

Convex combination belief propagation

Anna Grim*, Pedro Felzenszwalb

Applied Math, Brown University, 170 Hope Street Providence, RI 02906, United States



ARTICLE INFO

Article history:

Received 1 August 2021

Revised 17 September 2022

Accepted 23 September 2022

Keywords:

Belief propagation

Statistical inference

Graphical model

Image processing

Artificial intelligence

ABSTRACT

We present new message passing algorithms for performing inference with graphical models. Our methods are designed for the most difficult inference problems where loopy belief propagation and other heuristics fail to converge. Belief propagation is guaranteed to converge when the underlying graphical model is acyclic, but can fail to converge and is sensitive to initialization when the underlying graph has complex topology. This paper describes modifications to the standard belief propagation algorithms that lead to methods that converge to unique solutions on graphical models with arbitrary topology and potential functions.

© 2022 Elsevier Inc. All rights reserved.

1. Introduction

Graphical models provide a natural framework for probabilistic modeling and statistical inference with a large number of random variables. In this setting, statistical inference involves computing either marginal distributions for the individual random variables, or joint configurations with maximum probability. Exact inference is NP-hard in both cases, so it is essential to develop approximation algorithms that make inference computationally tractable [1].

Belief propagation (BP) is a widely used message passing algorithm that can be used to perform either exact or approximate inference in graphical models. This algorithm was introduced by Judea Pearl in the early 1980s as a method for performing exact inference on acyclic graphs in polynomial time [2]. For graphical models with cycles BP is a heuristic (often called *loopy BP*) that can be used to perform approximate inference. This method has gained popularity within the artificial intelligence community since it obtains state of the art results in certain settings such as error correcting codes and image analysis (see, e.g., [3–7]). However, BP is also known for failing to converge and for being sensitive to initialization. These issues are the main motivations behind this paper as well as many other works over the last few decades.

The BP literature has primarily focused on two key problems: (1) understanding why loopy belief propagation fails to converge and (2) using this knowledge to develop alternative local message passing algorithms. Broadly speaking, whether the algorithm converges depends on the topology of the graph and the nature of the potential functions in the graphical model.

We present *convex combination belief propagation* (CCBP), a convergent alternative to traditional belief propagation inspired by the work in [8]. Our primary motivation for developing CCBP is that loopy belief propagation is known to fail to converge on graphs with complex topology. CCBP addresses this problem by weighting the incoming messages when computing an outgoing message. This mitigates the effect of feedback loops and leads to globally convergent versions of both the sum-product and max-product BP algorithms.

* Corresponding author.

E-mail address: anna_grim@brown.edu (A. Grim).

Felzenszwalb and Svaiter used a type of non-linear diffusion to obtain globally convergent methods for approximate inference in graphical models [8]. The approach we take in this paper is closely related. CCBP involves fixed point iteration with contractive maps. For a fixed set of valid weights the process converges to a unique solution regardless of the topology of the graph and message initialization. It is simple to determine weights that guarantee uniqueness of the fixed point and convergence of fixed point iteration. The weights depend only on the local topology of the graph and do not depend on the strength or nature of the potentials in the graphical model. Moreover our weighting scheme can be readily incorporated into existing implementations of BP.

Tatikonda and Jordan showed that BP converges when the model satisfies Dobrushin's condition (see [9–11]). Intuitively, this condition states that the coupling factor (induced by the potentials) between variables must be sufficiently small. Ihler et al. derived a similar condition by bounding message error in terms of the dynamic range of the potentials [12]. Heskes (2004) derived a sufficient condition for the uniqueness of BP fixed points by equivalently determining when the Bethe free energy is convex (see [13–16]). The resulting conditions state that both cycles and the strength of the potentials affect whether this free energy is convex. Martin and Lasgouttes (2012) derive a sufficient condition for local convergence in terms of the graph structure and the beliefs values at the fixed point [17]. Their result provides insight into why BP is more likely to converge on sparser graphs.

These works have influenced the development of alternative local message passing algorithms that improve convergence by damping the potentials and/or mitigating the effects of feedback loops caused by message passing on graphs with cycles. Yedida et al. developed *generalized* belief propagation in which the number of cycles is reduced by formulating message passing between regions of nodes [18]. One drawback is that the performance is highly dependent upon how the graph is partitioned into regions, which can be a challenging task. Wainwright et al. developed *tree-reweighted* belief propagation by maximizing a lower bound on the log partition function via convex combinations of tree-structured distributions [19]. Kolmogorov introduced *sequential* tree-reweighted belief propagation [20] which improves the convergence of Wainwright's method by utilizing sequential (as opposed to parallel) updates. Although these alternatives have better convergence properties, they are more computationally demanding.

Roosta and Wainwright introduced a reweighted sum-product algorithm that incorporates edge-weights on the potentials and messages in each update [21]. The algorithm converges when the spectral radius of the update operator is bounded by one. However, it may be difficult to determine a set of weights that satisfies this condition. Knoll et al. introduced a homotopy continuation based approach called *self-guided* belief propagation [22]. Their method interpolates between a pairwise model and a simplification of that model with only unary potentials. Although the method always returns a solution, it's only guaranteed to converge on the first time step.

The remaining of the paper is organized as follows. Section 2 provides a brief overview of graphical models and belief propagation, while also establishing basic notation. We introduce CCBP in Section 3, then prove several theoretical properties of the algorithm. Section 4 discusses several numerical experiments that evaluate the performance of our algorithm.

2. Background

2.1. Probabilistic graphical models

We consider pairwise undirected graphical models (Markov random fields). Let $G = (V, E)$ be an undirected graph with the vertex set $V = \{1, \dots, n\}$. We use $N(i) = \{j : \{i, j\} \in E\}$ to denote the set of neighbors of node $i \in V$. Let $X = (X_1, \dots, X_n)$ be a random vector where X_i is a random variable with a set of possible outcomes $\Omega = \{1, \dots, m\}$. A configuration of the random vector is given by $x = (x_1, \dots, x_n) \in \Omega^n$.

The probability of a configuration is given by the joint distribution:

$$\mathbb{P}(X = x) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{\{i, j\} \in E} \psi_{ij}(x_i, x_j),$$

where Z is a normalization constant. The functions ϕ_i and ψ_{ij} are referred as *potentials* and assumed to be positive. The graphical model is said to be *pairwise* since the joint distribution is a product of potentials that depend on pairs of random variables.

Statistical inference is a central computational challenge in many applications. Exact inference is generally intractable for arbitrary distributions, especially when the joint distribution is defined over a large number of random variables. There are two inference tasks that frequently appear in applications: (1) finding the most probable (MAP) solution and (2) computing marginals.

