CCCP Algorithms to Minimize the Bethe and Kikuchi Free Energies: Convergent Alternatives to Belief Propagation

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

Abstract

This paper introduces a class of discrete iterative algorithms which are provably convergent alternatives to belief propagation (BP) and generalized belief propagation (GBP). Our work builds on recent results by Yedidia, Freeman and Weiss (2000) who showed that the fixed points of BP and GBP algorithms correspond to extrema of the Bethe and Kikuchi free energies respectively. We obtain two algorithms by applying CCCP to the Bethe and Kikuchi free energies respectively (CCCP is a procedure, introduced here, for obtaining discrete iterative algorithms by decomposing a cost function into a concave and a convex part). We implement our CCCP algorithms on 2D and 3D spin glasses and compare their results to BP and GBP. Our simulations show that the CCCP algorithms are stable and converge very quickly (the speed of CCCP is similar to that of BP/GBP). Unlike CCCP, BP will often not converge for these problems (GBP usually, but not always, converges). The results found by CCCP applied to the Bethe or Kikuchi free energies are equivalent, or slightly better than, those found by BP or GBP respectively (when BP/GBP converge). Note that for these, and other problems, BP/GBP give very accurate results (see Yedidia *et al* 2000) and failure to converge is their major error mode. Finally, we point out that our algorithms have a large range of inference and learning applications.

To appear in Neural Computation

1 Introduction

Recent work by Yedidia, Freeman and Weiss (2000) unified two approaches to statistical inference. They related the highly successful belief propagation (BP) algorithms (Pearl 1988) to variational methods from statistical physics and, in particular, to the Bethe and Kikuchi free energies (Domb and Green 1972). These BP algorithms typically give highly accurate results (Yedidia *et al* 2000). But BP algorithms do not always converge and, indeed, failure to converge is their major error mode. This paper develops new algorithms which are guaranteed to converge to extrema of the Bethe or Kikuchi free energies and hence are alternatives to belief propagation.

Belief propagation (Pearl 1988) is equivalent to the sum-product algorithm developed by Gallager to decode Low Density Parity Codes (LDPC) (Gallager 1963). In recent years, see (Forney 2001) for a review, the coding community has shown great interest in sum-product algorithms and LDPC's. It is predicted that this combination will enable the coding community to design practical codes which approach the Shannon performance limit (Cover and Thomas 1991) while requiring only limited computation. In particular, it has been shown that the highly successful turbo codes (Berrou *et al* 1993) can also be interpreted in terms of BP (McEliece *et al* 1998). Although BP has only been proven to converge for tree-like graphical models (Pearl 1988) it has been *amazingly* successful when applied to inference problems with closed loops (Freeman and Pasztor 1999, Frey 1998, Murphy *et al* 1999) including these, particularly important, coding applications (Forney 2001). When BP converges it (empirically) usually seems to converge to a good approximation to the true answer.

Statistical physics has long been a fruitful source of ideas for statistical inference (Hertz, Krogh, Palmer 1991). The mean field approximation, which can be formulated as minimizing a (factorized) mean field free energy, has been used to motivate algorithms for optimization and learning (see chapters by Peterson and Yuille in Arbib 1995). The Bethe and Kikuchi free energies (Domb and Green 1972) contain higher order terms than the (factorized) mean field free energies commonly used. It is therefore hoped that algorithms which minimize the Bethe or Kikuchi free energies will outperform standard mean field theory and be useful for optimization and learning applications. Overall, there is a hierarchy of variational approximations in statistical physics which start with mean field, proceed to Bethe, and finish with Kikuchi (which subsumes Bethe as a special cases).

Yedidia *et al*'s result (Yedidia *et al* 2000) proved that the fixed points of BP correspond to the extrema of the Bethe free energy. They also developed a generalized belief propagation (GBP) algorithm whose fixed points correspond to the extrema of the Kikuchi free energy. In practice, when BP and GBP converge they go to low energy minima of the Bethe/Kikuchi free energies. In general, we expect that GBP gives more accurate results than BP since Kikuchi is a better approximation than Bethe. Indeed, empirically GBP outperformed BP on 2D spin-glasses and converged very close to the true solution (Yedidia *et al* 2000).

But these results say little about failure to converge of BP (or GBP) (which are the dominant error modes of the algorithms – if BP or GBP converge then they typically converge to a close approximation to the true solution). This motivates the search for other algorithms to minimize the Bethe/Kikuchi free energies.

This paper develops new algorithms that are guaranteed to converge to extrema of the Bethe and Kikuchi free energies. (In computer simulations so far they always converged to minima). The algorithms have some similarities to BP/GBP algorithms which estimate “beliefs” by propagating “messages”. The new algorithms also propagate messages (formally Lagrange multipliers) but, unlike BP/GBP, the propagation depends on current estimates of the beliefs which must be re-estimated periodically. This similarity may help explain when BP and GBP converge. But, in any case, these new algorithms offer alternatives to BP/GBP which may be of particular use in regimes where BP/GBP do not converge.

The algorithms are developed using a Concave Convex procedure (CCCP) which starts by decomposing the free energy into concave and convex parts. From this decomposition it is possible to obtain a discrete update rule which decreases the free energy at each iteration step. This procedure is very general and is developed further by Yuille and Rangarajan (2001) with applications to many other optimization problems. It builds on results developed when studying mean field theory (Kosowsky and Yuille 1994, Yuille and Kosowsky 1994, Rangarajan *et al* 1996a, Rangarajan *et al* 1996b).

The algorithms were implemented and tested on 2D and 3D spin glass energy functions. These contain many closed loops (so BP is expected to have difficulties – Yedidia *et al* 2000). Our results show that our algorithms are stable, converge very rapidly, and give equivalent or better results than BP/GBP (BP often fails to converge for these problems).

We note that there has recently been a range of new algorithms which either act as variants or improvements of BP (this paper was originally submitted before we learnt of these alternative algorithms). These include Teh and Welling (2001), Wainwright *et al* (2001), and Chaing and Forney (2001). Of these, the algorithm by Teh and Welling seems most similar to the CCCP algorithms. Their algorithm is also guaranteed to converge to an extremum of the Bethe free energy. Comparative simulations of the two algorithms have been performed (Teh and Welling – private communication) which indicate that the differences between the two algorithms lie mainly in the convergence rates rather than in the quality of the solutions obtained. But these results are preliminary. The relative advantages of these algorithms is work for the future.

The structure of this paper is as follows. Section (2) describes the Bethe free energy and BP algorithms. In section (3) we describe the design principle, CCCP, of our algorithm. Section (4) applies CCCP to the Bethe free energy and gives a double-loop algorithm that is guaranteed to converge (we also discuss formal similarities to BP). In section (5) we apply CCCP to the Kikuchi free energy and obtain a provably convergent algorithm. Section (6) applies our CCCP algorithm to 2D and 3D spin glass energy models using either the Bethe or Kikuchi approximation and compares their performance to BP and GBP. Finally, we discuss related issues in Section (7).

2 The Bethe Free Energy and the BP algorithms

In this section we describe the Bethe free energy and the BP algorithm (following the formulation of Yedidia *et al* 2000).

The Bethe free energy (Domb and Green 1972, Yedidia *et al* 2000) is a variational technique from statistical physics. The idea is to replace an inference problem we cannot solve by an approximation which is solvable. For example, we may want to estimate the variables $x_1^*, ..., x_N^*$ which are the most probable states of a distribution $P(x_1, ..., x_N|y)$. This, however, may be computationally expensive (eg. NP-complete). Instead we can apply variational methods where we seek an approximate solution by minimizing a free energy function to obtain a mean field theory solution (see chapters by Peterson and Yuille in Arbib 1995).

The Bethe free energy is an approximation that uses joint distributions between variables. It can give exact solutions in situations where standard mean field theory only gives poor solutions (Weiss 2001). As we will discuss later, the Kikuchi free energy gives higher order approximations (see discussion at the end of this section).

Consider a graph with nodes $i = 1, ..., N$. The problem specification will determine connections between the nodes. We will only list connections ij for node pairs i, j which are connected. (Ie. variables, such as ψ_{ij}, b_{ij}, which depend on two nodes will only be defined for pairs of nodes which are connected in the graph). 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 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 $\psi_i(x_i, y_i)$. (Recall that if nodes i and j are not connected then we do not have a term ψ_{ij}).

For example, in the 2D spin-glass network, see figure (1), the variable i labels nodes on a two-dimensional grid (see section (6)). It is convenient to represent these nodes by vector labels $\vec{i}$ with the first and second components corresponding to the positions in the 2D lattice and with $\Delta\vec{h}, \Delta\vec{v}$ corresponding to shifts between lattice sites in the horizontal and vertical directions. The 2D spin-glass involves nearest neighbour interactions only. So we have potentials $\psi_{\vec{i}}(x_{\vec{i}})$ at each lattice site and potentials $\psi_{\vec{i},\vec{i}+\Delta\vec{h}}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{h}})$ and $\psi_{\vec{i},\vec{i}+\Delta\vec{v}}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{v}})$ linking neighbouring lattice sites.

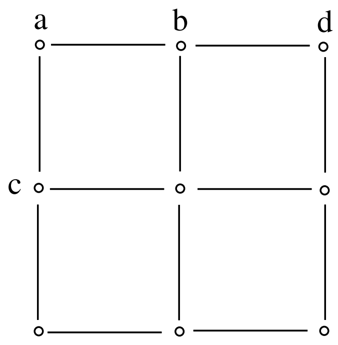

Figure 1: The grid for a 2D spin glass. Each node a, b, c, d represents a binary valued spin variable. Our convention is that a is site $\vec{i}$, b, c, d are sites $\vec{i}+\Delta\vec{h}, \vec{i}+\Delta\vec{v}, \vec{i}+2\Delta\vec{h}$.

The goal is to determine an 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 BP algorithm introduces variables $m_{ij}(x_j)$ which correspond to "messages" that node i sends to node j (later we will see how these messages correspond to Lagrange multipliers which impose consistency constraints on the beliefs). 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). Again we only have messages between nodes which are connected. Nodes are not connected to themselves (i.e $m_{ii}(x_i, t) = 1, \ \forall \ i$).

