{x_j}\lambda_{ij}(x_j)\{\sum_{x_i} b_{ij}(x_i,x_j)-b_j(x_j)\} + \sum_{i,j:i>j}\sum_{x_i}\lambda_{ji}(x_i)\{\sum_{x_j} b_{ij}(x_i,x_j)-b_i(x_i)\}, \tag{7}$$

where we have introduced Lagrange multipliers $\{\gamma_{ij}: i>j\}$ and $\{\lambda_{ij}(x_j): i \neq j\}$. Observe there are no self-interaction terms like $\lambda_{ii}(x_i)$. (Alternatively one can keep these terms in and set them automatically to zero).

Theorem 1. *Extremizing the Bethe free energy is equivalent to extremizing the dual energy* $G(\{\gamma_{ij}\},\{\lambda_{ij}\})$ *given by:*

$$G(\{\gamma_{ij}\},\{\lambda_{ij}\}) = -\sum_{i,j:i>j}\sum_{x_i,x_j}\phi_{ij}(x_i,x_j)e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}}e^{-1}e^{-\gamma_{ij}} + \sum_i (n_i-1)\sum_{x_i}\psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_j \lambda_{ji}(x_i)}e^{-1} - \sum_{i,j:i>j}\gamma_{ij}, \quad (8)$$

where the $\{b_i\},\{b_{ij}\}$ *are given by:*

$$b_{ij}(x_i,x_j) = \phi_{ij}(x_i,x_j)e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}}e^{-1}e^{-\gamma_{ij}},$$
$$b_i(x_i) = \psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_j \lambda_{ji}(x_i)}e^{-1}. \quad (9)$$

Proof. *We obtain* $G(\{\gamma_{ij}\},\{\lambda_{ij}\})$ *from the Bethe free energy* $F(\{b_{ij}\},\{b_i\},\{\gamma_{ij}\},\{\lambda_{ij}\})$ *by extremizing* F *and solving for* $\{b_i\},\{b_{ij}\}$ *as functions of* $\{\gamma_{ij},\lambda_{ij}\}$. *Then*

$$G(\{\gamma_{ij}\},\{\lambda_{ij}\}) = F(\{b_i(\{\gamma_{ij},\lambda_{ij}\})\},\{b_{ij}(\{\gamma_{ij},\lambda_{ij}\})\},\{\gamma_{ij}\},\{\lambda_{ij}\}). \quad (10)$$

In more detail, extremizing $F(\{b_{ij}\},\{b_i\},\{\gamma_{ij}\},\{\lambda_{ij}\})$ *with respect to* $\{b_{ij}\}$ *gives:*

$$1 + \log\frac{b_{ij}(x_i,x_j)}{\phi_{ij}(x_i,x_j)} + \lambda_{ij}(x_j) + \lambda_{ji}(x_i) + \gamma_{ij} = 0. \quad (11)$$

From this equation, we get two results. Firstly, we solve for $b_{ij}(x_i,x_j)$ *in terms of* $\{\lambda_{ij}\}$ *to get:*

$$b_{ij}(x_i,x_j) = \phi_{ij}(x_i,x_j)e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}}e^{-1}e^{-\gamma_{ij}}. \quad (12)$$

Secondly, we multiply equation (11) by $b_{ij}(x_i,x_j)$ *and sum over* i,j,x_i,x_j *to get:*

$$\sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j) + \sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j)\log\frac{b_{ij}(x_i,x_j)}{\phi_{ij}(x_i,x_j)} + \sum_{i,j:i>j}\sum_{x_i,x_j}\lambda_{ij}(x_j)b_{ij}(x_i,x_j) + \sum_{i,j:i>j}\sum_{x_i,x_j}\lambda_{ji}(x_i)b_{ij}(x_i,x_j) + \sum_{i,j:i>j}\sum_{x_i,x_j}\gamma_{ij}b_{ij}(x_i,x_j) = 0, \quad (13)$$

which helps simplify the form of $F(\{b_i(\{\gamma_{ij},\lambda_{ij}\})\},\{b_{ij}(\{\gamma_{ij},\lambda_{ij}\})\},\{\gamma_{ij}\},\{\lambda_{ij}\})$.

Similarly, we extremize $F(\{b_{ij}\},\{b_i\},\{\gamma_{ij}\},\{\lambda_{ij}\})$ *with respect to* $\{b_i\}$. *This gives analogous equations:*

$$-(n_i-1) \quad \{1+\log\tfrac{b_i(x_i)}{\psi_i(x_i)}\} - \textstyle\sum_j \lambda_{ji}(x_i) = 0,$$
$$-\sum_i \quad \textstyle\sum_{x_i}(n_i-1)b_i(x_i) - \sum_i(n_i-1) \quad \displaystyle\sum_{x_i} b_i(x_i)\log\frac{b_i(x_i)}{\psi_i(x_i)} - \sum_{i,j}\sum_{x_i}\lambda_{ji}(x_i)b_i(x_i) = 0, \quad (14)$$
$$b_i(x_i) \quad = \psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_j \lambda_{ji}(x_i)}e^{-1} \quad . \quad (15)$$

Using equations (13,14), we can write the Bethe energy as $-\sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j) + \sum_i(n_i-1)\sum_{x_i} b_i(x_i) - \sum_{i,j:i>j}\gamma_{ij}$. *Substituting for* $\{b_i\},\{b_{ij}\}$, *using equations (12,15), gives the result.*

We can rewrite the dual energy $G(\{\gamma_{ij}\},\{\lambda_{ij}\})$ as $f(\{\gamma_{ij}\},\{\lambda_{ij}\}) + g(\{\gamma_{ij}\},\{\lambda_{ij}\})$ where:

$$f(\{\gamma_{ij}\},\{\lambda_{ij}\}) = -\sum_{i,j:i>j}\sum_{x_i,x_j}\phi_{ij}(x_i,x_j)e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}}e^{-1}e^{-\gamma_{ij}} - \sum_{i,j:i>j}\gamma_{ij},$$
$$g(\{\gamma_{ij}\},\{\lambda_{ij}\}) = \sum_i(n_i-1)\sum_{x_i}\psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_j \lambda_{ji}(x_i)}e^{-1}. \quad (16)$$

We now show that the BP algorithm has simple dynamics in the dual energy space.

Theorem 2. *The BP algorithm corresponds to the update rule:*

$$\frac{\partial g}{\partial e^{-\lambda_{ji}(x_i)}}(\lambda_{ji}(x_i;t+1)) = -\frac{\partial f}{\partial e^{-\lambda_{ji}(x_i)}}(\lambda_{ji}(x_i;t)), \quad \frac{\partial g}{\partial \gamma_{ij}}(\gamma_{ij}(t+1)) = -\frac{\partial f}{\partial \gamma_{ij}}(\gamma_{ij}(t)). \tag{17}$$

provided we relate the $\{\lambda_{ij}\}$ *to the messages* $\{m_{ij}\}$ *by the equations:*

$$m_{ji}(x_i) = e^{\lambda_{ji}(x_i)} e^{-\frac{1}{n_i-1}\sum_k \lambda_{ki}(x_i)}, \text{ or the inverse } e^{-\lambda_{ij}(x_j)} = \prod_{k \neq i} m_{kj}(x_j). \tag{18}$$

Proof. *We calculate:*

$$\frac{\partial g}{\partial e^{-\lambda_{ji}(x_i)}} = -e^{\lambda_{ji}(x_i)} \frac{\partial g}{\partial \lambda_{ji}(x_i)} = e^{\lambda_{ji}(x_i)} \psi_i(x_i) e^{-\frac{1}{n_i-1}\sum_k \lambda_{ki}(x_i)} e^{-1},$$

$$\frac{\partial f}{\partial e^{-\lambda_{ji}(x_i)}} = -e^{\lambda_{ji}(x_i)} \frac{\partial f}{\partial \lambda_{ji}(x_i)} = -e^{\lambda_{ji}(x_i)} \sum_{x_j} \phi_{ij}(x_i,x_j) e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}} e^{-\gamma_{ij}} e^{-1}. \tag{19}$$

Combining equation (18) with equations (19,17) gives the BP update equations (2). Setting $\partial g/\partial \gamma_{ij}(\gamma_{ij}(t+1)) = -\partial f/\partial \gamma_{ij}(\gamma_{ij}(t)) = 0$ *gives the normalization (note that* g *is independent of* $\{\gamma_{ij}\}$*).*

Theorem 2 gives a new derivation of the BP algorithm where the messages $\{m_{ij}\}$ arise as Lagrange multipliers $\{\lambda_{ij}\}$ corresponding to the reformulation of the Bethe free energy into the dual energy $G(\{\gamma_{ij}\},\{\lambda_{ij}\})$.

From Theorem 2, we can immediately obtain Yedidia *et al*'s result as a corollary.

Corollary 1. *Fixed points of the BP algorithm correspond to extrema of the Bethe free energy and vice versa.*

Proof. *Fixed points of the BP algorithm occur when:*

$$\frac{\partial g}{\partial e^{-\lambda_{ji}(x_i)}} + \frac{\partial f}{\partial e^{-\lambda_{ji}(x_i)}} = 0, \quad \frac{\partial g}{\partial \gamma_{ij}} + \frac{\partial f}{\partial \gamma_{ij}} = 0, \tag{20}$$

which are precisely the conditions for extrema of the Bethe free energy.

This dual formulation of the Bethe free energy throws some light on the BP algorithm, see Theorem 2, and offers suggestions for how to create new algorithms. Unfortunately the minima of non-convex energy functions, such as the Bethe free energy, correspond to *saddle points* of the dual energy function [18] and it is usually hard to analyze global convergence to saddle points (analyzing convergence to a maximum or minimum is easier). (Note that if the Bethe free energy were convex then standard results in duality theory [18] would ensure that minima of the Bethe free energy correspond to maxima of the dual energy).