1. *MAP Inference.* Determine the most probable state of the random vector,

$$\hat{x}_{\text{MAP}} = \arg \max_{x \in \Omega^n} \mathbb{P}(X = x).$$

2. *Marginal Inference.* Compute the marginal distribution of each random variable $i \in V$,

$$\mathbb{P}(X_i = \tau) = \sum_{\{x : x_i = \tau\}} \mathbb{P}(X = x).$$

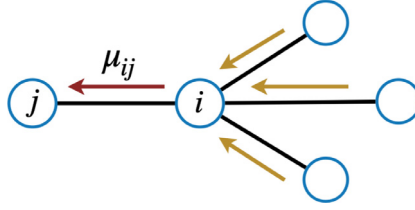


Fig. 1. Illustration of message passing. Node i sends a message to a neighboring node j that aggregates messages from all other neighbors of i .

There are two closely related belief propagation algorithms that can be used to address the two inference tasks. The max-product algorithm performs MAP inference, while the sum-product algorithm performs marginal inference.

2.2. Max-product algorithm

The max-product algorithm finds the MAP solution by computing max-marginals for each node. Given any node $i \in V$, the max-marginal of this node is

$$p_i(\tau) = \max_{\{x: x_i = \tau\}} \mathbb{P}(X = x).$$

Note that this definition bears a close resemblance to the definition of a marginal distribution. The value $p_i(\tau)$ specifies the maximum probability of a configuration where the state of the i -th random variable is τ . As long as there are no ties, the exact MAP solution can be obtained by selecting $\hat{x}_i = \arg \max p_i(\tau)$ for every node $i \in V$.

The main idea behind the max-product BP algorithm is to use dynamic programming to break up the calculation of the max-marginals into subproblems. This results in a local message passing algorithm, where computing messages is equivalent to solving subproblems. Each message sent from a node to a neighboring node incorporates messages from the other neighbors, and information propagates throughout the graph as illustrated in Fig. 1.

Let $\mu_{ij} \in \mathbb{R}^m$ be the message sent from node i to node j . Each message is defined using the max-product equation,

$$\mu_{ij}(x_j) = \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \mu_{ki}(x_i) \right\}.$$

Intuitively, the message μ_{ij} provides information regarding what state node i thinks node j should be in, where large values of $\mu_{ij}(x_j)$ correspond to favorable states. The messages are often normalized for numerical reasons, in which case μ_{ij} is a distribution over the set of possible states. The incoming messages at each node are combined to compute a *belief* function $b_j: \Omega \rightarrow \mathbb{R}_+$ given by

$$b_j(x_j) = \phi_j(x_j) \prod_{i \in N(j)} \mu_{ij}(x_j).$$

For acyclic graphs the resulting beliefs are the exact max-marginals (up to a non-consequential scaling factor due to the normalization of messages), and the MAP solution can be obtained by maximizing each belief individually.

When the graph contains cycles, this approach can be adapted into a fixed point iteration scheme referred to as *loopy* belief propagation. In this setting, the message passing equations are used to define an operator that updates all messages in parallel until convergence (there are sequential variants as well). This operator acts on the space

$$\mathcal{K} = \bigotimes_{i \in V} \bigotimes_{j \in N(i)} \Delta_+^m,$$

where Δ_+^m is the positive m -simplex. Here $\mu \in \mathcal{K}$ is a vector of messages.

Definition 1. The operator $T: \mathcal{K} \rightarrow \mathcal{K}$ in the max-product algorithm is $T = N\hat{T}$. \hat{T} computes new messages using the max-product equation and N normalizes each message.

$$(\hat{T}\mu)_{ij}(x_j) = \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \mu_{ki}(x_i) \right\}$$

$$(N\mu)_{ij}(x_j) = \frac{\mu_{ij}(x_j)}{\sum_{\tau} \mu_{ij}(\tau)}.$$

The messages are initialized as $\mu^{(0)} \in \mathcal{K}$ and repeatedly updated via the fixed point iteration scheme $\mu^{(n+1)} = T\mu^{(n)}$. This scheme is either run for a large number of iterations or stopped once the messages have sufficiently converged. The resulting messages $\mu^{(n)}$ are then used to compute beliefs $b_j^{(n)}$ for every node.

Definition 2. The belief $b_j^{(n)} : \Omega \rightarrow \mathbb{R}$ of node $j \in V$ after n iterations is given by

$$b_j^{(n)}(x_j) = \phi_j(x_j) \prod_{i \in N(j)} \mu_{ij}^{(n)}(x_j).$$

BP has been successfully used in a variety of applications that involves cyclic graphs and often leads to good results (see, e.g., [5–7]). By treating the final beliefs as approximations to max-marginals we can obtain a labelling by maximizing each belief individually. Although BP often obtains good results, there are several issues that may occur. First, this labelling may not correspond to the MAP solution since BP may have multiple fixed points. In addition, BP is also known to fail to converge and may be sensitive to the message initialization.

One numerical method that improves the performance of BP is to stabilize the fixed point iteration scheme with damping¹. This involves updating messages with the convex combination:

$$\mu^{(n+1)} := (1 - \alpha) T \mu^{(n)} + \alpha \mu^{(n)}$$

where $\alpha \in (0, 1)$ is referred as the damping factor. In practice, damped BP (i.e. BP with damping) often prevents the messages from oscillating and converges faster than non-damped BP when both algorithms converge. However, even damped BP is still not guaranteed to converge. We provide an example where damped BP does not converge below (Example 3).

2.3. Sum-product algorithm

The sum-product BP algorithm is a variation of the max-product BP algorithm that can be used to compute marginal distributions. The sum-product algorithm involves fixed point iteration with an operator that is nearly the same as in the max-product case. The only difference is that this operator includes a “sum” instead of a “max” in the message update equations.

Definition 3. The operator $T : \mathcal{K} \rightarrow \mathcal{K}$ in the sum-product algorithm is $T = N\hat{T}$. \hat{T} computes new messages using the sum-product equation and N normalizes messages to sum to one.

$$(\hat{T}\mu)_{ij}(x_j) = \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \mu_{ki}(x_i).$$

$$(N\mu)_{ij}(x_j) = \frac{\mu_{ij}(x_j)}{\sum_{\tau} \mu_{ij}(\tau)}.$$

As in the max-product case messages are initialized to arbitrary values and repeatedly updated via fixed point iteration. After convergence the messages are used to compute beliefs for every node. In the sum-product algorithm the beliefs provide an approximation to the marginal distributions of each random variable.