For the 2D spin-glass, see figure (1), we can represent the messages (and later the λ's from CCCP) by the nodes into which they flow, see figure (2).

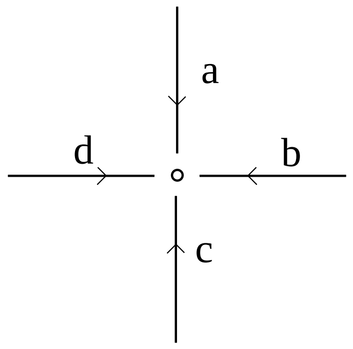

Figure 2: The quantities "a,c,b,d" flowing into the central pixel correspond to Lagrange multipliers or messages for CCCP or BP respectively. They correspond either to $\lambda_{\vec{i} \pm \Delta \vec{v}, \vec{i}}(x_{\vec{i}}), \lambda_{\vec{i} \pm \Delta \vec{h}, \vec{i}}(x_{\vec{i}})$ or $m_{\vec{i} \pm \Delta \vec{h}, \vec{i}}(x_{\vec{i}}), m_{\vec{i} \pm \Delta \vec{v}, \vec{i}}(x_{\vec{i}})$.

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 (Pearl 1988).

The Bethe free energy of this system is written as (Yedidia *et al* 2000):

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

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:

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

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

These constraints can be imposed by using Lagrange multipliers $\{\gamma_{ij} : i > j\}$ and $\{\lambda_{ij}(x_j) : i \neq j\}$ and adding terms:

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

$$+\sum_{i,j:i>j}\sum_{x_i}\lambda_{ji}(x_i)\{\sum_{x_j} b_{ij}(x_i,x_j)-b_i(x_i)\}. \tag{8}$$

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 K-L 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* (2000) proved that the fixed points of the BP algorithm correspond to extrema of the Bethe free energy (with the linear constraints of equation (6)). Their results can be obtained by differentiating the Bethe free energy and using the substitution $m_{ji}(x_i) = e^{\lambda_{ji}(x_i)} e^{-\frac{1}{n_i-1}\sum_k \lambda_{ki}(x_i)}$ and the inverse $e^{-\lambda_{ij}(x_j)} = \prod_{k\neq i} m_{kj}(x_j)$.

Observe that the standard (factorized) mean field free energy (see chapters by Peterson and Yuille in Arbib 1995) can be obtained from the Bethe free energy by setting $b_{ij}(x_i,x_j) = b_i(x_i)b_j(x_j),\ \forall\, i,j$. This gives:

$$F_{mean-field} = -\sum_i\sum_{x_i} b_i(x_i)\log\psi_i(x_i) - \sum_{i,j}\sum_{x_i,x_j} b_i(x_i)b_j(x_j)\log\psi_{ij}(x_i,x_j)$$

$$+\sum_i\sum_{x_i} b_i(x_i)\log b_i(x_i). \tag{9}$$

By comparing equation (5) to equation (9) we see that the Bethe free energy contains higher order terms than the mean field energy. It is therefore plausible that algorithms which minimize the Bethe free energy will perform better for optimization and learning than those which use the mean field approximation.

In general, (factorized) mean-field, Bethe and Kikuchi give a hierarchy of approximations to the underlying distribution $P(x_1,...,x_N|y)$. They can be derived by minimizing the K-L divergence $D(P_A||P) = \sum_{x_1,...,x_N} P_A(x_1,...,x_N)\log\frac{P_A(x_1,...,x_N)}{P(x_1,...,x_N|y)}$

with respect to an approximating distribution $P_A(x_1, ..., x_N)$. If the P_A are restricted to being factorized distributions, $P_A(x_1, ..., x_N) = \prod_{i=1}^N b_i(x_i)$, then we obtain the mean field free energy, see equation (9). We can obtain the Bethe and Kikuchi free energies by allowing $P_A(.)$ to have different forms. (Though additional approximations are still required to get Bethe or Kikuchi unless the graph has no loops).

3 The Concave Convex Procedure (CCCP)

Our algorithms to minimize the Bethe and Kikuchi free energies are based on the Concave Convex Procedure (CCCP) described in this section. This approach was developed (in this paper) for the specific case of Bethe and Kikuchi but is far more general. Indeed many existing discrete time iterative dynamical systems can be interpreted in terms of CCCP (Yuille and Rangarajan 2001).

Our main results are given by Theorem's 1,2,3 and show that we can obtain discrete iterative algorithms to minimize energy functions which are the sum of a convex and a concave term. These algorithms will typically, but not always (Yuille and Rangarajan 2001), require an inner and an outer loop. 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 1. *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:*

$$\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t), \tag{10}$$

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

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

$$\begin{aligned} E_{vex}(\vec{z}_2) &\geq E_{vex}(\vec{z}_1) + (\vec{z}_2 - \vec{z}_1) \cdot \vec{\nabla} E_{vex}(\vec{z}_1) \\ E_{cave}(\vec{z}_4) &\leq E_{cave}(\vec{z}_3) + (\vec{z}_4 - \vec{z}_3) \cdot \vec{\nabla} E_{cave}(\vec{z}_3), \end{aligned} \tag{11}$$

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 (11) and the algorithm definition (i.e. $\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t)$) we find that:

$$E_{vex}(\vec{z}^{t+1}) + E_{cave}(\vec{z}^{t+1}) \leq E_{vex}(\vec{z}^t) + E_{cave}(\vec{z}^t), \tag{12}$$

which proves the claim.

Observe that the convexity of $E_{vex}(\vec{z})$ means that the function $\vec{\nabla} E_{vex}(\vec{z}^{t+1})$ is invertible. In other words, the tangent to $E_{vec}(\vec{z}^{t+1})$ determines $\vec{z}^{t+1}$ uniquely.

Theorem 1 generalizes previous results by Marcus, Waugh and Westervelt (Marcus and Westervelt 1989, Waugh and Westervelt 1993) on the convergence of discrete iterated neural networks. (They discussed a special case where one function was convex and the other was *implicitly*, but not *explicitly*, concave).

The algorithm can be illustrated geometrically by the reformulation shown in figure (3) (suggested by James M. Coughlan). Think of decomposing the energy function $E(\vec{z})$ into $E_1(\vec{z}) - E_2(\vec{z})$ where both $E_1(\vec{z})$ and $E_2(\vec{z})$ are convex. (This is equivalent to decomposing $E(\vec{z})$ into a a convex term $E_1(\vec{z})$ *plus* a concave term $-E_2(\vec{z})$). The algorithm proceeds by matching points on the two terms which have the same tangents. For an input $\vec{z}_0$ we calculate the gradient $\vec{\nabla} E_2(\vec{z}_0)$ and find the point $\vec{z}_1$ such that $\vec{\nabla} E_1(\vec{z}_1) = \vec{\nabla} E_2(\vec{z}_0)$. We next determine the point $\vec{z}_2$ such that $\vec{\nabla} E_1(\vec{z}_2) = \vec{\nabla} E_2(\vec{z}_1)$, and repeat.

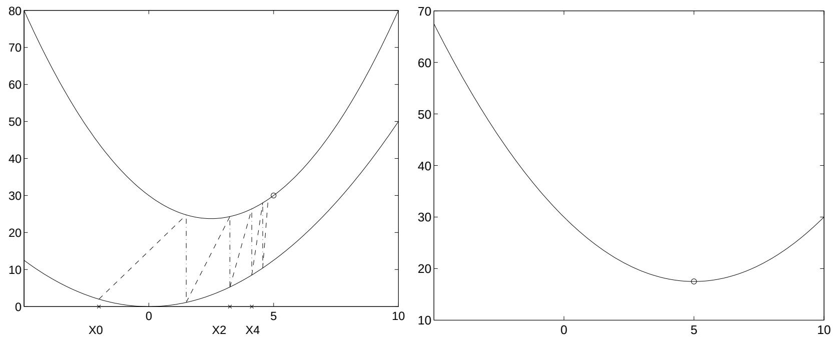

Figure 3: A CCCP algorithm illustrated for Convex minus Convex. We want to minimize the function in the Left Panel. We decompose it (Right Panel) into a convex part (top curve) minus a convex term (bottom curve). The algorithm iterates by matching points on the two curves which have the same tangent vectors, see text for more details. The algorithm rapidly converges to the solution at $x = 5.0$. (Figure concept courtesy of James M. Coughlan).

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 1 can then be applied to the variables on this hyperplane.

Theorem 2. *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:*

$$\vec{\nabla} E_{vex}(\vec{z}^{t+1}) = -\vec{\nabla} E_{cave}(\vec{z}^t) - \sum_{\mu=1}^{k} \alpha^\mu \vec{\phi}^\mu, \tag{13}$$

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 an extremum 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:

$$\hat{E}_{cave}(\vec{y}(\vec{z})) = E_{cave}(\vec{z}), \quad \hat{E}_{vex}(\vec{y}(\vec{z})) = E_{vex}(\vec{z}). \tag{14}$$

Then we can use the algorithm of Theorem 1 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:

$$\vec{\psi}^\nu \cdot \vec{\nabla}_{\vec{z}} E_{vex}(\vec{z}^{t+1}) = -\vec{\psi}^\nu \cdot \vec{\nabla}_{\vec{z}} E_{cave}(\vec{z}^t), \quad \nu = 1, ..., n-k. \tag{15}$$

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

It follows from Theorem 2 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

$$\bar{E}_{vex}(\vec{z}^{t+1}) = E_{vex}(\vec{z}^{t+1}) + \sum_\mu \alpha^\mu \{\vec{\phi}^\mu \cdot \vec{z}^{t+1} - c^\mu\}, \tag{16}$$

and use update equations:

$$\frac{\partial \bar{E}_{vex}}{\partial \vec{z}}(\vec{z}^{t+1}) = -\frac{\partial E_{cave}}{\partial \vec{z}}(\vec{z}^t), \tag{17}$$

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 need an additional result to determine how to solve for $\vec{z}^{t+1}$. Theorem 3 gives us a procedure which will work for the Bethe and Kikuchi energy functions. (In general, solving for $\vec{z}^{t+1}$ is, at worse, a convex minimization problem and can sometimes be done analytically (Yuille and Rangarajan 2001)).