If the graph has no loops then it is straightforward to analyze the extrema of the dual energy. It can be shown that there is a unique solution which can be found by a recursive algorithm. By duality, this implies that there is a single extremum of the Bethe free energy which must be the global minimum (the Bethe free energy always has at least one extremum corresponding to the global minimum). It can be shown, by explicitly calculating the Hessian of the dual energy, that the solution is a saddle point in the dual space (Yuille – unpublished calculations). Analysis of the dual energy may also help understand when BP, and other algorithms, converge to the correct results, see [20].

In conclusion, it may be hard to tell whether the algorithm will converge or not based on its dynamics in the dual space (although it may be possible to obtain local convergence results). However, it is straightforward to use the dual energy to obtain many algorithms whose fixed points correspond to extrema of the Bethe free energy. It is also interesting to see how the "messages" arise from the Lagrange multipliers.

There is, however, something special about the BP algorithm. As will be shown later, in section (5), there are formal similarities between BP and the double-loop algorithm that we present in this paper. Studying these similarities might help determine when BP does and does not converge.

3 Convexity and Concavity

We now develop an algorithm that is guaranteed to converge to a minimum of the Bethe free energy. This section describes how we can exploit the fact that the Bethe free energy is the sum of a convex and a concave term, see section (2).

Our main results are given by Theorem's 3,4,5 and show that we can obtain discrete iterative algorithms to minimize energy functions which are the sum of a convex and a concave term. (A similar result can be obtained using the Legendre transform – Rangarajan personal communication). We first consider the case where there are no constraints for the optimization. Then we generalize to the case where linear constraints are present.

Theorem 3. *Consider an energy function $E(\vec{z})$ (bounded below) of form $E(\vec{z}) = E_{vex}(\vec{z}) + E_{cave}(\vec{z})$ where $E_{vex}(\vec{z}), E_{cave}(\vec{z})$ are convex and concave functions of $\vec{z}$ respectively. Then the discrete iterative algorithm $\vec{z}^t \mapsto \vec{z}^{t+1}$ given by:*

$$\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t), \tag{21}$$

is guaranteed to monotonically decrease the energy $E(\vec{)}$ as a function of time and hence to converge to a minimum of $E(\vec{z})$.

Proof. *The convexity and concavity of $E_{vex}(.)$ and $E_{cave}(.)$ means that:*

$$\begin{aligned} E_{vex}(\vec{z}_2) &\geq E_{vex}(\vec{z}_1) + (\vec{z}_2 - \vec{z}_1) \cdot \vec{\nabla} E_{vex}(\vec{z}_1) \\ E_{cave}(\vec{z}_4) &\leq E_{cave}(\vec{z}_3) + (\vec{z}_4 - \vec{z}_3) \cdot \vec{\nabla} E_{cave}(\vec{z}_3), \end{aligned} \tag{22}$$

for all $\vec{z}_1, \vec{z}_2, \vec{z}_3, \vec{z}_4$. Now set $\vec{z}_1 = \vec{z}^{t+1}, \vec{z}_2 = \vec{z}^t, \vec{z}_3 = \vec{z}^t, \vec{z}_4 = \vec{z}^{t+1}$. Using equation (22) and the algorithm definition (i.e. $\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t)$) we find that:

$$E_{vex}(\vec{z}^{t+1}) + E_{cave}(\vec{z}^{t+1}) \leq E_{vex}(\vec{z}^t) + E_{cave}(\vec{z}^t), \tag{23}$$

which proves the claim.

Theorem 3 generalizes previous results by Marcus, Waugh and Westervelt [9],[19] on the convergence of discrete iterated neural networks. (They discussed the case where one function was convex and the second function was linear).

We can extend this result to allow for linear constraints on the variables $\vec{z}$. This can be given a geometrical intuition. Firstly, properties such as concavity and concaveness are preserved when linear constraints are imposed. Secondly, because the constraints are linear they determine a hyperplane on which the constraints are satisfied. Theorem 3 can then be applied to the variables on this hyperplane.

Theorem 4. *Consider a function $E(\vec{z}) = E_{vex}(\vec{z}) + E_{cave}(\vec{z})$ subject to k linear constraints $\vec{\phi}^\mu \cdot \vec{z} = c^\mu$ where $\{c^\mu : \mu = 1, ..., k\}$ are constants. Then the algorithm $\vec{z}^t \mapsto \vec{z}^{t+1}$ given by:*

$$\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t) - \sum_{\mu=1}^{k} \alpha^\mu \vec{\phi}^\mu, \tag{24}$$

where the parameters $\{\alpha^\mu\}$ are chosen to ensure that $\vec{z}^{t+1} \cdot \vec{\phi}^\mu = c^\mu$ for $\mu = 1, ..., k$, is guaranteed to monotonically decrease the energy $E(\vec{z}^t)$ and hence to converge to a minimum of $E(\vec{z})$.

Proof. *Intuitively the update rule has to balance the gradients of E_{vex}, E_{cave} in the unconstrained directions of $\vec{z}$ and the term $\sum_{\mu=1}^{k} \alpha^\mu \vec{\phi}^\mu$ is required to deal with the differences in the directions of the constraints. More formally, we define orthogonal unit vectors $\{\vec{\psi}^\nu : \nu = 1, ..., n-k\}$ which span the space orthogonal to the constraints $\{\vec{\phi}^\mu : \mu = 1, ..., k\}$. Let $\vec{y}(\vec{z}) = \sum_{\nu=1}^{n-k} \vec{\psi}^\nu (\vec{z} \cdot \vec{\psi}^\nu)$ be the projection of $\vec{z}$ onto this space. Define functions $\hat{E}_{cave}(\vec{y}), \hat{E}_{vex}(\vec{y})$ by:*

$$\hat{E}_{cave}(\vec{y}(\vec{z})) = E_{cave}(\vec{z}), \quad \hat{E}_{vex}(\vec{y}(\vec{z})) = E_{vex}(\vec{z}). \tag{25}$$

Then we can use the algorithm of Theorem 3 on the unconstrained variables $\vec{y} = (y_1, ..., y_{n-k})$. By definition of $\vec{y}(\vec{z})$ we have $\partial \vec{z} / \partial y_\nu = \vec{\psi}^\nu$. Therefore the algorithm reduces to:

$$\vec{\psi}^\nu \cdot \vec{\nabla}_{\vec{z}} E_{vex}(\vec{z}^{t+1}) = -\vec{\psi}^\nu \cdot \vec{\nabla}_{\vec{z}} E_{cave}(\vec{z}^t), \quad \nu = 1, ..., n-k. \tag{26}$$

This gives the result (recalling that $\vec{\phi}^\mu \cdot \vec{\psi}^\nu = 0$ for all μ, ν).

It follows from Theorem 4 that we only need to impose the constraints on $E_{vex}(\vec{z}^{t+1})$ and not on $E_{cave}(\vec{z}^{t})$. In other words, we set

$$\bar{E}_{vex}(\vec{z}^{t+1}) = E_{vex}(\vec{z}^{t+1}) + \sum_{\mu} a^{\mu} \vec{\phi}^{\mu} \cdot \vec{z}^{t+1}, \tag{27}$$

and use update equations:

$$\frac{\partial \bar{E}_{vex}}{\partial \vec{z}}(\vec{z}^{t+1}) = -\frac{\partial E_{cave}}{\partial \vec{z}}(\vec{z}^{t}), \tag{28}$$

where the coefficients $\{\alpha^{\mu}\}$ must be chosen to ensure that the constraints $\vec{\phi}^{\mu} \cdot \vec{z}^{t+1} = c^{\mu}\ \forall \mu$ are satisfied.

We now restrict ourselves to the specific case where $E_{vex}(\vec{z}) = \sum_i z_i \log \frac{z_i}{\zeta_i}$. (This form of $E_{vex}(\vec{z})$ will arise when we apply Theorem 4 to the Bethe free energy, see section (4)). We let $\vec{h}(\vec{z}) = \vec{\nabla} E_{cave}(\vec{z})$. Then the update rules of Theorem 4, see equation (24), *can be expressed as selecting* $\vec{z}^{t+1}$ *to minimize a convex cost function* $E^{t+1}(\vec{z}^{t+1})$ which, by duality, implies that *the constraint coefficients* $\{\alpha^{\mu}\}$ *can be found by maximizing a dual energy function.* More formally, we have the following result:

Theorem 5. *Let* $E_{vex}(\vec{z}) = \sum_i z_i \log \frac{z_i}{\zeta_i}$, *then the update equation of Theorem 4 can be expressed as minimizing the convex energy function:*

$$E^{t+1}(\vec{z}^{t+1}) = \vec{z}^{t+1} \cdot \vec{h} + \sum_i z_i^{t+1} \log \frac{z_i^{t+1}}{\zeta_i} + \sum_{\mu} \alpha^{\mu} \{\vec{\phi}^{\mu} \cdot \vec{z}^{t+1} - c^{\mu}\}, \tag{29}$$

where $\vec{h} = \vec{\nabla} E_{cave}(\vec{z}^{t})$. *By duality the solution corresponds to*

$$z_i^{t+1} = \zeta_i e^{-h_i} e^{-1} e^{-\sum_{\mu} \alpha^{\mu} \phi_i^{\mu}}, \tag{30}$$

where the Lagrange multipliers $\{\alpha^{\mu}\}$ *are constrained to maximize the (concave) dual energy:*

$$\hat{E}^{t+1}(\alpha) = -\sum_i \zeta_i e^{-h_i} e^{-1} e^{-\sum_{\nu} \phi_i^{\nu} \alpha^{\nu}} - \sum_{\mu} \alpha^{\mu} c^{\mu}. \tag{31}$$

.