Similar to the max-product algorithm the sum-product algorithm performs exact inference in polynomial time when the graph is acyclic. For cyclic graphs the sum-product algorithm is a heuristic. The sum-product algorithm (even with damping) suffers from the exact same issues as the max-product algorithm. Namely, the algorithm can fail to converge and can return different results depending on the initial set of messages.

3. Convex combination belief propagation

Now present *convex combination belief propagation* (CCBP), a globally convergent alternative to belief propagation. The main objectives of this section are to introduce the new message passing operator, then prove that CCBP converges to a unique fixed point on graphs with arbitrary topology and arbitrary potential functions. This section focuses on the max-product version of CCBP, the sum-product version is analogous and described in Appendix B.

3.1. Message passing operator

Graphs with many short cycles are especially problematic for BP because they create feedback loops where information within messages is over-counted. As a result, the messages may oscillate, converge to inaccurate beliefs, or converge to different fixed points depending on the initialization.

To define CCBP we take the operator from BP and weight the incoming messages when computing a new outgoing message. When computing the message from node i to node j we weight the incoming message from another neighbor k by w_{ki} . The only conditions imposed upon these weights is that they are non-negative and sum to (at most) one. We also discount all of the incoming messages by a factor $\gamma \in (0, 1)$.

¹ This scheme is referred to as the Krasnoselskij iteration scheme in the numerical analysis literature.

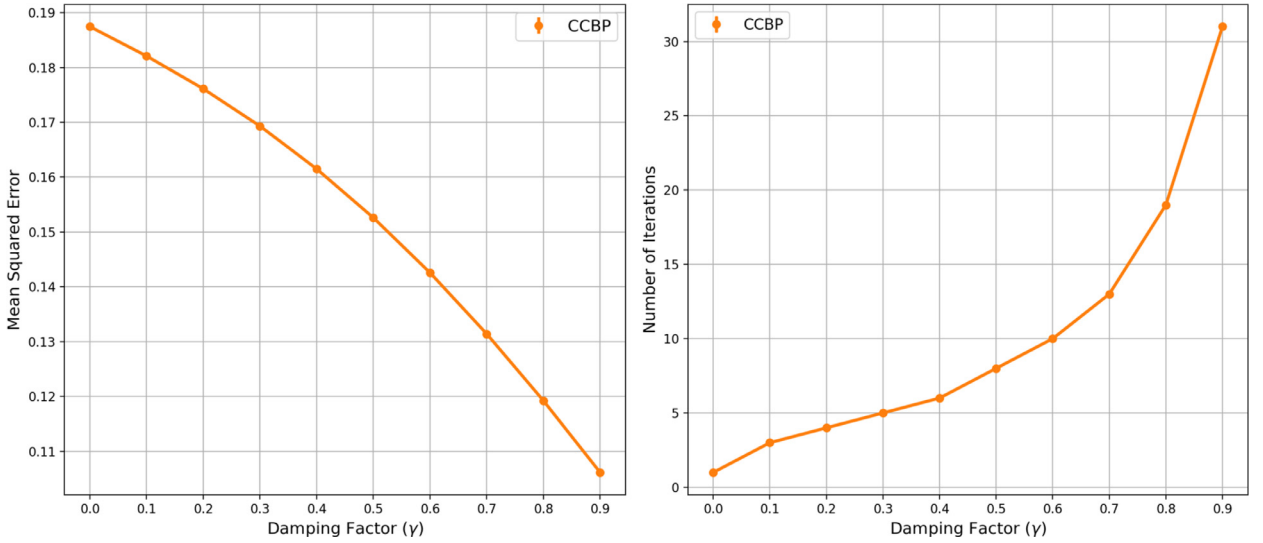


Fig. 2. Results of experiment in Example 1.

Definition 4. The operator $S : \mathcal{M} \rightarrow \mathcal{M}$ in the max-product CCBP algorithm is

$$(S\mu)_{ij}(x_j) = \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \left(\prod_{k \in N(i) \setminus j} \mu_{ki}(x_i)^{w_{ki}} \right)^\gamma \right\},$$

where the weights must be non-negative with $\sum_{k \in N(i) \setminus j} w_{ki} \leq 1$, and $\gamma \in (0, 1)$.

Note that we do not incorporate normalization in the definition of the operator. The exclusion of the normalization factor implies that the operator acts on the space:

$$\mathcal{M} := \bigotimes_{i \in V} \bigotimes_{j \in N(i)} \mathbb{R}_+^m$$

The simplest way to define each weight is to set them uniformly for each node based on the degree, $w_{ki} = 1/(d(i) - 1)$, where $d(i)$ is the degree of node i . Alternatively one can give more weight to some edges based on some additional information from a particular application (see [8]). Intuitively, the weights control how much influence neighboring nodes have upon each other. When the message sent from node i to j incorporates uniform weights, the other neighbors have equal influence upon node j . Non-uniform weights may be used to give some neighbors more influence.

This operator also incorporates a damping factor $\gamma \in (0, 1)$. Later we see that this term is the Lipschitz constant of the operator. Thus, the rate of convergence is dependent upon the magnitude of this parameter. Next we present an example and numerical experiments that illustrates how the magnitude of γ affects the performance of CCBP.

Example 1. Let $G = (V, E)$ be an undirected graph with 10 nodes, where each pair of nodes is connected with probability 0.5. Let $\Omega = \{-1, 1\}$ be the set of possible states for each random variable. Consider the joint distribution

$$\mathbb{P}(X = x) = \frac{1}{Z} \exp \left(- \sum_{i \in V} x_i y_i - \sum_{\{i, j\} \in E} \lambda_{ij} x_i x_j \right)$$

We generated a concrete problem instance from this model by independently sampling y_i from $\{-1, 1\}$ and λ_{ij} from a normal distribution. Then we repeatedly applied CCBP to this problem instance, while varying the magnitude of γ from 0 to 0.9 in increments of $\Delta\gamma = 0.1$. The weights w_{ki} were set uniformly. The performance of the algorithm is determined by (1) computing the mean square error between the resulting beliefs after convergence and true max-marginals and (2) the number of iterations until convergence.

The results from the numerical experiments with Example 1 are shown in Fig. 2. We see that the algorithm converges faster when the magnitude of γ is smaller. However, the algorithm obtains a better approximation of the true max-marginals for larger values of γ . We recommend setting this factor close to one (i.e. $\gamma = 0.9$) since the algorithm converges very quickly. Note that the algorithm is guaranteed to converge for any value $\gamma < 1$.