We 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 2 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 2, see equation (13), *can be expressed as selecting $\vec{z}^{t+1}$ to minimize a convex cost function $E^{t+1}(\vec{z}^{t+1})$.* Moreover, we can write the solution as an analytic function of Lagrange multipliers $\{\alpha^\mu\}$ which are chosen to maximize a concave (dual) energy function. More formally, we have the following result:

Theorem 3. *Let $E_{vex}(\vec{z}) = \sum_i z_i \log \frac{z_i}{\zeta_i}$, then the update equation of Theorem 2 can be expressed as minimizing the convex energy function:*

$$E^{t+1}(\vec{z}^{t+1}) = \vec{z}^{t+1} \cdot \vec{h} + \sum_i z_i^{t+1} \log \frac{z_i^{t+1}}{\zeta_i} + \sum_\mu \alpha^\mu \{\vec{\phi}^\mu \cdot \vec{z}^{t+1} - c^\mu\}, \tag{18}$$

where $\vec{h} = \vec{\nabla} E_{cave}(\vec{z}^t)$. The solution is of form:

$$z_i^{t+1}(\alpha) = \zeta_i e^{-h_i} e^{-1} e^{-\sum_\mu \alpha^\mu \phi_i^\mu}, \tag{19}$$

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

$$\hat{E}^{t+1}(\alpha) = -\sum_i z_i^{t+1}(\alpha) - \sum_\mu \alpha^\mu c^\mu = -\sum_i \zeta_i e^{-h_i} e^{-1} e^{-\sum_\nu \phi_i^\nu \alpha^\nu} - \sum_\mu \alpha^\mu c^\mu. \quad (20)$$

.

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:*

$$1 + \log \frac{z_i}{\zeta_i} = -h_i - \sum_\mu \alpha^\mu \phi_i^\mu, \quad (21)$$

which corresponds to the update equation (13) of Theorem 2. Substituting $\vec{z}^{t+1}(\{\alpha^\mu\})$ into $E^{t+1}(\vec{z}^{t+1})$ gives the dual energy function $\hat{E}^{t+1}(\alpha) = E^{t+1}(\vec{z}^{t+1}(\alpha^\mu)$. Since $E^{t+1}(\vec{z})$ is convex, duality ensures that the dual $\hat{E}^{t+1}(\alpha)$ is concave (Strang 1986), and hence has a unique maximum which corresponds to the constraints being satisfied. 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.

Theorem 3 specifies a *double-loop* algorithm where the outer loop is given by equation (19) and the inner loop determines the $\{\alpha^\mu\}$ by maximizing $\hat{E}^{t+1}(\alpha)$.

For the Bethe free energy, solving for the $\{\alpha^\mu\}$ can be done by a discrete iterative algorithms. The algorithm maximizes $\hat{E}^{t+1}(\alpha)$ with respect to each α^μ in turn. This maximization can be done analytically for each α^μ. This inner loop algorithm generalizes work by Kosowsky and Yuille (Kosowsky and Yuille 1994, Kosowsky 1995) who used a result similar to Theorem 3 to obtain an algorithm for solving the linear assignment problem. This relates to the Sinkhorn algorithm (Sinkhorn 1964) (which converts positive matrices into doubly stochastic ones.) Rangarajan *et al* (1996a) 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 2 and 3 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 2 can be done efficiently.

4 A CCCP algorithm 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 1,2 and 3. This CCCP *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* (Rangarajan *et al* 1996a, Rangarajan *et al* 1996b).

First we split the Bethe free energy in two parts:

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

$$E_{cave} = -\sum_i n_i \sum_{x_i} b_i(x_i) \log \frac{b_i(x_i)}{\psi_i(x_i)}. \tag{22}$$

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 2 and 3. To do this, we set $\vec{z} = (b_{ij}(x_i, x_j), b_i(x_i))$ so that the first components of $\vec{z}$ correspond to the $\{b_{ij}\}$ and the later to the $\{b_i\}$ (There are NM components for the $\{b_i\}$, but the number of $\{b_{ij}\}$ depends on the number of connections and is, at most, $\frac{N(N-1)}{2}M^2$). 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:

$$\vec{\phi}^{pq} = (\delta_{pi}\delta_{qj}, 0), \quad \alpha^{pq} = \gamma_{pq}, \quad c^{pq} = 1, \ \forall p, q. \tag{23}$$

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:

$$\begin{aligned} \vec{\phi}^{pqx_q} &= (\delta_{ip}\delta_{jq}\delta_{x_j,x_q}, -\delta_{iq}\delta_{x_i,x_q}), \ \alpha^{pqx_q} = \lambda_{pq}(x_q), \ c^{pqx_q} = 0, \ \forall p, q, x_q \\ \vec{\phi}^{qpx_p} &= (\delta_{ip}\delta_{jq}\delta_{x_i x_p}, -\delta_{ip}\delta_{x_i,x_p}), \ \alpha^{qpx_p} = \lambda_{qp}(x_p), \ c^{qpx_p} = 0, \ \forall q, p, x_p. \end{aligned} \tag{24}$$

We now apply Theorem 3 to the Bethe free energy and obtain Theorem 4 which specifies the outer loop of our algorithm:

Theorem 4 (CCCP Outer Loop for Bethe). *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):*

$$\begin{aligned} b_{ij}(x_i, x_j; t+1) &= \phi_{ij}(x_i, x_j) e^{-1} e^{-\lambda_{ij}(x_j)} e^{-\lambda_{ji}(x_i)} e^{-\gamma_{ij}}, \\ b_i(x_i; t+1) &= \psi_i(x_i) e^{-1} e^{n_i} (\frac{b_i(x_i; t)}{\psi_i(x_i)})^{n_i} e^{\sum_k \lambda_{ki}(x_i)}. \end{aligned} \tag{25}$$

Proof. *These equations correspond to equation (19) in Theorem 3, where we have substituted equation (22) into equation (13) and used equations (23,24) for the constraints. The constraint terms* $\sum_\mu \alpha^\mu \vec{\phi}^\mu$ *simplify owing to the form of the constraints (e.g.* $\sum_{p,q:p>q} \gamma_{pq}\delta_{ip}\delta_{jq} = \gamma_{ij}$*).*

We need an inner loop to determine the constraint coefficients $\{\lambda_{ij}, \lambda_{ji}, \gamma_{ij}\}$. This is obtained by using Theorem 3. 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.* As we will see in section (4.1), this inner loop is very similar to BP (provided we equate the messages with the exponentials of the Lagrange multipliers).

More formally we specify the inner loop by Theorem 5:

Theorem 5 (CCCP Inner Loop for Bethe). *The constraint coefficients* $\{\gamma_{pq}\}, \{\lambda_{pq}\}, \{\lambda_{qp}\}$ *of Theorem 4 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:*

$$\begin{aligned}
e^{\gamma_{pq}(\tau+1)} &= \sum_{x_p,x_q} \phi_{pq}(x_p,x_q) e^{-1} e^{-\lambda_{pq}(x_q,\tau)} e^{-\lambda_{qp}(x_p;\tau)}, \\
e^{2\lambda_{pq}(x_q;\tau+1)} &= \frac{\sum_{x_p} \phi_{pq}(x_p,x_q) e^{-\lambda_{qp}(x_p;\tau)} e^{-\gamma_{pq}(\tau)}}{\psi_q(x_q) e^{n_q} (\frac{b_q(x_q;t)}{\psi_q(x_q;t)})^{n_q} e^{\sum_{j\neq p} \lambda_{jq}(x_q;\tau)}}, \\
e^{2\lambda_{qp}(x_p;\tau+1)} &= \frac{\sum_{x_q} \phi_{pq}(x_p,x_q) e^{-\lambda_{pq}(x_q;\tau)} e^{-\gamma_{pq}(\tau)}}{\psi_p(x_p) e^{n_p} (\frac{b_p(x_p;t)}{\psi_p(x_p;t)})^{n_p} e^{\sum_{j\neq q} \lambda_{jp}(x_p;\tau)}}
\end{aligned} \qquad (26)$$

which monotically maximizes the dual energy:

$$\begin{aligned}
\hat{E}^{t+1}(\alpha) = &- \sum_{i,j:i>j} \sum_{x_i,x_j} \phi_{ij}(x_i,x_j) e^{-1} e^{-\lambda_{ij}(x_j)} e^{-\lambda_{ji}(x_i)} e^{-\gamma_{ij}} \\
&- \sum_i \sum_{x_i} \psi_i(x_i) e^{-1} e^{n_i} (\frac{b_i(x_i;t)}{\psi_i(x_i)})^{n_i} e^{\sum_k \lambda_{ki}(x_i)} - \sum_{i,j:i>j} \gamma_{ij}.
\end{aligned} \qquad (27)$$

Proof. *We use the update rule, equation (19), given by Theorem 3 and calculate the constraint equations,* $\vec{z} \cdot \vec{\phi}^{\mu} = c^{\mu}, \ \forall \ \mu$. *For the Bethe free energy we obtain equations (26) where the upper equation corresponds to the normalization constraints and the lower equations to the consistency constraints. 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 3, this is equivalent to maximizing the dual energy, see equation (20), 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 (26). The form of the dual energy is given by substituting equation (25) into equation (20).*

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 26) 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 26) 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 $i \neq q \neq j$.)

4.1 Connections between CCCP and BP

As we will show, CCCP and BP are fairly similar. The main difference is that the CCCP update rules for the Lagrange multipliers $\{\lambda, \gamma\}$ depend *explicitly* on the current estimates of the beliefs. The estimates of the beliefs must then be re-estimated periodically (after each run of the inner loop). By contrast, for BP the update rules for the messages are independent of the beliefs.