Moreover, maximizing $\hat{E}^{t+1}(\alpha)$ *with respect to a specific* α^{μ} *enables us to satisfy the corresponding constraint exactly.*

Proof. *This is given by straightforward calculations. Differentiating* E^{t+1} *with respect to* z_i^{t+1} *gives:*

$$1 + \log \frac{z_i}{\zeta_i} = -h_i - \sum_{\mu} \alpha^{\mu} \phi_i^{\mu}, \tag{32}$$

which corresponds to the update equation (24) of Theorem 4. Since $E^{t+1}(\vec{z})$ *is convex, duality ensures that the dual* $\hat{E}^{t+1}(\alpha)$ *is concave [18], and hence has a unique maximum which corresponds to the constraints being solved. Setting* $\frac{\partial \hat{E}^{t+1}}{\partial \alpha^{\mu}} = 0$ *ensures that* $\vec{z}^{t+1} \cdot \vec{\phi}^{\mu} = c^{\mu}$, *and hence satisfies the* μ^{th} *constraint.*

By using Theorem 5, we see that solving for the constraint coefficients in Theorem 4 can be reduced to maximizing a concave energy function in the $\{\alpha^{\nu}\}$.

Moreover, for the specific constraints used in the Bethe free energy we will be able to determine an efficient discrete iterative algorithms for solving for the constraints. This is by generalizing work by Kosowsky and Yuille [7],[8] who used a result similar to Theorem 5 to obtain an algorithm for solving the linear assignment problem. (Kosowsky and Yuille [7],[8] also showed that this result could be used to derive an energy function for the classic Sinkhorn algorithm [17] which converts positive matrices into doubly stochastic ones.) Rangarajan *et al* [13] applied this result to obtain double loop algorithms for a range of optimization problems subject to linear constraints.

In the next section we apply Theorems 4 and 5 to the Bethe free energy. In particular, we will show that the nature of the linear constraints for the Bethe free energy mean that solving for the constraints in Theorem 5 can be done efficiently.

4 Constraints for the Bethe Free energy

In this section we return to the Bethe free energy and describe how we can implement an algorithm of the form given by Theorems 3,4 and 5. This *double-loop* algorithm is designed by splitting the Bethe free energy into convex and concave parts. An inner loop is used to impose the linear constraints. (This design was influenced by the work of Rangarajan *et al* [13]).

First we split the Bethe free energy in two parts:

$$E_{vex} = \sum_{i,j:i>j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log \frac{b_{ij}(x_i,x_j)}{\phi_{ij}(x_i,x_j)} + \sum_i \sum_{x_i} b_i(x_i) \log \frac{b_i(x_i)}{\psi_i(x_i)},$$
$$E_{cave} = -\sum_i n_i \sum_{x_i} b_i(x_i) \log \frac{b_i(x_i)}{\psi_i(x_i)}. \quad (33)$$

This split enables us to get non-zero derivatives of E_{vex} with respect to both $\{b_{ij}\}$ and $\{b_i\}$. (Other choices of split are possible).

We now need to express the linear constraints, see equation (6), in the form used in Theorems 4 and 5. To do this, we set $\vec{z} = (b_{ij}(x_i,x_j), b_i(x_i))$ so that the first $\frac{N(N-1)}{2}M^2$ components of $\vec{z}$ correspond to the $\{b_{ij}\}$ and the later NM components to the $\{b_i\}$ (recall that there are N nodes of the graph each with M states). The dot product of $\vec{z}$ with a vector $\vec{\phi} = (T_{ij}(x_i,x_j), U_i(x_i))$ is given by $\sum_{i,j:i>j}\sum_{x_i,x_j} b_{ij}(x_i,x_j)T_{ij}(x_i,x_j) + \sum_i \sum_{x_i} b_i(x_i)U_i(x_i)$.

There are two types of constraints: (i) the normalization constraints $\sum_{x_p,x_q} b_{pq}(x_p,x_q) = 1 \ \forall \ p,q : p > q$, and (ii) the consistency constraints $\sum_{x_p} b_{pq}(x_p,x_q) = b_q(x_q) \ \forall p,q,x_q : p > q$ and $\sum_{x_q} b_{pq}(x_p,x_q) = b_p(x_p) \ \forall q,p,x_p : p > q$.

We index the normalization constraints by pq (with $p > q$). Then we can express the constraint vectors $\{\vec{\phi}^{pq}\}$, the constraint coefficients $\{\alpha^{pq}\}$, and the constraint values $\{c^{pq}\}$ by:

$$\vec{\phi}^{pq} = (\delta_{pi}\delta_{qj}, 0), \quad \alpha^{pq} = \gamma_{pq}, \quad c^{pq} = 1, \ \forall p,q. \quad (34)$$

The consistency constraints are indexed by pqx_q (with $p > q$) and qpx_p (with $p > q$). The constraint vectors, the constraint coefficients, and the constraint values are given by:

$$\vec{\phi}^{pqx_q} = (\delta_{ip}\delta_{jq}\delta_{x_j,x_q}, -\delta_{iq}\delta_{x_i,x_q}), \ \alpha^{pqx_q} = \lambda_{pq}(x_q), \ c^{pqx_q} = 0, \ \forall p,q,x_q$$
$$\vec{\phi}^{qpx_p} = (\delta_{ip}\delta_{jq}\delta_{x_i x_p}, -\delta_{ip}\delta_{x_i,x_p}), \ \alpha^{qpx_p} = \lambda_{qp}(x_p), \ c^{qpx_p} = 0, \ \forall q,p,x_p. \quad (35)$$

We now apply Theorem 4 to the Bethe free energy and obtain:

Theorem 6. *The following update rule is guaranteed to reduce the Bethe free energy provided the constraint coefficients $\{\gamma_{pq}\}, \{\lambda_{pq}\}, \{\lambda_{qp}\}$ can be chosen to ensure that $\{b_{ij}(t+1)\}, \{b_i(t+1)\}$ satisfy the linear constraints of equation (6):*

$$b_{ij}(x_i,x_j;t+1) = \phi_{ij}(x_i,x_j)e^{-1}e^{-\lambda_{ij}(x_j)}e^{-\lambda_{ji}(x_i)}e^{-\gamma_{ij}},$$
$$b_i(x_i;t+1) = \psi_i(x_i)e^{-1}e^{n_i}\left(\frac{b_i(x_i;t)}{\psi_i(x_i)}\right)^{n_i} e^{\sum_k \lambda_{ki}(x_i)}. \quad (36)$$

Proof. *We substitute equation (33) into equation (24) and use equations (34,35) for the constraints. The constraint terms $\sum_\mu \alpha^\mu \vec{\phi}^\mu$ simplify owing to the form of the constraints (e.g. $\sum_{p,q:p>q} \gamma_{pq}\delta_{ip}\delta_{jq} = \gamma_{ij}$).*

Finally, we use Theorem 5 to obtain an algorithm which ensures that the constraints are satisfied. Recall that finding the constraint coefficients is equivalent to maximizing the dual energy $\hat{E}^{t+1}(\alpha)$ and that performing the maximization with respect to the μ^{th} coefficient α^μ corresponds to solving the μ^{th} constraint equation. For the Bethe free energy it is possible to solve the μ^{th} constraint equation to *obtain an analytic expression for the μ^{th} coefficient* in terms of the remaining coefficients. Therefore we can maximize the dual energy $\hat{E}^{t+1}(\alpha)$ with respect to any coefficient α^μ analytically. Hence we have an algorithm that is guaranteed to converge to the maximum of $\hat{E}^{t+1}(\alpha)$: *select a constraint μ, solve for the equation $\frac{\partial \hat{E}^{t+1}}{\partial \alpha^\mu}$ for α^μ analytically, and repeat.*

More formally:

Theorem 7. *The constraint coefficients $\{\gamma_{pq}\}, \{\lambda_{pq}\}, \{\lambda_{qp}\}$ of Theorem 6 can be solved for by a discrete iterative algorithm, indexed by τ, guaranteed to converge to the unique solution. At each step we select coefficients $\gamma_{pq}, \lambda_{pq}(x_q)$ or $\lambda_{qp}(x_p)$ and update them by:*

$$
\begin{aligned}
e^{\gamma_{pq}(\tau+1)} &= \sum_{x_p,x_q} \phi_{pq}(x_p,x_q)e^{-1}e^{-\lambda_{pq}(x_q,\tau)}e^{-\lambda_{qp}(x_p;\tau)}, \\
e^{2\lambda_{pq}(x_q;\tau+1)} &= \frac{\sum_{x_p} \phi_{pq}(x_p,x_q)e^{-\lambda_{qp}(x_p;\tau)}e^{-\gamma_{pq}(\tau)}}{\psi_q(x_q)e^{n_q}(\frac{b_q(x_q;t)}{\psi_q(x_q;t)})^{n_q}e^{\sum_{j\neq p}\lambda_{jq}(x_q;\tau)}}, \\
e^{2\lambda_{qp}(x_p;\tau+1)} &= \frac{\sum_{x_q} \phi_{pq}(x_p,x_q)e^{-\lambda_{pq}(x_q;\tau)}e^{-\gamma_{pq}(\tau)}}{\psi_p(x_p)e^{n_p}(\frac{b_p(x_p;t)}{\psi_p(x_p;t)})^{n_p}e^{\sum_{j\neq q}\lambda_{jp}(x_p;\tau)}}
\end{aligned}
\tag{37}
$$

Proof. *We use the update rule given by Theorem 6 and calculate the constraint equations, $\vec{z}\cdot\vec{\phi}^{\mu} = c^{\mu}$, $\forall\ \mu$. For the Bethe free energy we obtain equations (37) where the upper equation corresponds to the normalization constraints and the lower equations to the consistency constraints. We observe that we can solve each equation analytically for the corresponding constraint coefficients $\gamma_{pq}, \lambda_{pq}(x_q), \lambda_{qp}(x_p)$. By Theorem 5, this is equivalent to maximizing the dual energy, see equation (31), with respect to each coefficient. Since the dual energy is concave solving for each coefficient $\gamma_{pq}, \lambda_{pq}(x_q), \lambda_{qp}(x_p)$ (with the others fixed) is guaranteed to increase the dual energy. Hence we can maximize the dual energy, and hence ensure the constraints are satisfied, by repeatedly selecting coefficients and solving the equations (37).*

Observe that we can update all the coefficients $\{\gamma_{pq}\}$ simultaneously because their update rule (i.e. the right hand side of the top equation of 37) depends only on the $\{\lambda_{pq}\}$. Similarly, we can update many of the $\{\lambda_{pq}\}$ simultaneously because their update rules (i.e. the right hand sides of the middle and bottom equation of 37) only depend on a subset of the $\{\lambda_{pq}\}$. (For example, when updating $\lambda_{pq}(x_q)$ we can simultaneously update any $\lambda_{ij}(x_j)$ provided $i \neq p \neq j$ and $i \neq q \neq j$.)