3.2. Convergence

Next we prove that CCBP converges to a unique fixed point independent of the topology of the graph and message initialization. Our general approach is to consider CCBP as a discrete-time map, then use results from the theory of dynamical systems. In particular, we invoke Banach's fixed point theorem which guarantees existence and uniqueness of fixed points of certain discrete-time maps referred to as *contractions*. Given a metric space (X, d) , an operator $F : X \rightarrow X$ is called a contraction if there is a constant $\gamma \in (0, 1)$ such that

$$d(F(x), F(y)) \leq \gamma d(x, y)$$

for any $x, y \in X$. Banach's fixed point theorem states that a contractive operator (i.e. contraction) defined over a complete metric space has a unique and globally attractive fixed point $x^* \in X$ [23]. In order to use this result, we must define a metric which is complete with respect to the space \mathcal{M} .

Proposition 1. Let $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ be the distance function given by

$$d(\mu, \nu) = \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}(x_j) - \log \nu_{ij}(x_j)|,$$

then the pair (\mathcal{M}, d) is a complete metric space.

Proof. See Appendix A for proof. \square

The metric d computes the distance between individual messages $\mu, \nu \in \mathcal{M}$. Next we prove that S is a contraction with respect to this metric.

Lemma 1. The operator S is a contraction with Lipschitz constant γ .

Proof. Choose any $\mu, \nu \in \mathcal{M}$, then

$$\begin{aligned} (S\mu)_{ij}(x_j) &= \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \mu_{ki}(x_i)^{\gamma w_{ki}} \right\} \\ &= \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \nu_{ki}(x_i)^{\gamma w_{ki}} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \right\} \\ &\leq \max_{x_i} \left\{ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \nu_{ki}(x_i)^{\gamma w_{ki}} \right\} \max_{x_i} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \\ &= (S\nu)_{ij}(x_j) \max_{x_i} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \end{aligned}$$

Given that both sides of the inequality are positive, we apply the logarithm to obtain:

$$\begin{aligned} \log (S\mu)_{ij}(x_j) - \log (S\nu)_{ij}(x_j) &\leq \gamma \sum_{k \in N(i) \setminus j} w_{ki} \max_{x_i} |\log \mu_{ki}(x_i) - \log \nu_{ki}(x_i)| \\ &\leq \gamma \max_{k \in N(i) \setminus j} \max_{x_i} |\log \mu_{ki}(x_i) - \log \nu_{ki}(x_i)| \end{aligned}$$

Since this inequality holds when μ and ν are interchanged, it also holds with respect to the absolute value of the left hand side. In addition, this inequality holds for any $x_j \in \Omega$ and so it holds for the maximum of the left hand side.

$$\begin{aligned} \max_{x_i} |\log (S\mu)_{ij}(x_j) - \log (S\nu)_{ij}(x_j)| &\leq \gamma \max_{k \in N(i) \setminus j} \max_{x_i} |\log \mu_{ki}(x_i) - \log \nu_{ki}(x_i)| \\ &\leq \gamma \max_{i \in V} \max_{j \in N(i)} \max_{k \in N(i) \setminus j} \max_{x_i} |\log \mu_{ki}(x_i) - \log \nu_{ki}(x_i)| \\ &= \gamma \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}(x_j) - \log \nu_{ij}(x_j)| \\ &= \gamma d(\mu, \nu) \end{aligned}$$

Since the inequality holds for any $\{i, j\} \in E$ on the left hand side we conclude

$$d(S\mu, S\nu) \leq \gamma d(\mu, \nu).$$

\square

Now our main theorem follows directly from the contraction result.

Theorem 1. The operator S has a unique fixed point $\mu^* \in \mathcal{M}$ and any sequence of messages defined by $\mu^{(n+1)} := S\mu^{(n)}$ converges to μ^* . Furthermore, after n iterations

$$d(S^{(n)}\mu^{(0)}, \mu^*) \leq \gamma^n d(\mu^{(0)}, \mu^*).$$

Proof. S is a contraction with respect to the metric d according to Lemma 1 and the space \mathcal{M} is complete with respect to d . The result holds by applying Banach's fixed point theorem. \square

This result shows that CCBP is guaranteed to converge when the weights satisfy the simple constraints stated in the definition of the operator. Note that the constraints we have on the weights do not depend on the strength of the potential functions in the graphical model. The weights can also be easily selected based on the degree of each node.

Finally we note that the CCBP operator has a unique fixed point once the weights w_{ki} and γ are fixed, but the choice of weights and γ do affect the resulting fixed point. A simple choice for these parameters involves setting $w_{ki} = 1/(d(i) - 1)$ and γ to be a value close to 1.

3.3. Characterization of beliefs

Although CCBP is guaranteed to converge, the algorithm is only useful if it returns a meaningful result. When the underlying graph is acyclic the traditional max-product algorithm computes the exact max-marginals. In this section, we provide an analogous characterization of the beliefs obtained with CCBP.

3.3.1. Min-Sum algorithm

CCBP along with other related methods improve convergence by changing the optimization landscape. For our purposes, it is more natural to discuss this optimization problem in terms of minimizing an energy as opposed to maximizing a joint distribution. Given that the potentials are assumed to be positive, the joint distribution can be equivalently written as

$$\mathbb{P}(X = x) = \frac{1}{Z} e^{-E(x)},$$

where E is an energy. In the case of a pairwise model, this energy is

$$E(x) = \sum_{i \in V} g_i(x_i) + \sum_{\{i,j\} \in E} h_{ij}(x_i, x_j), \quad (1)$$

where $g_i(x_i) = -\log \phi_i(x_i)$ and $h_{ij}(x_i, x_j) = -\log \psi_{ij}(x_i, x_j)$ are cost functions. With these definitions the most probable (MAP) solution is the configuration of the random vector that minimizes the energy $E(x)$.

The max-product CCBP algorithm can be implemented with negative log probabilities, where the max-product becomes a min-sum. The equivalent message passing operator is

$$(S\mu)_{ij}(x_j) = \min_{x_i} \left\{ g_i(x_i) + h_{ij}(x_i, x_j) + \gamma \sum_{k \in N(i) \setminus j} w_{ki} \mu_{ki}(x_i) \right\},$$

In this case, the resulting beliefs provide an approximation of the *min-marginals* of the energy. Alternatively, the min-sum algorithm can also be derived by using dynamic programming to compute the global minima of the energy $E(x)$. This derivation bears a close resemblance to the Viterbi algorithm for MAP estimation in hidden Markov model [24]. For the remainder of this section, we focus on the min-sum formulation.