We now give an alternative formulation of the CCCP algorithm which simplifies the algorithm (for implementation) and makes the connections to belief propagation more apparent.

Define new variables $\{h_{pq}, h_p, g_{pq}, g_p\}$ by:

$$h_{pq}(x_p, x_q) = \phi_{pq}(x_p, x_q)e^{-1}, \quad g_{pq}(\lambda, \gamma) = e^{-\gamma_{pq} - \lambda_{pq}(x_q) - \lambda_{qp}(x_p)}$$
$$h_q(x_q) = \psi_q(x_q)e^{-1}e^{n_q}\{\frac{b_q(x_q)}{\psi_q(x_q)}\}^{n_q}, \quad g_q(\lambda) = e^{\sum_j \lambda_{jq}(x_q)}. \tag{28}$$

The outer loop can be written as:

$$b_{ij}(x_i, x_j) = h_{ij}(x_i, x_j)g_{ij}(\lambda, \gamma) \;\; b_i(x_i) = h_i(x_i)g_i(\lambda). \tag{29}$$

The inner loop can be written as:

$$e^{\gamma_{pq}(\tau+1)} = e^{\gamma_{pq}(\tau)} \sum_{x_p, x_q} h_{pq}(x_p, x_q)g_{pq}(\lambda, \gamma),$$
$$e^{2\lambda_{pq}(x_q;\tau+1)} = e^{2\lambda_{pq}(x_q;\tau)} \frac{\sum_{x_p} h_{pq}(x_p, x_q)g_{pq}(\lambda, \gamma)}{h_q(x_q)g_q(\lambda)}. \tag{30}$$

This formulation of the algorithm is easier to program. As before, the inner loop is iterated until convergence and then we perform one step of the outer loop, and repeat.

We now show a formal similarity between CCCP and belief propagation. To do this, we *collapse the outer loop* of CCCP by solving for the $\{b_{ij}, b_i\}$ as functions of the variables $\{\lambda, \gamma\}$. This is done by solving the fixed point equation (29) using equation (28) to obtain $\{b_i^*, b_{ij}^*\}$ given by:

$$b_i^*(x_i) = \psi_i(x_i)e^{-1}\{g_i(\lambda)\}^{1/(n_i - 1)}, \quad b_{ij}^*(x_i, x_j) = \phi_{ij}(x_i, x_j)e^{-1}g_{ij}(\lambda, \gamma). \tag{31}$$

Now substitute $\{b_i^*, b_{ij}^*\}$ into the inner loop update equation (30). This *collapsed CCCP* algorithm reduces to:

$$e^{2\lambda_{pq}(x_q;\tau+1)} \times \{e^{-\lambda_{pq}(x_q;\tau)} e^{-\frac{1}{n_q-1}\sum_j \lambda_{jq}(x_q;\tau)}\} = \sum_{x_p} \psi_{pq}(x_p, x_q)\psi_p(x_p) e^{-\lambda_{qp}(x_p;\tau)} e^{-\gamma_{pq}}. \tag{32}$$

To relate this to belief propagation we recall from section (2) that the messages can be related to the Lagrange multipliers by:

$$m_{ji}(x_i) = e^{\lambda_{ji}(x_i)} e^{-\frac{1}{n_i-1}\sum_k \lambda_{ki}(x_i)}, \quad e^{-\lambda_{ij}(x_j)} = \prod_{k \neq i} m_{kj}(x_j). \tag{33}$$

We can re-express belief propagation as:

$$e^{\lambda_{ij}(x_j;t+1)} \quad e^{-\frac{1}{n_j-1}\sum_k \lambda_{kj}(x_j;t+1)} = c \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_i(x_i) e^{-\lambda_{ji}(x_i;t)}, \tag{34}$$

$$b_i(x_i;t) \quad = \hat{c}\psi_i(x_i) e^{-\frac{1}{n_i-1}\sum_l \lambda_{li}(x_i;t)}, \tag{35}$$

This shows that belief propagation is similar to the collapsed CCCP with the *only difference* that the factor $\{e^{-\lambda_{pq}(x_q;\tau)} e^{-\frac{1}{n_q-1}\sum_j \lambda_{jq}(x_q;\tau)}\}$ is evaluated at time τ for the collapsed double loop and at time $t+1$ for belief propagation.

(This derivation has ignored 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).

5 The Kikuchi Approximation

The Kikuchi free energy (Domb and Green 1972, Yedidia *et al* 2000) is a generalization of the Bethe free energy which enables us to include higher order interactions.

In this section we show that the results we obtained for the Bethe can be extended to Kikuchi. Recall that Yedidia *et al* (2000) derived a "generalized belief propagation algorithm" whose fixed points correspond to extrema of the Kikuchi free energy. Their computer simulations showed that GBP outperforms BP on 2D spin glasses and obtains results close to the optimum (presumably because Kikuchi is a higher order approximation than Bethe).

We now define the Kikuchi free energy (following the exposition in Yedidia *et al* (2000)). 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.

We illustrate these definitions on the 2D spin-glass, see figure (1). The basic regions are shown in figure (4) (other choices of basic regions are possible). The direct super-regions and the direct sub-regions are shown in figures (5,6).

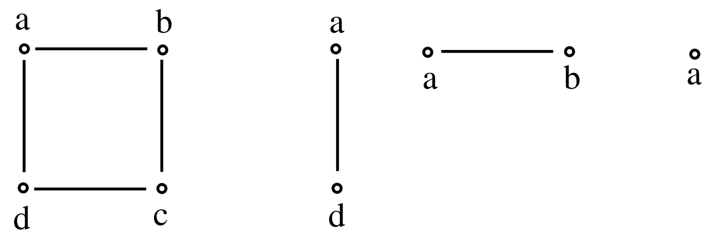

Figure 4: The regions for the Kikuchi implementation of the 2D spin-glass. The top regions (left), and the subregions (centre and right).

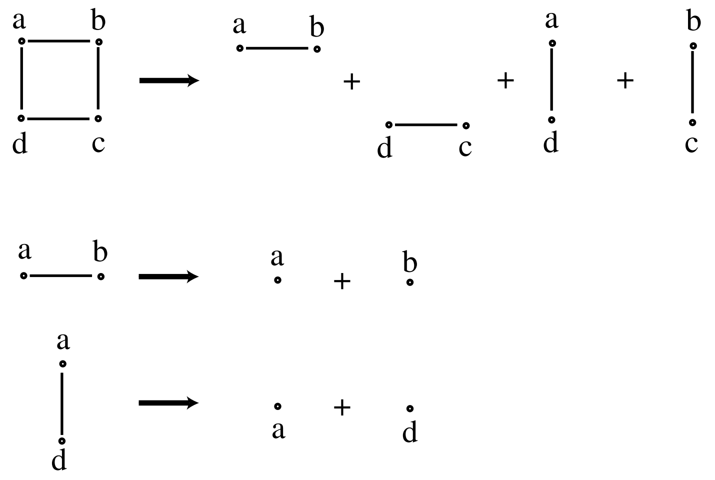

Figure 5: The direct sub-regions. The top-level regions (Top Panel) have four direct sub-regions (two horizontal and two vertical). The middle-level regions each have two direct sub-regions (Bottom Two Panels).

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:

$$F_K = \sum_{r \in R} c_r \{ \sum_{x_r} b_r(x_r) E_r(x_r) + \sum_{x_r} b_r(x_r) \log b_r(x_r) \} + L_K, \qquad (36)$$

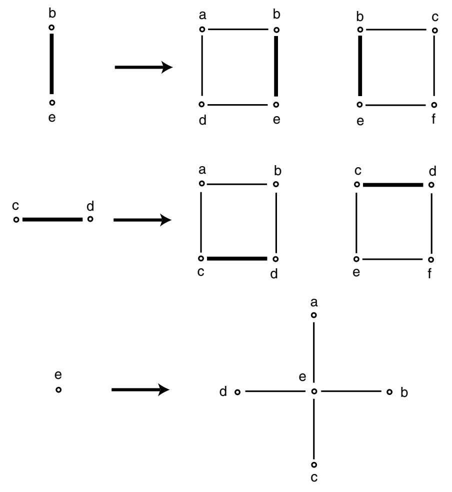

Figure 6: The direct super-regions. The middle-level regions have two direct super-regions (Top Two Panels). The bottom-level region have four direct super-regions (Bottom Panel).

where L_K gives the constraints (see equation (37) 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 $c_r = 1$. We let $c_{max} = \max_{r \in R} c_r$.

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:

$$L_K = \sum_{r \in R} \gamma_r (\sum_{x_r} b_r(x_r) - 1) + \sum_{r \in R} \sum_{s \in sub_d(r)} \sum_{x_s} \lambda_{r,s}(x_s) \{ \sum_{x \in r \setminus s} b_r(x_r) - b_s(x_s) \}. \quad (37)$$

5.1 A CCCP Algorithm for the Kikuchi Free Energy

Now we obtain a CCCP algorithm guaranteed to converge to an extrema of the Kikuchi free energy.

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):

$$F_K = F_{K,vex} + F_{K,cave}, \quad (38)$$

where

$$\begin{aligned} F_{K,vex} &= c_{max} \sum_{r \in R} \{ \sum_{x_r} b_r(x_r) E_r(x_r) + \sum_{x_r} b_r(x_r) \log b_r(x_r) \} \quad (39) \\ &+ \sum_{r \in R} \gamma_r (\sum_{x_r} b_r(x_r) - 1) + \sum_{r \in R} \sum_{s \in sub_d(r)} \sum_{x_s} \lambda_{r,s}(x_s) \{ \sum_{x \in r \setminus s} b_r(x_r) - b_s(x_s) \} \end{aligned}$$

$$F_{K,cave} = \sum_{r \in R} (c_r - c_{max}) \{ \sum_{x_r} b_r(x_r) E_r(x_r) + \sum_{x_r} b_r(x_r) \log b_r(x_r) \}, \quad (40)$$

where we have used the results stated in equations (16,17) 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. Recall that $c_{max} = max_{r \in R} c_r$. This ensures that $F_{K,cave}$ is concave.