5 The Double-Loop Algorithm and BP

In this section we summarize the double-loop algorithm and then show that it has some interesting similarities to the belief propagation algorithm as formulated by Yedidia *et al* [22].

The double-loop algorithm consists of an outer loop which implements equation (36) plus an inner loop, given by equations (37) which imposes the constraints. The double loop design of the algorithm was influenced by the work of Rangarajan *et al* [13].

A convergence proof and theoretical analysis of Rangarajan *et al*'s algorithm was given in [14]. It is not clear how many iterations of the internal loop are needed to ensure the constraints are satisfied (though techniques developed by [7],[8] and [14] might be used to bound the number of iterations). In practice, the convergence of these inner loops seems to go quickly [7],[8] [13].

The algorithm consists of an inner and an outer loop. The outer loop has (discrete) time parameter t and is given by:

$$b_{ij}(x_i,x_j;t+1) = \phi_{ij}(x_i,x_j)e^{-1}e^{-\lambda_{ij}(x_j)}e^{-\lambda_{ji}(x_i)}e^{-\gamma_{ij}}, \tag{38}$$

$$b_i(x_i;t+1) = \psi_i(x_i)e^{-1}e^{n_i}(\frac{b_i(x_i;t)}{\psi_i(x_i)})^{n_i}e^{\sum_k \lambda_{ki}(x_i)}, \tag{39}$$

and the inner loop to determine the $\{\gamma_{pq}\}, \{\lambda_{pq}\}, \{\lambda_{qp}\}$ has time parameter τ and is given by:

$$e^{\gamma_{pq}(\tau+1)} = \sum_{x_p,x_q} \phi_{pq}(x_p,x_q) e^{-1} e^{-\lambda_{pq}(x_q;\tau)} e^{-\lambda_{qp}(x_p;\tau)},$$
$$e^{2\lambda_{pq}(x_q;\tau+1)} = \frac{\sum_{x_p} \phi_{pq}(x_p,x_q) e^{-\lambda_{qp}(x_p;\tau)} e^{-\gamma_{pq}(\tau)}}{\psi_q(x_q) e^{n_q} (\frac{b_q(x_q;t)}{\psi_q(x_q;t)})^{n_q} e^{\sum_{j\neq p} \lambda_{jq}(x_q;\tau)}}$$
$$e^{2\lambda_{qp}(x_p;\tau+1)} = \frac{\sum_{x_q} \phi_{pq}(x_p,x_q) e^{-\lambda_{pq}(x_q;\tau)} e^{-\gamma_{pq}(\tau)}}{\psi_p(x_p) e^{n_p} (\frac{b_p(x_p;t)}{\psi_p(x_p;t)})^{n_p} e^{\sum_{j\neq q} \lambda_{jp}(x_p;\tau)}}. \quad (40)$$

We now show that there are formal similarities between this algorithm and belief propagation as specified by equations (2,3, 4).

Recall that we can relate the messages $\{m_{ij}\}$ of BP to lagrange multipliers $\{\lambda_{ij}\}$ by equation (18) of Theorem 2. The relations are $m_{ji}(x_i) = e^{\lambda_{ji}(x_i)} e^{-\frac{1}{n_i-1}\sum_k \lambda_{ki}(x_i)}$ and the inverse $e^{-\lambda_{ij}(x_j)} = \prod_{k\neq i} m_{kj}(x_j)$ $\forall i, j, x_i, x_j$.

Using these relations, and comparing equations (4,38), we see that the update equations for the $\{b_{ij}\}$ are identical for the double-loop algorithm and for BP. (In fact, the $\{b_{ij}\}$ variables play no role in the dynamics of either algorithm and need not be evaluated at all until the algorithm has converged).

The main difference between the two algorithms is that the double-loop algorithm has an outer loop to update the $\{b_i\}$ (equation (39)) and an inner loop (equation (40)) to update the $\{\lambda_{ij}, \gamma_{ij}\}$. By contrast, in the BP algorithm all the dynamics (equation (2)) can be expressed in terms of the messages $\{m_{ij}\}$, or the lagrange parameters $\{\lambda_{ij}\}$ (which directly determine the $\{b_i\}$ by equation (3)).

To go further, we express the BP update rules, equations (2,3), in terms of the lagrange multipliers. This gives:

$$e^{\lambda_{ij}(x_j;t+1)} \quad e^{-\frac{1}{n_j-1}\sum_k \lambda_{kj}(x_j;t+1)} = c \sum_{x_i} \psi_{ij}(x_i,x_j)\psi_i(x_i) e^{-\lambda_{ji}(x_i;t)}, \quad (41)$$
$$b_i(x_i;t) \quad = \hat{c}\psi_i(x_i) e^{-\frac{1}{n_i-1}\sum_l \lambda_{li}(x_i;t)}, \quad (42)$$

which we can compare to the equations for the double-loop algorithm:

$$e^{2\lambda_{pq}(x_q;\tau+1)} = \frac{\sum_{x_p} \psi_{pq}(x_p,x_q)\psi_p(x_p) e^{-\lambda_{qp}(x_p;\tau)} e^{-\gamma_{pq}}}{e^{n_q}\{\frac{b_q(x_q)}{\psi_q(x_q)}\}^{n_q} e^{\sum_{j\neq p}\lambda_{jq}(x_q;\tau)}}, \quad (43)$$
$$b_i(x_i;t+1) = \psi_i(x_i) e^{-1} e^{n_i} \{\frac{b_i(x_i;t)}{\psi_i(x_i)}\}^{n_i} e^{\sum_k \lambda_{ki}(x_i)}, \quad (44)$$

where (as before) we use the variables τ and t for the time steps in the inner and outer loops respectively. We also use the relationship $\phi_{pq}(x_p,x_q) = \psi_{pq}(x_p,x_q)\psi_p(x_p)\psi_q(x_q)$. (We ignore the "normalization dynamics" – updating the $\{\gamma_{pq}\}$ – because it is straightforward and can be taken for granted. With this understanding we have dropped the dependence of the $\{\gamma_{pq}\}$ on τ).

Now suppose we try to approximate the double-loop algorithm by setting $b_i(x_i;t) = \hat{c}\psi_i(x_i) e^{-\frac{1}{n_i-1}\sum_l \lambda_{li}(x_i;t)}$ from equation (42). This "collapses" the outer loop of the double-loop algorithm and so only the inner loop remains. This reduces equation (43) to:

$$e^{2\lambda_{pq}(x_q;\tau+1)} \times \{e^{-\lambda_{pq}(x_q;\tau)} e^{-\frac{1}{n_q-1}\sum_j \lambda_{jq}(x_q;\tau)}\} = \sum_{x_p} \psi_{pq}(x_p,x_q)\psi_p(x_p) e^{-\lambda_{qp}(x_p;\tau)} e^{-\gamma_{pq}}, \quad (45)$$

which is almost exactly the same as the BP algorithm, equation (41). (This can be verified using the identity $\{e^{-\frac{1}{n_q-1}\sum_l \lambda_{lq}(x_q)}\}^{n_q} e^{\sum_{j\neq p}\lambda_{jq}(x_q)} = e^{-\frac{1}{n_q-1}\sum_j \lambda_{jq}(x_q)} e^{-\lambda_{pq}(x_q)}$). The only difference is the factor $\{e^{-\lambda_{pq}(x_p;\tau)} e^{-\frac{1}{n_q-1}\sum_j \lambda_{jq}(x_q;\tau)}\}$ is evaluated at time τ from equation (45) and at time $t+1$ for equation (41).

It is also straightforward to check that equation (44) for updating the outer loop has $b_i(x_i) = \hat{c}\psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_l \lambda_{li}(x_i)}$ as a fixed point. More precisely, if we converge to a state where $\lambda_{ij}(x_j;\tau+1) = \lambda_{ij}(x_j;\tau)\ \forall\ i,j,x_j$ then substituting $b_i(x_i;t) = \hat{c}\psi_i(x_i)e^{-\frac{1}{n_i-1}\sum_l \lambda_{li}(x_i;t)}$ into equation (44) gives $b_i(x_i;t+1) = b_i(x_i;t)\ \forall\ i, x_i$.

To conclude, the BP algorithm can be thought of as an approximation to the double-loop algorithm. The approximation is done by assuming a fixed functional form for the $\{b_i\}$ in terms of the $\{\lambda_{ij}\}$, thereby collapsing the outer loop of the algorithm, and modifying the update equations for the inner loop (by evaluating certain terms at $\tau+1$ instead of at τ.

6 The Kikuchi Approximation

In this section we show that the results we obtained for the Bethe free energy can be extended to the more general Kikuchi approximation [3]. (Recall that Yedidia *et al* [22] derived a "generalized belief propagation algorithm" whose fixed points correspond to extrema of the Kikuchi free energy.)

Firstly, we generalize the results of Section (2) and show that we can get a dual formulation of the Kikuchi approximation. This enables us to generate a class of algorithms whose fixed points correspond to the extrema of the Kikuchi free energy. Secondly, we generalize the results of Section (3, 4) to obtain a double-loop algorithm which is guaranteed to converge to a minimum of the Kikuchi free energy.

We now define the Kikuchi free energy (following the exposition in [22]). For a general graph, let R be a set of regions that include some basic clusters of nodes, their intersections, the intersections of the intersections, and so on. The Bethe approximation corresponds to the special case where the basic clusters consist of all linked pairs of nodes.