3.3.2. Tree-structured graphs

Let μ be the unique fixed point of S , the belief $b_j : \Omega \rightarrow \mathbb{R}$ is

$$b_j(x_j) = g_j(x_j) + \sum_{i \in N(j)} \mu_{ij}(x_j).$$

The main objective of this section is prove that these beliefs are the exact min-marginals of a weighted energy $E_j(x)$. This energy is closely related to $E(x)$, but a key difference is that each cost function is weighted according to the weights and damping factor used in the definition of S . We prove that each belief b_j is the min-marginal of a different energy function $E_j(x)$ as the weights of each term depend on the choice of j .

Before defining the weighted energy for an arbitrary tree-structured graph we provide a simple example of how to write down this energy for a small graph. In order to simplify the expressions for the energies, we define the cost function $H_{ij}(x_i, x_j) = g_i(x_i) + h_{ij}(x_i, x_j)$.

Example 2. Let G be the graph shown in Fig. 3. In this case, the energy E is given by

$$E(x) = g_1(x_1) + H_{21}(x_2, x_1) + H_{32}(x_3, x_2) + H_{43}(x_4, x_3) + H_{53}(x_5, x_3).$$

Let $w_{ki} = 1/(d(i) - 1)$ in the definition of S . When CCBP is applied to this graph, the resulting belief function b_1 is the min-marginal of the energy:

$$E_1(x) = g_1(x_1) + H_{21}(x_2, x_1) + \gamma H_{32}(x_3, x_2) + \frac{\gamma^2}{2} H_{43}(x_4, x_3) + \frac{\gamma^2}{2} H_{53}(x_5, x_3),$$

This energy contains the same cost functions as the energy E , but the difference is that each term $H_{ij}(x_i, x_j)$ is multiplied by a product of weights and a power of γ .

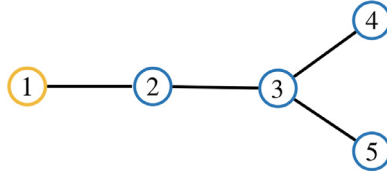
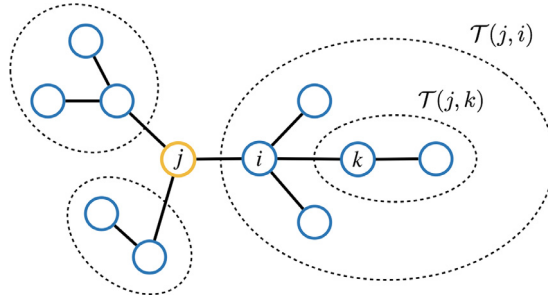


Fig. 3. A simple tree-structured graph.

Fig. 4. Subtrees of $\mathcal{T}(j)$.

Next we generalize the energy in the example above to the case of an arbitrary acyclic graph. Let $\mathcal{T}(j)$ be a tree-structured graph with a distinguished root node $j \in V$. Once a tree \mathcal{T} is rooted at a node j , every node $i \neq j$ has a unique parent $\mathcal{P}(j, i)$, a set of children $\mathcal{C}(j, i)$, and a set of descendants $\mathcal{D}(j, i)$. Let $\mathcal{T}(j, i)$ denote the subtree of $\mathcal{T}(j)$ rooted at a node i as illustrated in Fig. 4. The subtree $\mathcal{T}(j, i)$ includes node i and its descendants. Let $R(\mathcal{T}(j))$ be the depth of a rooted tree. Let $N_d(\mathcal{T}(j)) \subseteq V$ with $d \geq 0$ be the set of nodes at distance d from the root j .

Definition 5. Let $w : V \times V \rightarrow \mathbb{R}$ be the weight function

$$w(i, j) = \prod_{(k, \ell) \in E(i, j)} w_{k\ell},$$

where $E(i, j)$ is the collection of directed edges along the path from node i to j .

Definition 6. Let $E_j : \Omega^n \rightarrow \mathbb{R}$ be the weighted energy given by

$$E_j(x) = g_j(x_j) + \sum_{i \in N(j)} \sum_{d=0}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}),$$

This energy is closely related to $E(x)$ but it puts more weight on terms “near” node j . For each edge $(k, \mathcal{P}(j, k))$ the cost associated with that edge is scaled by $\gamma^d w(k, i)$. When the damping factor and weights are removed from E_j , this energy is exactly $E(x)$.

Proposition 2. If $w_{ki} = 1$ for all $\{k, i\} \in E$ and $\gamma = 1$, then $E_j(x) = E(x)$.

Proof.

$$\begin{aligned} E_j(x) &= g_j(x_j) + \sum_{i \in N(j)} \sum_{d=0}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \\ &= g_j(x_j) + \sum_{i \in N(j)} \sum_{d=0}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \left(g_k(x_k) + h_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right) \\ &= g_j(x_j) + \sum_{k \in V \setminus j} \left(g_k(x_k) + h_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right) \\ &= \sum_{k \in V} g_k(x_k) + \sum_{\{k,i\} \in E} h_{ki}(x_k, x_i) \\ &= E(x). \end{aligned}$$

□

Next, we prove that CCBP computes the exact min-marginals of the weighted energy in Definition 6. The main idea of the argument is to use that computing the min-marginals in a tree-structured graphical model can be broken down into subproblems that can be solved recursively. Each subproblem involves a function $F_{ji} : \Omega \rightarrow \mathbb{R}$ that corresponds to the min-marginals of an optimization problem defined by $\mathcal{T}(j, i)$.

Lemma 2. Let $F_{ji} : \Omega \rightarrow \mathbb{R}$ be the function given by

$$F_{ji}(x_i) = \min_{x_{\mathcal{D}(j,i)}} \left\{ \sum_{d=1}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right\},$$

then $F_{ji}(x_i) = \gamma \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} w_{ki} \mu_{ki}(x_i)$ where μ is the fixed point of S .

Proof. This claim can be proven by inducting on the depth of the subtree $\mathcal{T}(j, i)$. In the base case when the depth is 1, then $F_{ji}(x_i) = 0$ and the result follows trivially.