Theorem 6 (CCCP for Kikuchi). *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:*

$$b_r(x_r; t+1) = e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}} e^{-\gamma_r} \{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\sum_{s \in sub_d(r)} \lambda_{r,s}(x_s)} e^{\sum_{v \in sup_d(r)} \lambda_{v,r}(x_r)}. \quad (41)$$

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

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

$$e^{2\lambda_{r,u}(x_u;\tau+1)} = \frac{\sum_{x \in r \backslash u} e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}} \{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\gamma_r(\tau)} e^{-\sum_{s \in sub_d(r): s \neq u} \lambda_{r,s}(x_s;\tau)} e^{\sum_{v \in sup_d(r)} \lambda_{v,r}(x_r;\tau)}}{e^{-\frac{c_u}{c_{max}}\{E_u(x_u)+1\}} \{b_u(x_u;t)\}^{\frac{c_{max}-c_u}{c_{max}}} e^{-\gamma_u(\tau)} e^{-\sum_{s \in sub_d(u)} \lambda_{u,s}(x_s;\tau)} e^{\sum_{v \in sup_d(u): v \neq r} \lambda_{v,u}(x_u;\tau)}}. \quad (42)$$

Moreover, the inner loop is guaranteed to satisfy the constraints by converging to the unique maximum of the dual energy:

$$\hat{E}^{t+1}(\lambda, \gamma) = - \sum_{r \in R} \sum_{x_r} e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}} e^{-\gamma_r} \{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\sum_{s \in sub_d(r)} \lambda_{r,s}(x_s)} e^{\sum_{v \in sup_d(r)} \lambda_{v,r}(x_r)} - \sum_{r \in R} \gamma_r. \quad (43)$$

Proof. *We split the Kikuchi free energy F_K into the convex and concave parts, $F_{K,vex}, F_{K,cave}$, given by equation (40). We can use Theorems 1,2 and 3 to obtain an algorithm to minimize the Kikuchi free energy.*

In more detail, we calculate the derivatives:

$$\frac{\partial F_{K,vex}}{\partial b_r(x_r)} = c_{max}\{E_r(x_r) + 1 + \log b_r(x_r) + \gamma_r + \sum_{s \in sub_d(r)} \lambda_{r,s}(x_s) - \sum_{v \in sup_d(r)} \lambda_{v,r}(x_r)\}, \quad (44)$$

$$\frac{\partial F_{K,cave}}{\partial b_r(x_r)} = (c_r - c_{max})\{E_r(x_r) + 1 + \log b_r(x_r)\}, \quad (45)$$

where we have not included 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 1,2,3 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 (44,45), we can obtain the outer loop of our update algorithm (using Theorems 2,3):

$$b_r(x_r;t{+}1) = e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\gamma_r} e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)} e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{46}$$

The inner loop, to solve for the $\{\gamma_r\}, \{\lambda_{r,s}\}$, is determined by Theorem 2. 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:

$$e^{\gamma_r} = \sum_{x_r} e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)} e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{47}$$

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:

$$\sum_{x\in r\setminus u} e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}}\{b_r(x_r;t)\}^{\frac{c_{max}-c_r}{c_{max}}} e^{-\gamma_r} e^{-\sum_{s\in sub_d(r)}\lambda_{r,s}(x_s)} e^{\sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}$$
$$= e^{-\frac{c_u}{c_{max}}\{E_u(x_u)+1\}}\{b_u(x_u;t)\}^{\frac{c_{max}-c_u}{c_{max}}} e^{-\gamma_u} e^{-\sum_{s\in sub_d(u)}\lambda_{u,s}(x_s)} e^{\sum_{v\in sup_d(u)}\lambda_{v,u}(x_u)}, \tag{48}$$

which can be rearranged to give an analytic solution for $\lambda_{r,u}(x_u)$, see equation (42). The dual energy is given by Theorem 3. Hence result.

5.2 CCCP Kikuchi and Generalized Belief Propagation

We now generalize our arguments for the Bethe free energy, see subsection (4.1), and show there is a connection between CCCP for Kikuchi and GBP. Once again, CCCP is very similar to message passing but it requires us to periodically update our estimates of the beliefs.

First we simplify the form of the CCCP Kikuchi. Then we briefly describe GBP. Finally we show connections between CCCP and GBP by collapsing the outer loop.

We simplify the double-loop Kikuchi algorithm by defining new variables:

$$h_r(x_r) = e^{-\frac{c_r}{c_{max}}\{E_r(x_r)+1\}}\{b_r(x_r)\}^{\frac{c_{max}-c_r}{c_{max}}},\ \ g_r(x_r) = e^{-\gamma_r - \sum_{s\in sub_d(r)}\lambda_{r,s}(x_s) + \sum_{v\in sup_d(r)}\lambda_{v,r}(x_r)}. \tag{49}$$

The outer loop rule becomes

$$b_r(x_r) = h_r(x_r) g_r(x_r). \tag{50}$$

The inner loop becomes:

$$e^{\gamma_r(\tau+1)} = e^{\gamma_r(\tau)} \sum_{x_r} h_r(x_r) g_r(x_r),$$

$$e^{2\lambda_{r,u}(x_u;\tau+1)} = e^{2\lambda_{r,u}(x_u;\tau)} \frac{\sum_{x\in r\setminus u} h_r(x_r) g_r(x_r)}{h_u(x_u) g_u(x_u)}. \tag{51}$$

This form of the update rules is straightforward to program and, as we now show, relate to the generalized belief propagation algorithm of Yedidia *et al* (2000).

We now give a formulation of generalized belief propagation. Following Yedidia *et al* (2000), we allow messages $m_{r,s}(x_s)$ between regions r and any direct sub-regions $s \in sub_d(r)$. We also define $M(r)$ to be the set of all messages that flow into r, or any of its subregions, from *outside* r. (I.e. $m_{r',s'}(x_{s'}) \in M(r)$ provided s' is a subregion of r, or r itself, and $r' \setminus s$ is outside r). The messages will correspond to (new) Lagrange multipliers $\{\mu\}$ related to the messages by $\mu_{r,s}(x_s) = \log m_{r,s}(x_s)$.

The main idea is to introduce new Lagrange multipliers $\{\mu_{r,s}\}$ so that:

$$\sum_{r\in R} \sum_{s\in sub_d(r)} \sum_{x_s} \lambda_{r,s}(x_s)\{\sum_{x\in r\setminus s} b_r(x_r) - b_s(x_s)\} = \sum_r c_r \sum_{x_r} b_r(x_r) \sum_{\{r',s'\}\in M(r)} \mu_{r',s'}(x_{s'}). \tag{52}$$

By extremizing the Kikuchi free energy, we can find the solution to be:

$$b_r(x_r) = \psi_r(x_r) \prod_{\{r',s'\}\in M(r)} m_{r',s'}(x_{s'}), \tag{53}$$

where we relate the messages to the Lagrange multipliers by $\mu_{r,s}(x_s) = \log m_{r,s}(x_s)$.

Generalized belief propagation can be reformulated as:

$$m_{r,s}(x_s; t+1) = m_{r,s}(x_s; t) \frac{\sum_{x\in r\setminus s} \psi_r(x_r) \prod_{\{r',s'\}\in M(r)} m_{r',s'}(x_{s'})}{\psi_s(x_s) \prod_{\{r',s'\}\in M(s)} m_{r',s'}(x_{s'})}. \tag{54}$$

It is clear that fixed points of this equation will correspond to situations where the constraints are satisfied (because the numerator will equal $\sum_{x\in r/s} b_r(x_r)$ and the denominator is $b_s(x_s)$). The form of $b_r(x_r)$ means that this is all we need to do.

To relate this to the Kikuchi double-loop, we once again collapse the outer loop by solving for the $\{b_r\}$ in terms of the Lagrange multipliers $\{\lambda\}$, this gives:

$$b_r^*(x_r) = e^{-E_r(x_r)-1}\{g_r(\lambda, \gamma)\}^{c_{max}/c_r}. \tag{55}$$

These are precisely the same form as those given by generalized belief propagation after we solve for the relationship between the Lagrange multipliers to be:

$$\sum_{s\in sub_d(r)} \lambda_{r,s}(x_s) - \sum_{p\in sup_d(r)} \lambda_{p,r}(x_r) = c_r \sum_{\{r'.s'\}\in M(r)} \mu_{r',s'}(x_{s'}). \tag{56}$$

The updates are then given by:

$$e^{\gamma_r(\tau+1)} = e^{\gamma_r(\tau)} \sum_{x_r} b_r^*(x_r),$$

$$e^{2\lambda_{r,u}(x_u;\tau+1)} = e^{2\lambda_{r,u}(x_u;\tau)} \frac{\sum_{x\in r\setminus s} b_r^*(x_r)}{b_u^*(x_u)}. \tag{57}$$

So these are very similar to generalized belief propagation, using equations (53,54).

The only difference is that we are updating the $\{\lambda\}$ instead of the $\{\mu\}$, which are related by a linear transformation.

6 Implementation

We implemented the Bethe and Kikuchi CCCP algorithms in Numeric Python on spin glass energy functions. We used binary state variables defined on two-dimensional and three-dimensional lattices (using toroidal boundary conditions). This is similar to the setup used in (Yedidia *et al* 2000, Teh and Welling 2001). We also implemented BP and GBP for comparison. (These examples were chosen because it is known that BP has difficulties with graphs with so many closed loops).

The grid was shown in figure (1) (imagine a third dimension to get the 3-D cube). We label the nodes by $\vec{i}$ where each component of $\vec{i}$ takes values from 1 to N. We let $\vec{i} \pm \Delta\vec{h}$ and $\vec{i} \pm \Delta\vec{v}$ be the horizontal and vertical neighbours of site $\vec{i}$. Each site has a spin state $x_{\vec{i}} \in \{0,1\}$.