For any region r, we define the *super-regions of* r to be the set $sup(r)$ of all regions in R which contain r. Similarly, we define the *sub-regions of* r to be the set $sub(r)$ of all regions in R which lie completely within r. In addition, we define the *direct sub-regions of* r to be the set $sub_d(r)$ of all sub-regions of r which have no super-regions which are also sub-regions of r. Similarly, define the *direct super-regions of* r to be the set $sup_d(r)$ of super-regions of r which have no sub-regions which are super-regions of r. If s is a direct sub-region of r, then we define $r \setminus s$ to be the those nodes which are in r but not in s.

Let x_r be the state of the nodes in region r and $b_r(x_r)$ be the "belief" in x_r. $x \in r \setminus s$ denotes the state of the nodes which are in r but not in s. Any region r has an energy $E_r(x_r)$ associated with it (e.g. for a region with pairwise interactions we have $E_r(x_r) = -\log\prod_{i\in r,\ j\in r: i>j}\psi_{ij}(x_i,x_j) - \log\prod_{i\in r}\psi_i(x_i))$. Then the Kikuchi free energy is:

$$F_K = \sum_{r\in R} c_r\{\sum_{x_r} b_r(x_r)E_r(x_r) + \sum_{x_r} b_r(x_r)\log b_r(x_r)\} + L_K, \tag{46}$$

where L_K gives the constraints (see equation (47) below), c_r is an "over-counting" number of region r, defined by $c_r = 1 - \sum_{s\in sup(r)} c_s$ where $sup(r)$ is the set of all regions in R which contain r. For the largest regions, we have $c_r = 1$.

The beliefs $b_r(x_r)$ must obey two types of constraints: (i) they must all sum to one, (ii) they must be consistent with the beliefs in regions which intersect with r. This consistency can be ensured by imposing consistency between all regions with their direct sub-regions (i.e. between all $r \in R$ with their direct sub-regions $s \in sub_d(r)$). We can impose the constraints by lagrange multipliers:

$$L_K = \sum_{r\in R}\gamma_r(\sum_{x_r} b_r(x_r) - 1) + \sum_{r\in R}\sum_{s\in sub_d(r)}\sum_{x_s}\lambda_{r,s}(x_s)\{\sum_{x\in r\setminus s} b_r(x_r) - b_s(x_s)\}. \tag{47}$$

6.1 A Dual Formulation of the Kikuchi Free Energy

We now derive a dual energy for the Kikuchi free energy. This is the direct extension of our results in subsection (2.2) for the Bethe free energy. From this dual energy we can derive algorithms whose fixed points are extrema of the Kikuchi free energy. These algorithms are expressed in terms of updating values of the Lagrange multipliers $\{\gamma_r\}, \{\lambda_{r,s}\}$. The algorithms could be expressed in terms of new "message" variables defined as functions of the $\{\lambda_{r,s}\}$ but there is no obvious advantage to introducing new variables.

Theorem 8. *The extrema of the Kikuchi Free Energy, see equation (46), correspond to extrema of the Kikuchi dual energy $G(\{\gamma_r\},\{\lambda_{r,s}\})$ given by:*

$$G_K(\{\gamma_r\},\{\lambda_{r,s}\}) = -\sum_{r\in R} c_r \sum_{x_r} e^{-1} e^{-E_r(x_r)} e^{-\gamma_r/c_r} e^{-(1/c_r)\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)} e^{(1/c_r)\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)} - \sum_{r\in R}\gamma_r. \tag{48}$$

Proof. *We obtain $G_K(\{\gamma_r\},\{\lambda_{r,s}\})$ from the Kikuchi free energy $F_K(\{b_r\}$ by extremizing F_K and solving for $\{b_r\}$ as a function of $\{\gamma_r,\lambda_{r,s}\}$. Then we define $G_K(\{\gamma_r\},\{\lambda_{r,s}\})$ to be $F_K(\{b_r(\{\gamma_r,\lambda_{r,s}\}))$. In more detail, extremizing the Kikuchi Free Energy, see equation (46), with respect to $b_r(x_r)$ gives:*

$$c_r E_r(x_r) + c_r + c_r\log b_r(x_r) + \gamma_r + \sum_{s\in sub_d(r)}\lambda_{r,s}(x_s) - \sum_{v\in sup_d(r)}\lambda_{v,r}(x_r) = 0. \tag{49}$$

This yields:

$$b_r(x_r) = e^{-1} e^{-E_r(x_r)} e^{-\gamma_r/c_r} e^{-(1/c_r)\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)} e^{(1/c_r)\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{50}$$

In addition, by multiplying equation (49) by $b_r(x_r)$ and summing over r, x_r we obtain:

$$\begin{aligned}\sum_{r\in R}\sum_{x_r} c_r b_r(x_r)E_r(x_r) + \sum_{r\in R}\sum_{x_r} c_r b_r(x_r) + \sum_{r\in R}\sum_{x_r} c_r b_r(x_r)\log b_r(x_r) + \sum_{r\in R}\sum_{x_r}\gamma_r b_r(x_r)\\ +\sum_{r\in R}\sum_{x_r}\{\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)b_r(x_r) - \sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)b_r(x_r)\} = 0.\end{aligned} \tag{51}$$

We can then calculate $F_K(\{b_r(\{\gamma_r,\lambda_{r,s}\}))$ using equations (50,51). Using the identities:

$$\begin{aligned}&\textstyle\sum_{r\in R}\sum_{s\in sub_d(r)}\sum_{x_s}\lambda_{r,s}(x_s)\{\sum_{x\in r\backslash s} b_r(x_r) - b_s(x_s)\}\\ =\;& \sum_{r\in R}\sum_{s\in sub_d(r)}\sum_{x_r}\lambda_{r,s}(x_s)b_r(x_r) - \sum_{r\in R}\sum_{s\in sub_d(r)}\sum_{x_s}\lambda_{r,s}(x_s)b_s(x_s)\\ =\;& \sum_{r\in R}\sum_{x_r}\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)b_r(x_r) - \sum_{r\in R}\sum_{x_r}\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)b_r(x_r),\end{aligned} \tag{52}$$

we obtain the result.

As for the Bethe dual energy we can obtain a variety of algorithms whose fixed points correspond to extrema of the Kikuchi free energy. For any functions $f_K(\{\lambda_{r,s}\},\{\gamma_r\}), g_K(\{\lambda_{r,s}\},\{\gamma_r\})$ such that $f_K+g_K = G_K$, we can derive algorithms such as:

$$\begin{aligned}h(\lambda_{r,s},\gamma_r;t+1)\frac{\partial f_K}{\partial\lambda_{r,s}}(t+1) &= -h(\lambda_{r,s},\gamma_r;t)\frac{\partial g_K}{\partial\lambda_{r,s}}(t),\\ l(\lambda_{r,s},\gamma_r;t+1)\frac{\partial f_K}{\partial\gamma_r}(t+1) &= -l(\lambda_{r,s},\gamma_r;t)\frac{\partial g_K}{\partial\gamma_r}(t),\end{aligned} \tag{53}$$

where $l(\lambda_{r,s},\gamma_r;t)$ and $h(\lambda_{r,s},\gamma_r;t)$ are any functions of $\{\lambda_{r,s}\},\{\gamma_r\}$.

The fixed points of these algorithms will be extrema of the Kikuchi free energy. It is plausible that Yedidia *et al*'s generalized belief propagation algorithm [22] corresponds to one such algorithm.

6.2 A Double-Loop Algorithm to Minimize the Kikuchi Free Energy

Now we obtain a double-loop algorithm guaranteed to minimize the Kikuchi free energy. This is found by generalizing our results of Sections (3, 4).

As we derive the algorithm we will restrict ourselves to the case where $c_r \leq 1,\ \forall\ r\in R$. Then we will show how to generalize the algorithm to cases which do not satisfy this constraint.

First we observe that the Kikuchi free energy, like the Bethe free energy, can be split into a concave and a convex part. We choose the following split (many others are possible):

$$F_K = F_{K,vex} + F_{K,cave}, \tag{54}$$

where

$$F_{K,vex} = \sum_{r\in R}\{\sum_{x_r} b_r(x_r)E_r(x_r) + \sum_{x_r} b_r(x_r)\log b_r(x_r)\} + \sum_{r\in R}\gamma_r(\sum_{x_r} b_r(x_r) - 1) \tag{55}$$

$$+\sum_{r\in R}\sum_{s\in sub_d(r)}\sum_{x_s}\lambda_{r,s}(x_s)\{\sum_{x\in r\backslash s} b_r(x_r) - b_s(x_s)\} \tag{56}$$

$$F_{K,cave} = \sum_{r\in R}(c_r - 1)\{\sum_{x_r} b_r(x_r)E_r(x_r) + \sum_{x_r} b_r(x_r)\log b_r(x_r)\}, \tag{57}$$

where we have used the results stated in equations (27,28) to impose the constraints only on the convex term. It can be readily verified that $F_{K,vex}$ is a convex function of $\{b_r(x_r)\}$ and $F_{K,cave}$ is a concave function. Note the constraint $c_r \leq 1,\ \forall\ r \in R$ ensures that $F_{K,cave}$ is a concave function.

We now state and our result when the constraint is satisfied. (We extend the result to the general case in a Corollary directly after the Theorem).