Now for the induction step,

$$\begin{aligned} F_{ji}(x_i) &= \min_{x_{\mathcal{D}(j,i)}} \left\{ \sum_{d=1}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right\} \\ &= \min_{x_{\mathcal{D}(j,i)}} \left\{ \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} \gamma w_{ki} H_{ki}(x_k, x_i) + \sum_{d=2}^{R(\mathcal{T}(j,i))-1} \sum_{k' \in N_d(\mathcal{T}(j,i))} \gamma^d w(k', i) H_{k'\mathcal{P}(j,k')}(x_{k'}, x_{\mathcal{P}(j,k')}) \right\} \\ &= \min_{x_{\mathcal{D}(j,i)}} \left\{ \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} \gamma w_{ki} \left(H_{ki}(x_k, x_i) + \sum_{d=1}^{R(\mathcal{T}(j,k))-1} \sum_{k' \in N_d(\mathcal{T}(j,k))} \gamma^d w(k', k) H_{k'\mathcal{P}(j,k')}(x_{k'}, x_{\mathcal{P}(j,k')}) \right) \right\} \\ &= \gamma \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} w_{ki} \min_{x_k} \left\{ H_{ki}(x_k, x_i) + \min_{x_{\mathcal{D}(j,k)}} \left\{ \sum_{d=1}^{R(\mathcal{T}(j,k))-1} \sum_{k' \in N_d(\mathcal{T}(j,k))} \gamma^d w(k', k) H_{k'\mathcal{P}(j,k')}(x_{k'}, x_{\mathcal{P}(j,k')}) \right\} \right\} \\ &= \gamma \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} w_{ki} \min_{x_k} \left\{ H_{ki}(x_k, x_i) + F_{jk}(x_k) \right\} \\ &= \gamma \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} w_{ki} \min_{x_k} \left\{ H_{ki}(x_k, x_i) + \gamma \sum_{k' \in N(k) \setminus \mathcal{P}(j,k)} w_{k'k} \mu_{k'k}(x_{k'}) \right\} \\ &= \gamma \sum_{k \in N(i) \setminus \mathcal{P}(j,i)} w_{ki} \mu_{ki}(x_i). \end{aligned}$$

□

Theorem 2. The belief b_j is the exact min-marginal of E_j with respect to the j -th variable.

Proof. Choose any $j \in V$, then the min-marginal p_j of E_j is

$$\begin{aligned} p_j(\tau) &= \min_{\{x: x_j = \tau\}} E_j(x) \\ &= g_j(\tau) + \min_{\{x: x_j = \tau\}} \left\{ \sum_{i \in N(j)} \sum_{d=0}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right\} \\ &= g_j(\tau) + \sum_{i \in N(j)} \min_{x_i} \left\{ H_{ij}(x_i, \tau) + \min_{x_{\mathcal{D}(j,i)}} \left\{ \sum_{d=1}^{R(\mathcal{T}(j,i))-1} \sum_{k \in N_d(\mathcal{T}(j,i))} \gamma^d w(k, i) H_{k\mathcal{P}(j,k)}(x_k, x_{\mathcal{P}(j,k)}) \right\} \right\} \end{aligned}$$

Now Lemma 2 implies that

$$\begin{aligned} p_j(\tau) &= g_j(\tau) + \sum_{i \in N(j)} \min_{x_i} \left\{ H_{ij}(x_i, \tau) + F_{ji}(x_i) \right\} \\ &= g_j(\tau) + \sum_{i \in N(j)} \min_{x_i} \left\{ H_{ij}(x_i, \tau) + \gamma \sum_{k \in N(i) \setminus j} w_{ki} \mu_{ki}(x_i) \right\} \\ &= g_j(\tau) + \sum_{i \in N(j)} \mu_{ij}(\tau) \\ &= b_j(\tau). \end{aligned}$$

□

4. Numerical experiments

In this section, we present a series of numerical experiments that evaluate the performance of CCBP. First, we use the sum-product version of BP and CCBP to perform inference in the spin glass model. We compare the results obtained by these algorithms to the exact marginals computed by brute force. We also demonstrate the practicality of the max-product CCBP algorithm by using it to restore a noisy image.

4.1. Spin glass model

The objective of this experiment is to evaluate the performance of CCBP on inference tasks where traditional BP fails to converge. We apply each algorithm to a large number of problems, then evaluate the results with several performance metrics.

4.1.1. Experimental settings

The main idea behind this experiment is evaluate our algorithm on random graphical models while varying some parameter which influences the difficulty of performing inference. The purpose of this section is to describe how problems are generated and define the performance metrics used to evaluate the algorithms.

Let $G = (V, E)$ be an undirected graph. Let $\Omega = \{-1, 1\}$ be the set of possible outcomes for each random variable. The probability of a configuration of the random variables is given by

$$\mathbb{P}(X = x) = \frac{1}{Z} \exp \left(- \sum_{i \in V} x_i y_i - \sum_{\{i, j\} \in E} \lambda_{ij} x_i x_j \right).$$

In each problem instance, we generate a graph with 10 nodes such that the probability of two nodes being connected is 0.5. We refer to this value as the edge appearance probability. Each y_i is uniformly sampled from $\{-1, 1\}$ and the λ_{ij} are independently sampled from a uniform distribution. The magnitude of λ_{ij} controls how strongly neighboring states are coupled and the sign determines whether neighbors prefer to be aligned or misaligned.

For each algorithm, we use the initialization $\mu_{ij}^{(0)} = (1, 1)$ for all $\{i, j\} \in E$ and update the messages in parallel. Let $N = 10^3$ be the maximum number of iterations. Let $\epsilon = 10^{-2}$ be a threshold that indicates when the messages have sufficiently converged. For traditional BP, we use the damping factor $\alpha = 0.9$ to prioritize convergence since we consider difficult problem instances. We use the damping factor $\gamma = 0.9$ in CCBP and set the weights uniformly.

The performance of each algorithm is determined by the accuracy of the resulting beliefs, runtime, and rate of convergence. The accuracy of the beliefs is determined by computing the measure square error (MSE) between the normalized beliefs and the true marginals,

$$\text{MSE} = \frac{1}{n} \sum_i \sum_{x_i} |\hat{b}_i(x_i) - p_i(x_i)|^2,$$

where p_i is the exact marginal of node $i \in V$ (computed by brute force). The runtime refers to the number of iterations until convergence the algorithm sufficiently converges. We only report the runtime for instances where the algorithm converges.

4.1.2. Coupling factor

BP is known to fail when the message passing operator has repulsive fixed points. This complicates fixed point iteration because the scheme becomes unstable and is unlikely to converge for any damping factor. This scenario can be simulated by sampling the coupling factors λ_{ij} from a uniform distribution centered about zero. In this case, repulsive fixed point are likely to emerge because the coupling factors are both positive and negative.

In this experiment, we consider models where $\lambda_{ij} \sim \text{Unif}(-\sigma, \sigma)$ with $\sigma \geq 0$. The magnitude of σ was varied from 0 to 5 in increments of $\Delta\sigma = 0.5$. For each value of σ , we generate 100 graphical models. The results of this experiment in terms of the previously defined performance metrics are shown in Figs. 5 and 6. Note that each data point represents the average over all runs corresponding to a given value of σ .