There are (unary) potentials at each lattice site labelled by $\psi_{\vec{i}}(x_{\vec{i}})$. There are horizontal and vertical connections between neighbouring lattice sites which we label as $\psi_{\vec{i},\vec{i}+\Delta\vec{h}}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{h}})$ and $\psi_{\vec{i},\vec{i}+\Delta\vec{v}}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{v}})$ respectively (ie. linking node a to b and a to c in figure (1)). In sum, we have N^2 unary potentials $\psi_{\vec{i}}$, N^2 horizontal pairwise potentials $\psi_{\vec{i},\vec{i}+\Delta\vec{h}}$, and N^2 vertical pairwise potentials $\psi_{\vec{i},\vec{i}+\Delta\vec{v}}$. The size of N was varied from 10 to 100.

The $\psi_{\vec{i}}(x_{\vec{i}})$ are of form $e^{\pm h_{\vec{i}}}$ where the $h_{\vec{i}}$ are random samples from a Gaussian distribution $N(\mu,\sigma)$.. The horizontal potentials are of form $e^{\pm h_{\vec{i},\vec{i}+\Delta\vec{h}}}$ where the $h_{\vec{i},\vec{i}+\Delta\vec{h}}$ are random samples from a Gaussian $N(\mu,\sigma)$. The vertical potentials are generated in a similar manner. The parameters of these three Gaussians were varied (typically we set $\mu = 0$ and $\sigma \in \{1,2,5\}$).

6.1 Bethe Implementations

We derived the Bethe free energy for the problem specified above and implemented the CCCP and the BP algorithm. We use randomized initial conditions on the messages or the Lagrange multipliers (as appropriate).

We used unary beliefs $b_{\vec{i}}(x_{\vec{i}})$ and horizontal joint beliefs $b_{\vec{i},\vec{i}+\Delta h}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{h}})$ and vertical joint beliefs $b_{\vec{i},\vec{i}+\Delta v}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{v}})$.

For CCCP, we have Lagrange parameter terms $\lambda_{\vec{i}\pm\Delta\vec{v},\vec{i}}(x_{\vec{i}})$ and $\lambda_{\vec{i}\pm\Delta\vec{h},\vec{i}}(x_{\vec{i}})$. These are analogous to the BP messages terms $m_{\vec{i}\pm\Delta\vec{v},\vec{i}}(x_{\vec{i}})$ and $m_{\vec{i}\pm\Delta\vec{h},\vec{i}}(x_{\vec{i}})$. They can be thought of as the messages that flow into node $\vec{i}$, see figure (2). For CCCP we also have normalization terms $\gamma_{\vec{i},\vec{i}+\Delta\vec{h}}$ and $\gamma_{\vec{i},\vec{i}+\Delta\vec{v}}$ to normalize the horizontal and vertical joint beliefs.

For the CCCP algorithm we used five iterations of the inner-loop as a default. This gave stable convergence although it did not always ensure that the constraints were satisfied. We experimented with fewer iterations. The algorithm was unstable if only one inner loop iteration was used. But it already showed stability when we used two iterations.

The BP algorithm was implemented in the standard manner. We used a parallel update rule.

For both algorithms we measured convergence by the Bethe free energy or by the "Belief Difference": the differences in the belief estimates between adjacent iterations. Note that the Bethe free energy is not very meaningful for the BP algorithm until the algorithm has converged (because only then will the constraints be satisfied).

The CCCP algorithm converged rapidly with typically less than six iterations of the outer loop – see figure (7) for 2D spin-glass figure (9) for 3D spin-glass. Most of the Bethe free energy decrease happened in the first two iterations. Varying the standard deviation of the Gaussians (which generate the potential) made no difference. It also happened when we altered the size of the lattice (our default lattice size was 10 x 10 but we also explored 50 x 50 and 10 x 10 x 10 and 20 x 20 x 20. During the first six iterations the constraints were not always satisfied (recall we are using five iterations of the inner loop). See, for example, figure (9) (top right panel) where the free energy appears to increase at iteration 5. But this (temporary) non-compliance with the constraints did not seem to matter and the Bethe free energy (almost) always decreased monotonically.

The BP algorithm behaved differently as we varied the situation. It was far less stable in general. For the 2D spin-glass, the algorithm would not always converge. The convergence was better the smaller the variance of the Gaussian distributions generating the potentials. So if we set $\sigma_p = \sigma_h = \sigma_v = 1.0$ then BP would often converge. But it became far more unstable when we increased the standard deviations to $\sigma_p = \sigma_h = \sigma_v = 5.0$. BP did *not converge at all* for the 3D spin-glass case. (Note we did not try to help convergence by adding inertia terms – see Kikuchi section).

Overall, CCCP was far more stable than BP and the convergence rate was roughly as fast (taking into account the inner-loops). The Bethe free energy of the CCCP solution was always as small, or smaller, than that found by BP. The form of the solutions found by both algorithms (when BP converged) were very similar, see figure (8), though we observed that BP tended to favour beliefs biased towards 0 or 1 while CCCP often gave beliefs closer to 0.5. (By contrast, CCCP for Kikuchi did not show any tendency towards 0.5).

We conclude that on these examples CCCP outperformed BP as far as stability and quality of the results (particularly for 3D spin-glasses). These results are preliminary and more systematic experiments should be performed. But they do provide proof of concept for our approach.

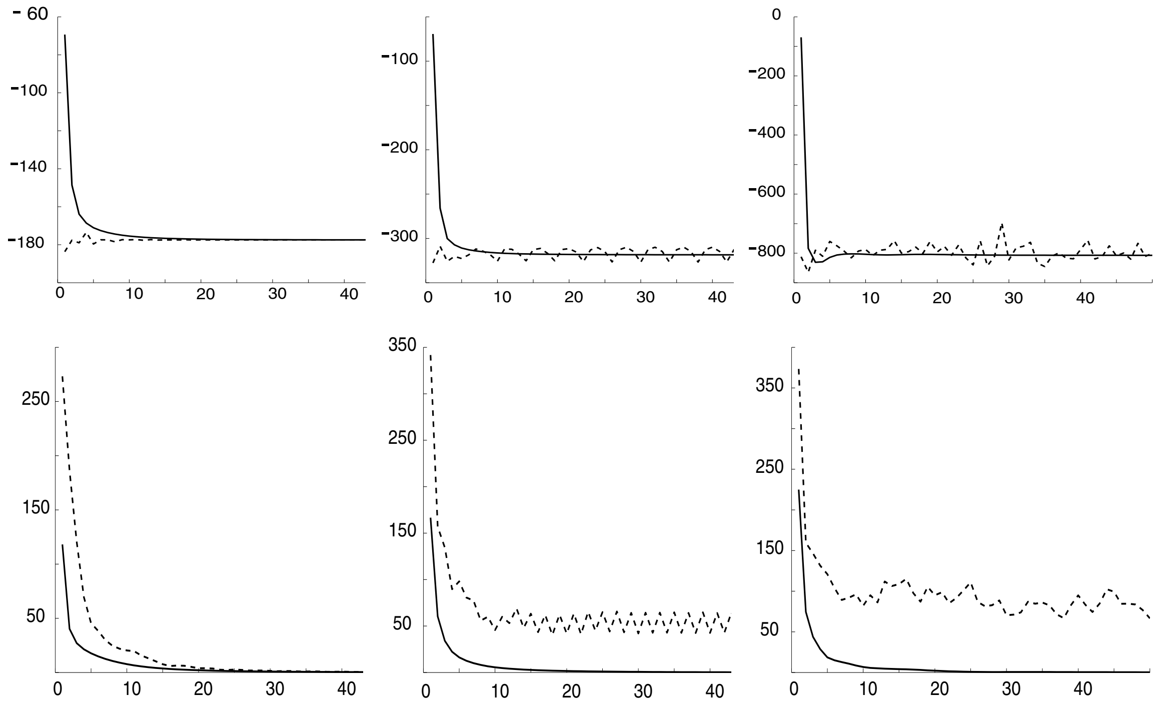

Figure 7: Performance of CCCP and BP on 2D spin-glasses. Bethe Free Energy plots (Top Panels) and Belief Difference plots (Bottom Panels). Left to Right $\sigma = 1.0, 2.0, 5.0$. CCCP solution (full lines) BP solution (dashed lines). Observe that CCCP is more reliable for $\sigma \geq 2$ (although BP often converged correctly for $\sigma = 2.0$). Each iteration of CCCP counts as five iterations of BP (we use five iterations in the inner loop). For BP the Bethe Free energy is not very meaningful until convergence.

6.2 Kikuchi Implementations

We also implemented CCCP and Generalized Belief Propagation (GBP) for the Kikuchi free energy. This was done for the 2D spin-glass case only. Once again, we found that CCCP converged very quickly (again we used five iterations of the inner loop as a default). Following Yedidia *et al* (2000), we implemented GBP with inertia (we could not get the algorithm to converge without inertia). Yedidia *et al* (2000) used an inertia factor of $\alpha = 0.5$. We were more conservative and set $\alpha = 0.9$ (which improved stability). (Inertia means that $mess_{new} = \alpha mess_{old} + (1-\alpha) mess_{update}$ where $\alpha = 1$ is standard BP/GBP).

Both algorithms converged with these settings. But CCCP converged more quickly while GBP appeared to oscillate a bit about the solution before converging. Both gave similar results for the final Kikuchi free energy, see figure (10).

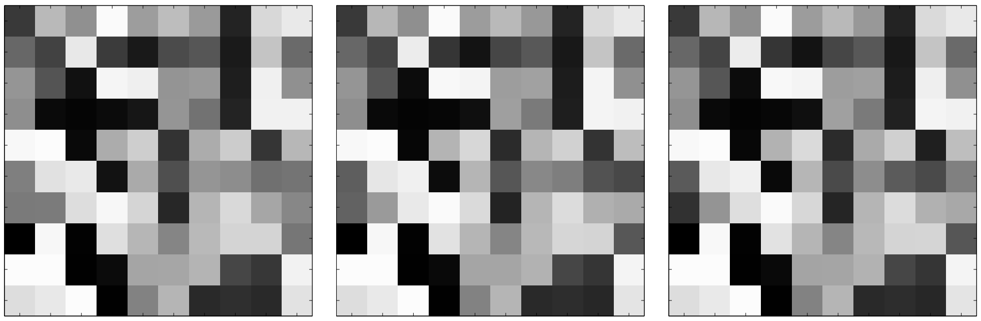

Figure 8: Bethe 2D-spin glass. CCCP solution (left). Two BP solutions with different initial conditions (centre and right). We plot the beliefs $b_i(x_i = 0)$ for all nodes i on the 2D lattice. Observe the similarly between them.