Theorem 9. *If $c_r \leq 1,\ \forall\ r \in R$ then the Kikuchi free energy F_K can be minimized by a double-loop algorithm. The outer loop has time parameter t and is given by:*

$$b_r(x_r;t+1) = e^{-c_r\{E_r(x_r)+1\}}e^{-\gamma_r}\{b_r(x_r;t)\}^{1-c_r}e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)}e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{58}$$

The inner loops, to solve for the $\{\gamma_r\},\{\lambda_{r,s}\}$ has time parameter τ and is given by:

$$e^{\gamma_r(\tau+1)} = \sum_{x_r} e^{-c_r\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{1-c_r}e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s;\tau)}e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r;\tau)},$$

$$e^{2\lambda_{r,u}(x_u;\tau+1)} = \frac{\sum_{x\in r\backslash u} e^{-c_r\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{1-c_r}e^{-\gamma_r(\tau)}e^{-\sum_{s\in sub_d(r):s\neq u}\lambda_{r,s}(x_s;\tau)}e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r;\tau)}}{e^{-c_u\{E_u(x_u)+1\}}\{b_u(x_u;t)\}^{1-c_u}e^{-\gamma_u(\tau)}e^{-\sum_{s\in sub_d(u)}\lambda_{u,s}(x_s;\tau)}e^{\sum_{v\in sup_d(u):v\neq r}\lambda_{v,u}(x_u;\tau)}}. \tag{59}$$

Moreover, the inner loop is guaranteed to converge to the unique solution of the constraints.

Proof. *We split the Kikuchi free energy F_K into the convex and concave parts, $F_{K,vex}, F_{K,cave}$, given by equation (57). We can use Theorems 3,4 and 5 to obtain an algorithm to minimize the Kikuchi free energy.*

In more detail, we calculate the derivatives:

$$\frac{\partial F_{K,vex}}{\partial b_r(x_r)} = E_r(x_r) + 1 + \log b_r(x_r) + \gamma_r + \sum_{s\in sub_d(r)}\lambda_{r,s}(x_s) - \sum_{v\in sup_d(r)}\lambda_{v,r}(x_r), \tag{60}$$

$$\frac{\partial F_{K,cave}}{\partial b_r(x_r)} = (c_r - 1)\{E_r(x_r) + 1 + \log b_r(x_r)\}, \tag{61}$$

where we have ignored the constraint terms in $\frac{\partial F_{K,cave}}{\partial b_r(x_r)}$ because they can be absorbed into the constraints in $\frac{\partial F_{K,vex}}{\partial b_r(x_r)}$. (Recall from Theorems 3,4,5 that we are setting $\frac{\partial F_{K,vex}}{\partial b_r(x_r)}(t+1) = -\frac{\partial F_{K,cave}}{\partial b_r(x_r)}(t)$).

From equations (60,61), we can obtain the outer loop of our update algorithm (using Theorems 4,5):

$$b_r(x_r;t+1) = e^{-c_r\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{1-c_r}e^{-\gamma_r}e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)}e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{62}$$

The inner loop, to solve for the $\{\gamma_r\},\{\lambda_{r,s}\}$, is determined by Theorem 5. We know that solving the constraint equations one by one is guaranteed to converge to the unique solution for the constraints. The constraint equation for γ_r is $\sum_{x_r} b_r(x_r) = 1$ which can be solved to give an analytic expression for γ_r:

$$e^{\gamma_r} = \sum_{x_r} e^{-c_r\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{1-c_r}e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)}e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{63}$$

The constraint equation for $\lambda_{r,u}(x_u)$ is $\sum_{x \in r \setminus u} b_r(x_r) = b_u(x_u)$ which yields:

$$\sum_{x \in r \setminus u} e^{-c_r\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{1-c_r} e^{-\gamma_r} e^{-\sum_{s \in sub_d(r)} \lambda_{r,s}(x_s)} e^{\sum_{v \in sup_d(r)} \lambda_{v,r}(x_r)}$$
$$= e^{-c_u\{E_u(x_u)+1\}}\{b_u(x_u;t)\}^{1-c_u} e^{-\gamma_u} e^{-\sum_{s \in sub_d(u)} \lambda_{u,s}(x_s)} e^{\sum_{v \in sup_d(u)} \lambda_{v,u}(x_u)}, \tag{64}$$

which can be rearranged to give an analytic solution for $\lambda_{r,u}(x_u)$, see equation (59). Hence result.

We next show how to modify the algorithm to cases where some c_r are greater than 1.

Corollary. *Let $d = max_{r \in R} c_r$, then the double-loop algorithm can be extended to the general case by replacing c_r by c_r/d $\forall r \in R$ in the algorithm described in Theorem 9.*

Proof. *If there are regions with $c_r > 1$, then the concave-convex decomposition in equations (56, 57) is wrong (the "concave" part is not concave). But we can get a concave-convex decomposition by replacing $(c_r - 1)$ by $(c_r - d)$ for the concave part, see equation (57), (where $d = max_{r \in R} c_r$) and multiplying the convex part by d, see equation (56. By rescaling the energy (which doesn't alter the algorithm) this becomes equivalent to replacing $(c_r - 1)$ by $(c_r - d)/d$ – or, more simply, replacing c_r by c_r/d. With this change the algorithm proceeds as before.*

7 Discussion

This section covers three additional topics which are essentially review material (i.e. no novel results are presented here). Firstly, we discuss the relationship of the Bethe/Kikuchi approximation to estimation and how a temperature parameter can be introduced to obtain deterministic annealing. Secondly, we describe how the Bethe/Kikuchi and mean field free energies can be obtained from an information geometry viewpoint [1] using Kullback-Leibler divergences. Thirdly, we describe mean field algorithms analogous to the double-loop algorithm presented in this paper for Bethe/Kikuchi free energies.

7.1 MV and MAP Estimation and Deterministic Annealing

We now briefly discuss the use of the Bethe/Kikuchi approximations for estimation and how they can be modified to allow for temperature annealing [6]. For simplicity we concentrate only on the Bethe Free Energy.

The input to a double-loop algorithm is a probability distribution function $P(x_1, ..., x_N|y)$, see equation (1). The output of the algorithm is an estimate of the marginal and joint probability distributions.

In many optimization problems it is desired to estimate the variables $\{x_i\}$ from $P(x_1, ..., x_N|y)$. Two common estimators are the the Minimal Variance (MV) Estimator and the Maximum a Posteriori (MAP). From the estimates of the marginal probability distributions $\{b_i(x_i)\}$, provided by minimizing the Bethe/Kikuchi free energy, it is possible to directly obtain an approximation to the MV estimate by computing $\bar{x}_i = \sum_{x_i} x_i b_i(x_i)$.

The MAP estimate can be obtained by introducing a temperature parameter T. We replace the probability distribution $P(x_1, ..., x_N|y)$ by $\{P(x_1, ..., x_N|y)\}^{1/T}$ which implies, by equation (1), replacing the $\{\psi_i\}, \{\psi_{ij}\}, \{\phi_{ij}\}$ by $\{\psi_i^{1/T}\}, \{\psi_{ij}^{1/T}\}, \{\phi_{ij}^{1/T}\}$ and scaling by a factor T. This gives a family of Bethe free energies:

$$F_\beta^T(\{b_{ij}, b_i\}) = T \sum_{i,j:i>j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log b_{ij}(x_i,x_j) - \sum_{i,j:i>j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log \phi_{ij}(x_i,x_j)$$
$$-T \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log b_i(x_i) + \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log \psi_i(x_i), \tag{65}$$

where the original Bethe Free Energy, see equation (5), is be obtained by setting $T = 1$. We see that equation (65) is equivalent to having energy terms $-\sum_{i,j:i>j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log \phi_{ij}(x_i,x_j) + \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log \psi_i(x_i)$ (independent of temperature T) and temperature dependent entropy terms given by $T \sum_{i,j:i>j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log b_{ij}(x_i,x_j) - T \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log b_i(x_i)$. This is the same form for mean field annealing, see for example [7], [13].

Minimizing F_β^T gives a family of marginal distributions $\{b_i^T(x_i)\}$. As $T \mapsto 0$ the marginals $\{b_i^T(x_i)\}$ will become sharply peaked about the most probable values $x_i^{*,T} = \arg\max_{x_i} b_i^T(x_i)$. Therefore the estimates $\{x_i^{*,T}\}$ will become approximations to the MAP estimates $\{x_i^*\} = \arg\max_{\{x_i\}} P(x_1, ..., x_N|y)$. (Provided some technical conditions apply, see [23] for a theoretical analysis of what conditions are needed for the mean field theory approximation).

In addition, for many optimization problems it may be desirable to have some form of temperature annealing to help avoid local minima. This can be achieved by calculating the estimates of the marginals $\{b_i^T(x_i)\}$, and the joint distribution $\{b_{ij}^T(x_i, x_j)\}$, at high values of the temperature T and using these as initializations for the estimates for smaller values of the T. See [23], [13] for examples of this approach and for further references.

7.2 Kullback-Leibler Approximations and Information Geometry

Amari's theory of information geometry [1] gives a framework for obtaining approximations to probability distributions (where the approximations are restricted to have specific functional forms). This gives some insight into the Bethe/Kikuchi approximations. The material on the mean field theory approximation is related to Saul *et al* [16] and the Bethe free energy is similar to the derivation by Yedidia *et al* [22].

Let the target distribution is given by equation (1), $P(x_1, ..., x_N) = \frac{1}{Z}\prod_{i,j:i>j}\psi_{ij}(x_i, x_j)\prod_i \psi_i(x_i, y_i)$. Suppose we want to approximate it by a factorizable distribution $P_F(x_1, ..., x_N) = \prod_{i=1}^N b_i(x_i)$ (with $\sum_{x_i} b_i(x_i) = 1 \, \forall \, i$). One measure of similarity, suggested by information geometry, is to seek the distribution $P_F(x_1, ..., x_N)$ which minimizes the Kullback-Leibler divergence to $P(x_1, ..., x_N)$:

$$D(P_F||P) = \sum_{x_1,\ldots,x_N} P_F(x_1, ..., x_N) \log \frac{P_F(x_1, ..., x_N)}{P(x_1, ..., x_N)}. \tag{66}$$

We can re-express this as:

$$D(P_F||P) = -\sum_i \sum_{x_i} b_i(x_i) \log \psi_i(x_i) - \sum_{i,j}\sum_{x_i,x_j} b_i(x_i)b_j(x_j)\log\psi_{ij}(x_i, x_j) + \sum_i\sum_{x_i} b_i(x_i)\log b_i(x_i). \tag{67}$$

Minimizing $D(P_F||P)$ with respect to the $\{b_i\}$ (with the constraints that $\sum_i b_i(x_i) = 1 \ \forall i$) gives the mean field free energy used for many optimization applications, see for example [23], [13]. (The notation is different in those papers. For example to obtain the formulation used in [23] we relate the variables x_i, x_j to a, b and identify the variables $\{b_i(x_i)\}$ with the matching variables $\{S_{ia}\}$ and the terms $-\log\psi_i(x_i)$ and $-\log\psi_{ij}(x_i, x_j)$ with terms $-A_{ia}$ and T_{ijab}. The variables are constrained so that $\sum_a S_{ia} = 1, \ \forall \ i$.)