The results shown in Fig. 5 and 6 illustrate some key advantages of CCBP. First, we see that the algorithm always converges as expected due the result in Theorem 1. Moreover, we see that the runtime of the algorithm is relatively constant even as the strength of the coupling factor increases. This behavior is also expected due the operator being a contraction. In this case, the distance between the unique fixed point and current set of messages decreases by a factor of γ on each iteration (see Theorem 1).

One key distinction between CCBP and other alternative message passing algorithms is that we apply weights to the messages instead of the potentials. In previous methods weights were applied to potentials so that the strength of the coupling factor is sufficiently small to ensure convergence. This idea traces back to previous works that aim to derive criteria that guarantees when BP converges (e.g. Dobrushin's condition, see [10,11]). One drawback of this approach is that the weights must be smaller as the strength of the potentials increases. One important advantage of CCBP is that the condition on the weights is independent of the strength of the potentials. In the experiments described here the weights were set uniformly based on the degree of each node.

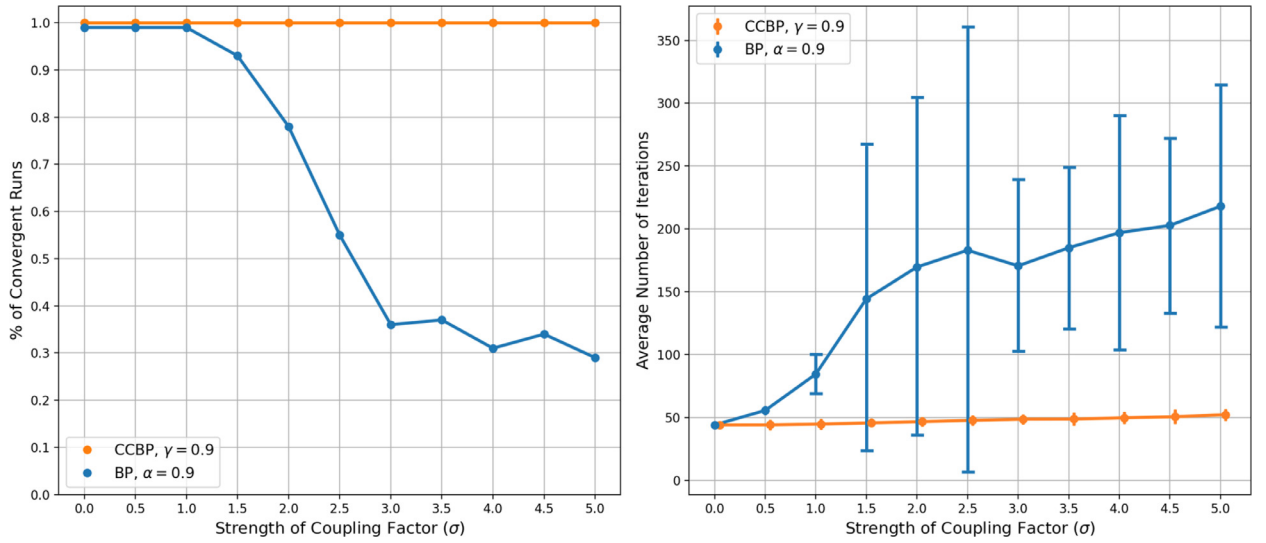


Fig. 5. Convergence rate and runtime of BP algorithms.

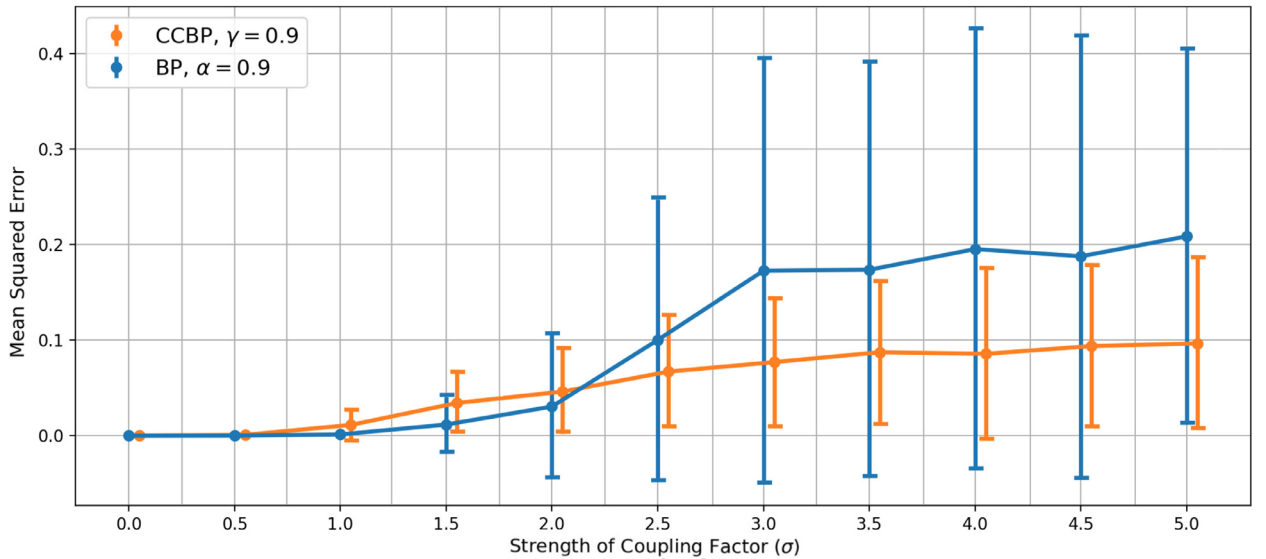


Fig. 6. Accuracy of BP algorithms.

4.1.3. Edge connectivity

BP is also known to either fail to converge or return a poor approximation when the topology of the graph is complex. In particular, the performance of the algorithm suffers when the graph contains many cycles because this leads to feedback loops. In this experiment, we compare the performance of the belief propagation algorithms while gradually increasing the connectivity of the graph.

We consider graphical models where the edge appearance probability p is varied from 0 to 1 with increments of $\Delta p = .1$. For each value of p , we obtain 100 problem instances by generating a random graph and independently sampling $\lambda_{ij} \sim \text{Unif}(-5, 5)$ for each problem instance. The results of this experiment are shown in Figs. 7 and 8.