The solutions found by the two algorithms were very similar, see figure (11). Moreover, the solutions appeared to be practically independent of the starting conditions (which were randomized).

Yedidia *et al* (2000) reported that GBP gave results on 2D spin-glasses which were almost identical to the true solutions (found by using a maximum flow algorithm). This is consistent with our finding that both GBP and CCCP converged to very similar solutions despite randomized starting conditions).

We now give more details of the implementation. The top-level regions are 2x2 squares. These are labelled as $b_{\vec{i},\vec{i}+\Delta\vec{h},\vec{i}+\Delta\vec{v},\vec{i}+\vec{\Delta}\vec{h}+\vec{\Delta}\vec{v}}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{h}}, x_{\vec{i}+\Delta\vec{v}}, x_{\vec{i}+\Delta\vec{h}+\Delta\vec{v}})$. There are two "medium-level" regions (obtained by taking the intersections of adjacent top-level regions). These are horizontal $b_{\vec{i},\vec{i}+\Delta h}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{h}})$ and vertical $b_{\vec{i},\vec{i}+\Delta v}(x_{\vec{i}}, x_{\vec{i}+\Delta\vec{v}})$. There are bottom-level regions corresponding to the pixels $b_{\vec{i}}(x_{\vec{i}})$.

To implement CCCP for Kikuchi, we need to specify the direct sub-regions and the direct super-regions. It can be seen that the direct sub-regions of $b_{\vec{i},\vec{i}+\Delta\vec{h},\vec{i}+\Delta\vec{v},\vec{i}+\vec{\Delta}\vec{h}+\vec{\Delta}\vec{v}}$ are $b_{\vec{i},\vec{i}+\Delta h}$, $b_{\vec{i}+\Delta v,\vec{i}+\Delta h+\Delta v}$, $b_{\vec{i},\vec{i}+\Delta v}$ and $b_{\vec{i}+\Delta h,\vec{i}+\Delta v+\Delta h}$. The direct sub-regions of $b_{\vec{i},\vec{i}+\Delta h}$ are $b_{\vec{i}}$ and $b_{\vec{i}+\Delta\vec{h}}$. Those of $b_{\vec{i},\vec{i}+\Delta v}$ are $b_{\vec{i}}$ and $b_{\vec{i}+\Delta v}$. Of course, $b_{\vec{i}}$ has no direct (or indirect) sub-regions.

The direct super-regions of $b_{\vec{i}}$ are $b_{\vec{i},\vec{i}+\Delta h}$, $b_{\vec{i}-\Delta h,\vec{i}}$, $b_{\vec{i},\vec{i}+\Delta v}$ and $b_{\vec{i}-\Delta v,\vec{i}}$. The direct super-regions of $b_{\vec{i},\vec{i}+\Delta h}$ are $b_{\vec{i},\vec{i}+\Delta\vec{h},\vec{i}+\Delta\vec{v},\vec{i}+\vec{\Delta}\vec{h}+\vec{\Delta}\vec{v}}$ and $b_{\vec{i}-\Delta v,\vec{i}+\Delta\vec{h}-\Delta v,\vec{i},\vec{i}+\vec{\Delta}\vec{h}}$. Similarly, the direct super-regions of $b_{\vec{i},\vec{i}+\Delta v}$ are $b_{\vec{i},\vec{i}+\Delta\vec{h},\vec{i}+\Delta\vec{v},\vec{i}+\vec{\Delta}\vec{h}+\vec{\Delta}\vec{v}}$ and $b_{\vec{i}-\Delta\vec{h},\vec{i},\vec{i}+\Delta\vec{v}-\Delta\vec{h},\vec{i}+\vec{\Delta}\vec{v}}$.

7 Discussion

It is known that BP and GBP algorithms are guaranteed to converge to the correct solution on graphs with no loops. We now argue that CCCP will also converge to the correct answer on such problems.

As shown by Yedidia *et al* (2000), the fixed points of BP/GBP correspond to extrema of the Bethe/Kikuchi free energy. Hence for graphs with no loops there can

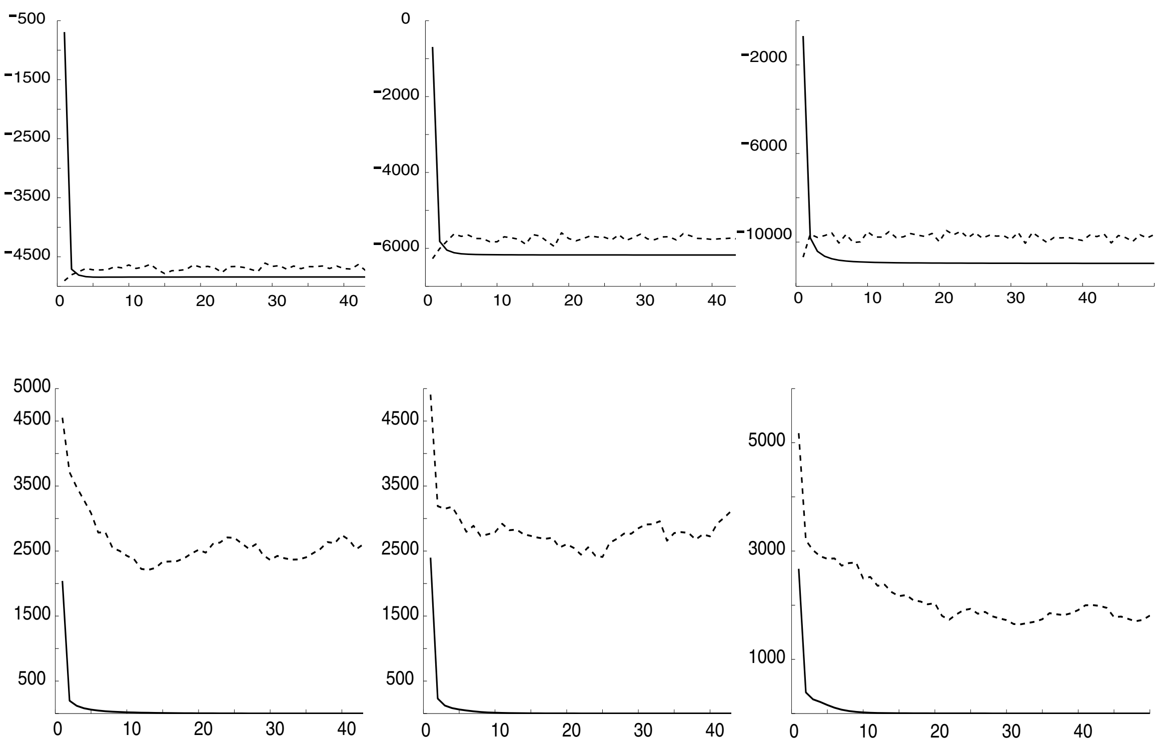

Figure 9: Performance of CCCP and BP on 3D spin-glasses. Bethe Free Energy plots (Top Panels) and Belief Difference plots (Bottom Panels). Left to Right $\sigma = 1.0, 2.0, 5.0$. CCCP solution (full lines) BP solution (dashed lines).

only be a single extremum of the Bethe/Kikuchi free energies. These free energies are bounded below and so the extremum must be a minimum. The free energy therefore have a single unique global minimum. CCCP is then guaranteed to find it.

We also emphasize the generality of the Kikuchi approximation. In this paper we considered distributions $P(x_1, ..., x_N|y)$ which had pairwise interactions only (ie. the highest order terms were of form $\psi_{ij}(x_i, x_j)$). The Kikuchi approximation, however, can be extended to interactions of arbitrary high order (eg. terms such as $\psi_{i_1,...i_N}(x_{i_1}, ..., x_{i_N})$). Good results are reported for these cases (Yedidia – private communication).

Finally we observe that both the Bethe and Kikuchi free energies can be reformulated in terms of dual variables (Strang 1986). This duality is standard for quadratic optimization problems. The BP algorithm can be expressed in terms of the dynamics in this dual space.

If we take the Bethe free energy and extremize with respect to the variables $\{b_{ij}\}, \{b_i\}$ it is possible to solve directly for these variables in terms of the La-