The Bethe free energy can be obtained in a similar way (although, as we will discuss, an additional approximation is needed if the graph has closed loops). We seek to approximate the distribution $P(x_1, ..., x_N)$ by a distribution $P_\beta(x_1, ..., x_N)$ which has marginals $\{b_i(x_i)\}$ and joint distributions $\{b_{ij}(x_i, x_j)\}$. Once again, we attempt to find the approximation $P_\beta(x_1, ..., x_N)$ which minimizes the Kullback-Leibler divergence:

$$\begin{aligned} D(P_\beta||P) = \sum_{x_1,\ldots,x_N} P_\beta(x_1, ..., x_N)\log\frac{P_\beta(x_1, ..., x_N)}{P(x_1, ..., x_N)} = \quad & \textstyle\sum_{x_1,\ldots,x_N} P_\beta(x_1, ..., x_N)\log P_\beta(x_1, ..., x_N) \\ & -\textstyle\sum_{x_1,\ldots,x_N} P_\beta(x_1, ..., x_N)\log P(x_1, ..., x_N). \end{aligned} \tag{68}$$

The second term of the Kullback-Leibler, see equation (68), can be expressed as $-\sum_i\sum_{x_i} b_i(x_i)\log\psi_i(x_i) - \sum_{i,j}\sum_{x_i,x_j} b_{ij}(x_i, x_j)\log\psi_{ij}(x_i, x_j)$.

The first term is the entropy of the approximating distribution $P_\beta(x_1, ..., x_N)$. The Bethe approximation consists of assuming that we can express this as:

$$\sum_{x_1,\ldots,x_N} P_\beta(x_1, ..., x_N)\log P_\beta(x_1, ..., x_N) \approx \sum_{i,j}\sum_{x_i,x_j} b_{ij}(x_i, x_j)\log b_{ij}(x_i, x_j) - \sum_i (n_i - 1)\sum_{x_i} b_i(x_i)\log b_i(x_i). \tag{69}$$

It can be shown that this approximation is exact *if the graph has no loops.* (Recall that we only define joint distribution $b_{ij}(x_i, x_j)$ for nodes i, j which are directly joined in the graph. Also n_i is the number of neighbours). If the graph has no loops then the marginal and joint distributions (combined with the maximum entropy principle) determine a unique probability distribution (see, for example, [15]):

$$P_\beta(x_1, ..., x_N) = \frac{\prod_{i,j} b_{ij}(x_i, x_j)}{\prod_i \{b_i(x_i)\}^{n_i - 1}}. \quad (70)$$

In this case, the entropy can be computed directly to give equation (69). But *if the graph has loops then $P_\beta(x_1, ..., x_N)$ given by equation (70) is not a normalized distribution.* The Bethe approximation assumes that the entropy can be written as equation (69) even when the graph has closed loops.

Using equation (69) for the entropy gives the result:

$$D(P_\beta||P) \approx -\sum_i \sum_{x_i} b_i(x_i) \log \psi_i(x_i) - \sum_{i,j} \sum_{x_i, x_j} b_{ij}(x_i, x_j) \log \psi_{ij}(x_i, x_j) \quad + \sum_{i,j} \sum_{x_i, x_j} b_{ij}(x_i, x_j) \log b_{ij}(x_i, x_j)$$
$$- \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log b_i(x_i), (71)$$

which is exact for graphs with no loops and an approximation otherwise.

It can be quickly verified that the right hand side of equation (71) is the Bethe free energy (which involves using the relations $\phi_{ij}(x_i, x_j) = \psi_{ij}(x_i, x_j)\psi_i(x_i)\psi_j(x_j)$ and the identity $\sum_{i,j} \sum_{x_i, x_j} b_{ij}(x_i, x_j) \log \psi_i(x_i) = \sum_i n_i \sum_{x_i} b_i(x_i) \log \psi_i(x_i)$).

In summary, the mean field and the Bethe free energies can be obtained replacing the true distribution $P(x_1, ..., x_n)$ by approximations with probability distributions with specified marginal distributions (mean field) or specified marginal and joint distributions (Bethe). The Bethe free energy requires an additional approximation, about the entropy, if the graph has closed loops. The Kikuchi approximation can be obtained in a similar way. Essentially all these approaches (mean field, Bethe, Kikuchi) seek to approximate the true distributions $P(x_1, ..., x_N)$ by distributions which are easier to compute with.

7.3 Mean Field Algorithms

For completeness, we now describe how the mean field equations can be solved using algorithms similar to the double-loop. This summarizes, and perhaps simplifies, work by [7],[23],[13]. (See also chapter by Peterson and Yuille in [2]). As we will describe, mean field theory has been used to motivate algorithms for solving optimization problems.

Many optimization problems (e.g. the Traveling Salesman problem) – see references in [7],[23],[13] – can be formulated as minimizing a cost function whose arguments are *discrete variables*:

$$E[\{V_{ia}\}] = -\sum_i \sum_a V_{ia} A_{ia} + \sum_{ij} \sum_{ab} T_{ijab} V_{ia} V_{jb}, \quad (72)$$

where the $\{V_{ia}\}$ are binary variables (i.e. $V_{ia} \in \{0, 1\}\ \forall i, a$) which are required to satisfy *linear constraints*, such as $\sum_a V_{ia} = 1,\ \forall i$.

For the Traveling Salesman Problem, the goal is to find the shortest tour that passes through a set of cities. The energy function can be written as $E_{TSP}[\{V_{ia}\}] = \sum_{i,a,b} V_{i,a} V_{i+1,b} d_{ab}$ where d_{ab} is the distance between cities a and b, $V_{i,a} = 1$ if city a is the i^{th} city on the tour (and $V_{i,a} = 0$ otherwise). The constraints $\sum_i V_{i,a} = 1,\ \forall\ a$ and $\sum_a V_{i,a} = 1,\ \forall\ i$ impose the requirements that each city is visited once, and once only, by the tour.

Mean field algorithms, as we will describe, can be designed to obtain local minima of energy functions such as $E[\{V_{ia}\}]$. The strategy is to transform the *discrete energy function* $E[\{V_{ia}\}]$ into a *continuous free energy function* $F[\{b_{ia}\}; T]$ which is a function of continuous variables $\{b_{ia}\}$ and a temperature parameter T. The free energy function should be constructed so that, as $T \mapsto 0$, the variables $\{b^*_{ia}(T)\}$ which minimize $F[\{b_{ia}\}; T]$ should be *identical* to the variables $\{V^*_{ia}\}$ which minimize $E[\{V_{ia}\}]$.

To develop a mean field algorithm to attempt to *minimize* $E[\{V_{ia}\}]$ requires four stages. *Firstly,* we reformulate the problem as finding the $\{V_{ia}\}$ which *maximize* the probability distribution:

$$P[\{V_{ia}\}] = \frac{1}{Z} e^{-E[\{V_{ia}\}]}, \quad (73)$$

where Z is a normalization constant. Observe that this probability distribution is of the same form as equation (1). (We identify the $\{A_i\}, \{T_{ijab}\}$ with the exponentials of the $\{\psi_i\}, \{\psi_{ij}\}$ where the $\{V_{ia}\}$ are used to indicate the states $\{x_i\}$. I.e. $V_{ia} = 1$ means that x_i takes its a^{th} value).

Secondly, we apply the factorization assumption to obtain a mean field free energy, see equation (67). Here the $\{b_i(x_i)\}$ correspond to the (approximate) marginal distributions of $P[\{V_{ia}\}]$. *Thirdly*, we need to introduce a temperature parameter T and anneal so that, as $T \mapsto 0$, we can obtain estimates of the variables $\{V_{ia}\}$ (those that minimize the energy $E[\{V_{ia}\}]$). *Fourthly*, we need to find an algorithm which will minimize the mean field free energy for small T.

From subsection (7.2), the mean field free energy (see equation (67)) can be expressed as :

$$F^T_{mft}(\{b_i\}) = -\sum_i \sum_{x_i} b_i(x_i) \log \psi_i(x_i) - \sum_{i,j} \sum_{x_i,x_j} b_i(x_i) b_j(x_j) \log \psi_{ij}(x_i,x_j) + T \sum_i \sum_{x_i} b_i(x_i) \log b_i(x_i), \tag{74}$$

where we have added a temperature parameter T by the annealing method described in subsection (7.1).

We now describe how these mean field theory approach relates to the double-loop algorithm described in the paper and, in particular, the techniques described in section (3). First we point out an apparent major difference between the mean field free energy and the Bethe/Kikuchi free energies.

In general, the mean field free energy cannot be decomposed into a convex and a concave part. The problem lies with the term $\sum_{i,j} \sum_{x_i,x_j} b_i(x_i) b_j(x_j) \log \psi_{ij}(x_i,x_j)$ in equation (74) which is a quadratic potential which, in general, cannot be decomposed into a sum of concave and convex parts. For the Bethe free energy, see equation (5), this term is replaced by $\sum_{i,j} \sum_{x_i,x_j} b_{ij}(x_i,x_j) \log \psi_{ij}(x_i,x_j)$ which is linear in $\{b_{ij}\}$ and so the problem does not arise (nor does it for the Kikuchi approximation).

Moreover, the form of this energy term $\sum_{i,j} \sum_{x_i,x_j} b_i(x_i) b_j(x_j) \log \psi_{ij}(x_i,x_j)$ also causes potential difficulties when mean field algorithms are applied to solve optimization problems. In the limit as $T \mapsto 0$ the marginal probability distributions should be strongly peaked so that the mean state $\sum_{x_i} x_i b_i(x_i)$ corresponds to a possible value of the discrete variable V_{ia}. Intuitively, as $T \mapsto 0$ the distribution $\{P[\{V_{ia}\}]\}^{1/T}$ (after normalization) should be sharply peaked about its mode $V^*_{ia} = \arg\max_{V_{ia}} P[\{V_{ia}\}]$. Therefore the estimated marginal distributions $\{b_i(x_i)\}$ should be sharply peaked as $T \mapsto 0$. (Indeed, as $T \mapsto 0$, the minima $\{b^*_{ia}\}$ of the continuous free energy should correspond to the minima of the discrete energy $E[\{V_{ia}\}]$). However, see [23], *the minimum of the mean field free energy given by equation (74) may not correspond to sharply peaked marginal distributions as* $T \mapsto 0$.

It turns out, however, that both these problems (i.e. the lack of a concave plus convex decomposition and the behaviour of the marginals as $T \mapsto 0$) can be fixed by adding an additional term to the mean field free energy. This term is of form $-K \sum_i \sum_{x_i} \{b_i(x_i)\}^2$. It does not affect the minima of the discrete energy function because it is constant in this case (for the discrete case the constraints imply that $-K \sum_i \sum_{x_i} \{b_i(x_i)\}^2 = -KN$, where N is the size of the problem). The constant K is chosen to ensure that the "energy" terms $E_{mft}[\{b_i(x_i)\}]$ are concave, where:

$$E_{mft}[\{b_i(x_i)\}] = -\sum_i \sum_{x_i} b_i(x_i) \log \psi_i(x_i) - \sum_{i,j} \sum_{x_i,x_j} b_i(x_i) b_j(x_j) \log \psi_{ij}(x_i,x_j) - K \sum_i \sum_{x_i} \{b_i(x_i)\}^2. \tag{75}$$

The mean field free energy can then be split into a concave part $E_{mft}[\{b_i(x_i)\}]$ plus a convex (entropy) part $E_{ent}[\{b_i(x_i)\}] = T \sum_i \sum_{x_i} b_i(x_i) \log b_i(x_i)$.

The modified mean field free energy $E_{mft}[\{b_i(x_i)\}] + E_{ent}[\{b_i(x_i)\}]$ is now of the concave plus convex form. This means we can apply the results of section (3) to derive algorithms which converge to mimima of the mean field free energy. The precise forms of these algorithms will depend on the types of linear constraints imposed. For certain optimization problems the constraints can be solved directly. For others, a double-loop algorithm is required [13].

We note that adding an additional energy term of form $-K \sum_i \sum_{x_i} \{b_i(x_i)\}^2$ to the mean field free energy has been often used as a heuristic to ensure that the mean field solutions are "meaningful". Yuille and Kosowsky [23] pointed out one could guarantee that mean field solutions corresponded to solutions of the discrete optimization problem by requiring that the energy term $E_{mft}[\{b_i(x_i)\}]$ be concave, and gave bounds on the magnitude of K required to ensure this. Rangarajan [13] realized that these bounds on K

were sufficient to ensure that his discrete update rule was guaranteed to converge. (Ranagarajan's derivation is in terms of Legendre transforms, but it appears to be equivalent to the concave plus convex decomposition described in this paper).

Why does the mean field free energy not have a natural convex/concave decomposition? The reason is that the mean field approach attempts to use a factorizable probability distribution to approximate a distribution, see equation (1), which has pairwise energy potentials $\log \psi_{ij}(x_i, x_j)$. The Bethe free energy, in the form given in this paper, is different because it uses joint (i.e. pairwise) probability distributions in its approximation. In general, we will always obtain a convex/concave decomposition *provided the order of interaction of the approximating probability distributions is as great as the order of interactions of the energy potentials in the distribution being approximated.*

In brief, the mean field free energy can be turned into the concave plus convex form by adding an additional energy term (which does not affect the solution of the discrete optimization problem). This enables one to construct algorithms to minimize the mean field free energy which may, or may not, be double-loop algorithms (depending on the form of the constraints).

8 Conclusion

The aims of this paper were to analyze the BP algorithm of Yedidia *et al* [22] and to design a double-loop algorithm which can be proven to converge to a minimum of the Bethe free energy. We have shown that the dual space formulation throws some light on BP. It may also be helpful to determine when BP, and related algorithms, converge to the correct result – see [20].

More constructively, we derived a double-loop algorithm based on separating the Bethe free energy into concave and convex parts and then using a discrete iterative algorithm to solve for the constraints. We showed that the BP algorithm is formally similar to the double-loop algorithm (and might be an effective approximation in some cases).

We also generalized our results to the Kikuchi free energy. This yielded both a dual formulation of the free energy and a double-loop algorithm that can be proven to converge to a minimum of the Kikuchi free energy.

Finally, we described how previous mean field theory work could be interpreted within this framework. It seems that the differences between mean field theory and the Bethe/Kikuchi methods is merely the degree of the approximation used (i.e. do we try to approximate the target distribution by a factorizable distribution – the mean field approach – or do we look for a higher order Bethe/Kikuchi approximation). Mean field theory algorithms can be very effective on difficult optimization problems (e.g., see [13]) provided the factorization assumptions is good enough but they also fail on other difficult problems such as the non-Euclidean Traveling Salesman Problem (Rangarajan – personal communication).

Current work is also to implement the double-loop algorithm on computer vision problems and compare its performance to alternatives such as mean field theory algorithms. It has already been demonstrated [21] that the BP algorithms give good results in situations where mean field algorithms break down.

Acknowledgements

I would like to acknowledge helpful conversations with James Coughlan and Huiying Shen. Yair Weiss pointed out a conceptual bug in the original dual formulation. Anand Rangarajan pointed out the connections to Legendre transforms. Sabino Ferreira read the manuscript carefully, made many useful comments, and gave excellent feedback. Jonathan Yedidia pointed out a faulty assumptions about the $\{c_r\}$. This work was supported by the National Institute of Health (NEI) with grant number RO1-EY 12691-01.

References

[1] S. Amari. "Differential Geometry of curved exponential families – Curvature and information loss. Annals of Statistics, vol. 10, no. 2, pp 357-385. 1982.

[2] M. Arbib (Ed.) **The Handbook of Brain Theory and Neural Networks**. A Bradford Book. The MIT Press. 1995.

[3] C. Domb and M.S. Green (Eds). **Phase Transitions and Critical Phenomena**. Vol. 2. Academic Press. London. 1972.

[4] W.T. Freeman and E.C. Pasztor. "Learning low level vision". In *Proc. International Conference of Computer Vision.* ICCV'99. pp 1182-1189. 1999.

[5] B. Frey. **Graphical Models for Pattern Classification, Data Compression and Channel Coding.** MIT Press. 1998.

[6] S. Kirkpatrick, C. Gelatt (Jr.), and M. Vecchi. "Optimization by Simulated Annealing". *Science*, 220:671-680. 1983.

[7] J. Kosowsky and A.L. Yuille. "The Invisible Hand Algorithm: Solving the Assignment Problem with Statistical Physics". *Neural Networks.*, Vol. 7, No. 3, pp 477-490. 1994.

[8] J. Kosowsky. **Flows Suspending Iterative Algorithms**. PhD Thesis. Division of Applied Sciences. Harvard University. Cambridge, Massachusetts. 1995.

[9] C.M. Marcus and R.M. Westervelt. "Dynamics of iterated- map neural networks". *Physical Review A*, 40, pp 501-504. 1989.

[10] R.J. McEliece, D.J.C. Mackay, and J.F. Cheng. "Turbo decoding as an instance of Pearl's belief propagation algorithm". *IEEE Journal on Selected Areas in Communication.* 16(2), pp 140-152. 1998.

[11] K.P. Murphy, Y. Weiss, and M.I. Jordan. "Loopy belief propagation for approximate inference: an empirical study". In *Proceedings of Uncertainty in AI.* 1999.

[12] J. Pearl. **Probabilistic Reasoning in Intelligent Systems**. Morgan Kaufmann. 1988.

[13] A. Rangarajan, S. Gold, and E. Mjolsness. "A Novel Optimizing Network Architecture with Applications". *Neural Computation,* 8(5), pp 1041-1060. 1996.

[14] A. Rangarajan, A.L. Yuille, S. Gold. and E. Mjolsness."A Convergence Proof for the Softassign Quadratic assignment Problem". In *Proceedings of NIPS'96.* Snowmass. Colorado. 1996.

[15] B.D. Ripley. "Pattern Recognition and Neural Networks". Cambridge University Press. 1996.

[16] L. K. Saul, T. Jaakkola, and M. I. Jordan. "Mean Field Theory for Sigmoid Belief Networks". *Journal of Artificial Intelligence Research*, 4, 61-76, 1996.

[17] R. Sinkhorn. "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices". *Ann. Math. Statist.*. 35, pp 876-879. 1964.

[18] G. Strang. **Introduction to Applied Mathematics**. Wellesley-Cambridge Press. Wellesley, Massachusetts. 1986.

[19] F.R. Waugh and R.M. Westervelt. "Analog neural networks with local competition: I. Dynamics and stability". *Physical Review E*, 47(6), pp 4524-4536. 1993.

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

[21] Y. Weiss. "Comparing the mean field method and belief propagation for approximate inference in MRFs". To appear in **Advanced Mean Field Methods**. Saad and Opper (Eds). MIT Press. 2001.

[22] J.S. Yedidia, W.T. Freeman, Y. Weiss. "Bethe free energy, Kikuchi approximations and belief propagation algorithms". To appear in NIPS'2000. 2000.

[23] A.L. Yuille and J.J. Kosowsky. "Statistical Physics Algorithms that Converge." Neural Computation. **6**, pp 341-356. 1994.