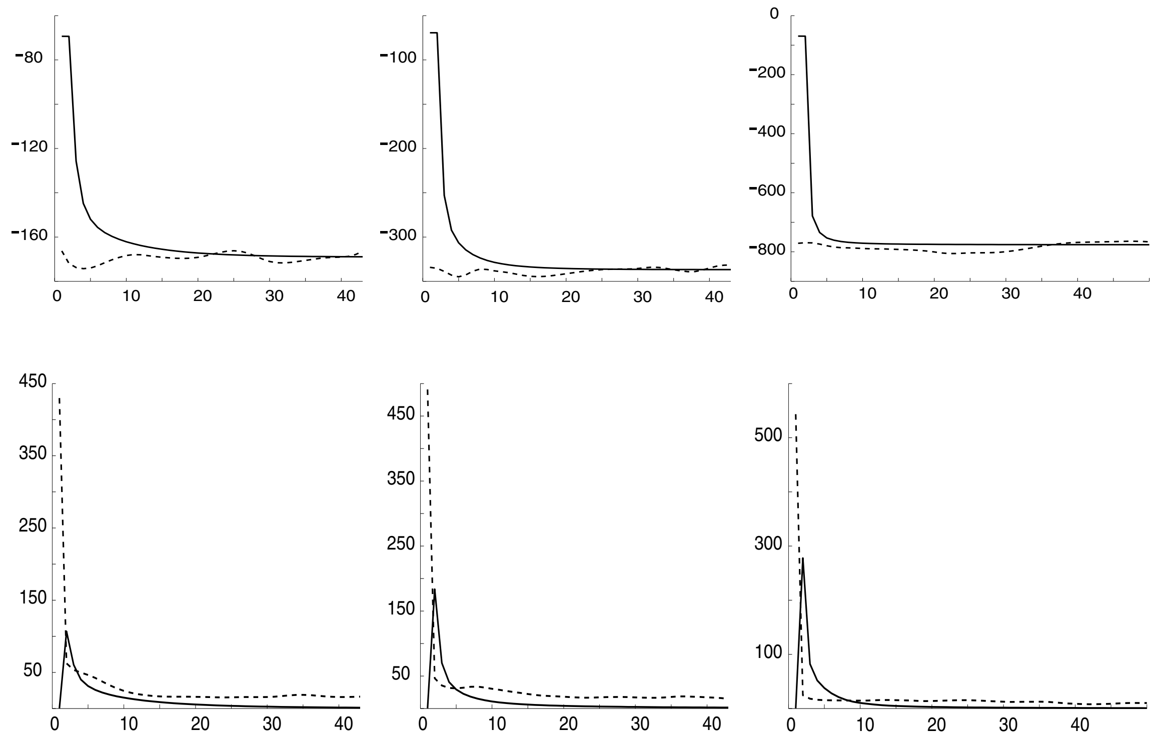

Figure 10: Performance of CCCP Kikuchi and GBP on 2D spin-glasses. Kikuchi Free Energy plots (Top Panels) and Belief Difference plots (Bottom Panels). Left to Right $\sigma = 1.0, 2.0, 5.0$. CCCP Kikuchi solution (full lines) GBP solution (dashed lines).

grange multipliers $\{\lambda_{ij}\}, \{\gamma_{ij}\}$. We can then substitute this back into the Bethe free energy to obtain the dual energy.

$$\begin{aligned} G(\{\gamma_{ij}\}, \{\lambda_{ij}\}) = &- \sum_{i,j:i>j} \sum_{x_i,x_j} \phi_{ij}(x_i, x_j) e^{-\{\lambda_{ij}(x_j)+\lambda_{ji}(x_i)\}} e^{-1} e^{-\gamma_{ij}} \\ &+ \sum_i (n_i - 1) \sum_{x_i} \psi_i(x_i) e^{-\frac{1}{n_i-1}\sum_j \lambda_{ji}(x_i)} e^{-1} - \sum_{i,j:i>j} \gamma_{ij}, \end{aligned} \tag{58}$$

BP is given by the update rule

$$\frac{\partial g}{\partial e^{-\lambda_{ji}(x_i)}}(\lambda_{ji}(x_i;t+1)) = -\frac{\partial f}{\partial e^{-\lambda_{ji}(x_i)}}(\lambda_{ji}(x_i;t)), \quad \frac{\partial g}{\partial \gamma_{ij}}(\gamma_{ij}(t+1)) = -\frac{\partial f}{\partial \gamma_{ij}}(\gamma_{ij}(t)), \tag{59}$$

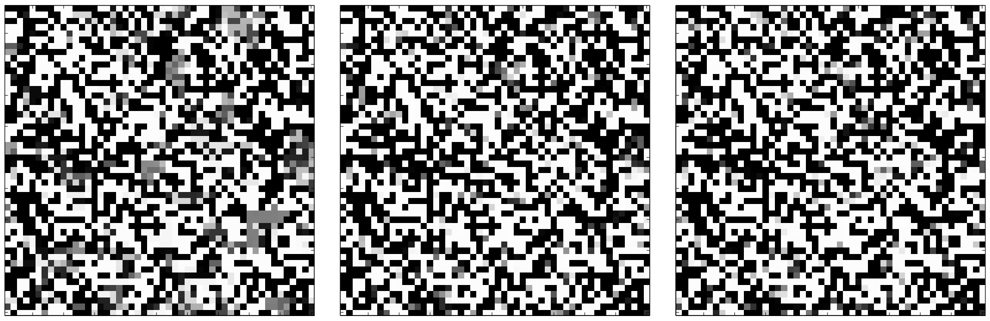

Figure 11: Kikuchi 2D-spin glass. CCCP solution (left). Two GBP solutions with different initial conditions (centre and right). We plot the beliefs $b_i(x_i = 0)$ for all nodes i on the 2D lattice. Observe the similarity of the results on this 50 x 50 grid.

where $f(.)$ and $g(.)$ are the first and second terms in G.

By standard properties of duality, extrema of the dual energy G correspond to extrema of the Bethe free energy. Unfortunately minima do not correspond to maxima (which they would if the Bethe free energy was convex). This means that it is hard to obtain global convergence results by analyzing the dual energy.

Similarly, we can calculate the dual of the Kikuchi free energy. This is of form.

$$G_K(\{\gamma_r\}, \{\lambda_{r,s}\}) = -\sum_{r\in R} c_r \sum_{x_r} e^{-1} e^{-E_r(x_r)} e^{-\gamma_r/c_r} e^{-(1/c_r)\sum_{s\in sub_d(r)} \lambda_{r,s}(x_s)} e^{(1/c_r)\sum_{v\in sup_d(r)} \lambda_{v,r}(x_r)} - \sum_{r\in R} \gamma_r. \quad (60)$$

8 Conclusion

This paper introduced CCCP double loop algorithms which are proved to converge to extrema of the Bethe and Kikuchi free energies. They are convergent alternatives to BP and GBP (whose main error mode is failure to converge).

We showed that there are similarities between CCCP and BP/GBP. More precisely, the CCCP algorithms updates Lagrange parameter variables $\{\lambda, \gamma\}$ while BP/GBP updates messages $\{m\}$ – but the exponentials of the $\{\lambda\}$ correspond to linear combinations of the messages (and the $\{\gamma\}$ are implicit in BP/GBP). In BP/GBP the beliefs $\{b\}$ are expressed in terms of the messages $\{m\}$ but for CCCP they are expressed in terms of the $\{\lambda, \gamma\}$. The difference is that for BP/GBP the update rules for the $\{m\}$ do not explicitly depend on the current estimation of the $\{b\}$. But the CCCP updates for $\{\lambda, \gamma\}$ do depend on the current $\{b\}$ and the estimates of the $\{b\}$ must be periodically re-estimated. We can make BP/GBP and CCCP very similar by expressing the $\{b\}$ as a function of the $\{\lambda, \gamma\}$ (this is collapsing the double loop) which makes the algorithms very similar (but which prevents the convergence proof from holding).

Our computer simulations on spin glasses showed that the CCCP algorithms are stable, converge rapidly, and give solutions as good, or better than, those found by BP/GBP. (Convergence rates of CCCP were similar to those of BP/GBP when five iterations of the inner loop were used). In particular, we found that BP often did not converge on 2D spin glasses and never converged at all on 3D spin glasses. GBP, however, did converge on 2D spin-glasses provided inertia was used.

Finally, we stress that the Bethe and Kikuchi free energies are variational techniques that can be applied to a range of inference and learning applications (Yedidia *et al* 2000). They are better approximations that standard mean field theory and hence give better results on certain applications (Weiss 2001). Mean field theory algorithms can be very effective on difficult optimization problems (e.g., see Rangarajan *et al* 1996a). But it is highly probable that using Bethe and Kikuchi approximations (and applying CCCP – or BP/GBP – algorithms) will perform well on a large range of applications.

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 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 $\{c_r\}$. Two referees gave helpful feedback. This work was supported by the National Institute of Health (NEI) with grant number RO1-EY 12691-01. It was also supported by the National Science Foundation with award number IRI-9700446.

References

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

- Berrou, C., Glavieux, A. and Thitimajshima, P. 1993. "Near Shannon Limit Error-correcting Coding and Decoding: Turbo codes (I)." *Proc. ICC'93* (Geneva), pp 1064-1070.

- Chiang, M. and Forney, Jr., C.D. 2001. "Statistical Physics, Convex Optimization and the Sum Product Algorithm". LIDS Technical Report. Laboratory for Information and Decision Systems. MIT.

- Cover, T.M. and Thomas, J.A. 1991 **Elements of Information Theory**. Wiley Interscience Press: New York.

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

- Forney, Jr., C.D. 2001. "Codes on Graphs: News and Views". *2001 Conference on Information Sciences and Systems.* The John Hopkins University, March 21-23.

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

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

- Gallager, R.G. 1963. **Low-density parity-check codes**. Cambridge: MA: MIT Press.

- Hertz, J., Krogh, A. and Palmer, R.G. 1991. **Introduction to the Theory of Neural Computation**. Addison Wesley. Reading, Mass.

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

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

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

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

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

- Pearl, J. 1988. **Probabilistic Reasoning in Intelligent Systems**. Morgan Kaufmann.

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

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

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

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

- Teh, Y.W. and Welling, M. 2001. "Passing and Bouncing Messages for Generalized Inference". GCTU TR 2001-001. Gatsby Computational Neuroscience Unit. University College London.

- Wainwright, M., Jaakkola, T. and Willsky, A. 2001. "Tree-based Reparameterization Framework for Approximate Estimation of Stochastic Processes on Graphs with Cycles". LIDS TR P-2510. Laboratory for Information and Decision Systems. MIT.

- Waugh, F.R. and Westervelt, R.M. 1993. "Analog neural networks with local competition: I. Dynamics and stability". *Physical Review E*, 47(6), pp 4524-4536.
- Weiss, Y. 2001. "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.
- Yedidia, J.S., Freeman, W.T. and Weiss, Y. 2000. "Bethe free energy, Kikuchi approximations and belief propagation algorithms". To appear in NIPS'2000.
- Yuille, A.L. and Kosowsky, J.J. 1994. "Statistical Physics Algorithms that Converge." Neural Computation. **6**, pp 341-356.
- Yuille, A.L. and Rangarajan, A. 2001. "The Concave Convex Principle (CCCP)". To appear in NIPS. 2001.