As expected, we see that CCBP converges independent of the topology of the graph. It is also important to note that the rate of convergence is also independent of the topology. This is again a consequence of the message passing operator being a contraction with Lipschitz constant γ . We see that CCBP converged after at most 50 iterations in both experiments described in this section (see Figs. 5 and 7).

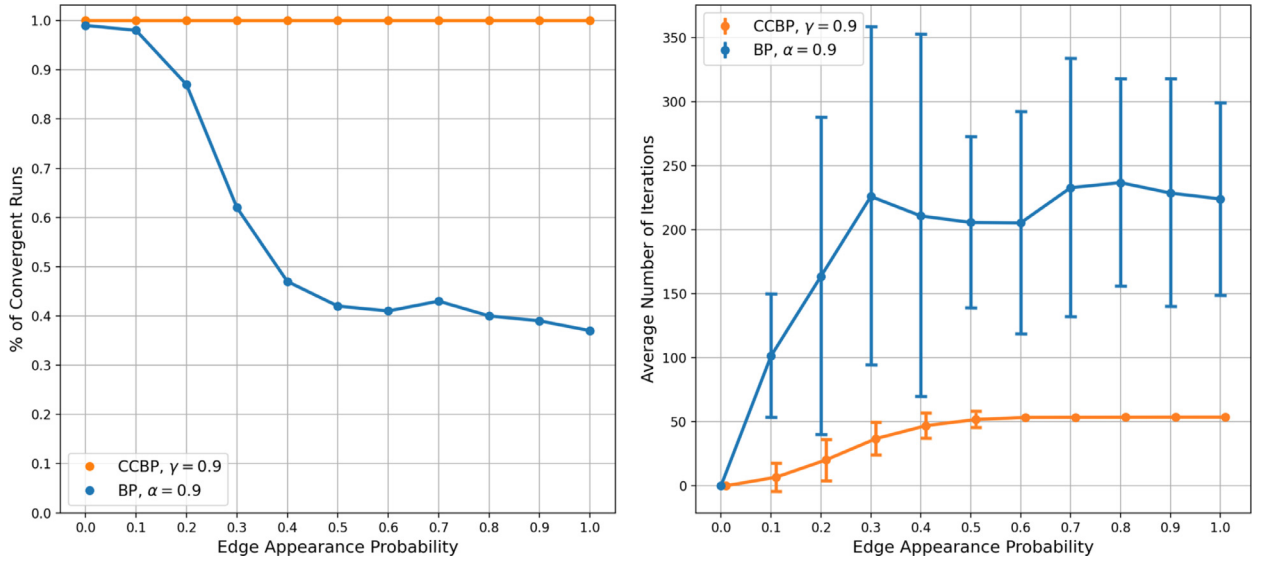


Fig. 7. Convergence rate and runtime of BP algorithms.

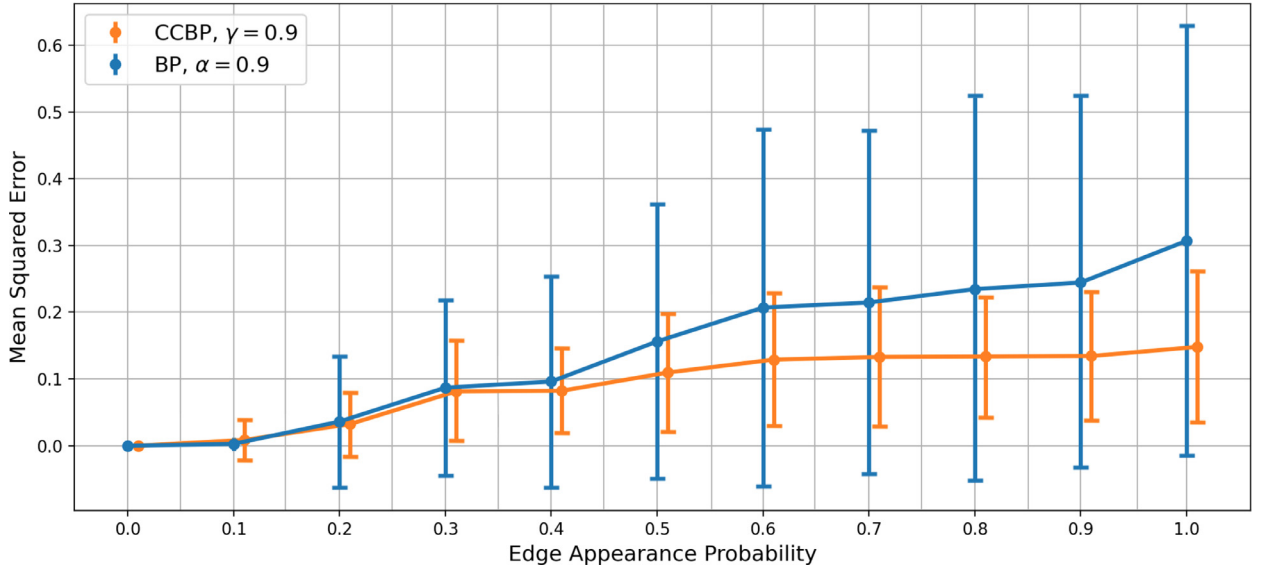


Fig. 8. Accuracy of BP algorithms.

4.2. Image restoration

In this section, we demonstrate a practical application of max-product CCBP by using this algorithm to perform image restoration. The objective of image restoration is to estimate a clean (original) image given a corrupted version of it. A classical approach is to formulate this problem as finding the optimal labelling of a graph with respect to an energy (see, e.g., [25–27]). The cost functions used in the energy enforce that the restored image is both piecewise smooth and consistent with the observed image. Once the image restoration model is formulated under this framework, CCBP can be used to restore the corrupted image by obtaining an approximation to the minimum of the energy.

To formalize this approach, let \mathcal{I} be an $n \times m$ image and let $G_{n,m} = (V, E)$ be an $n \times m$ grid graph that provides a graphical representation of the image. Each pixel in the image corresponds to a node in this graph and the edges connect neighboring pixels as shown in Fig. 9. Under the assumption that \mathcal{I} is an 8-bit image, the label space $\Omega = \{0, \dots, 255\}$ is the set of possible pixel values. Let $x = (x_1, \dots, x_{nm}) \in \Omega^{nm}$ denote a labelling of the graph and note that a labelling defines an image. Let $y = (y_1, \dots, y_{nm}) \in \Omega^{nm}$ denote the observed (corrupted) image. The cost functions used in this application are

$$g_i(x_i) = |x_i - y_i|^2 \quad \text{and} \quad h_{ij}(x_i, x_j) = \lambda \min(|x_i - x_j|^2, \tau).$$

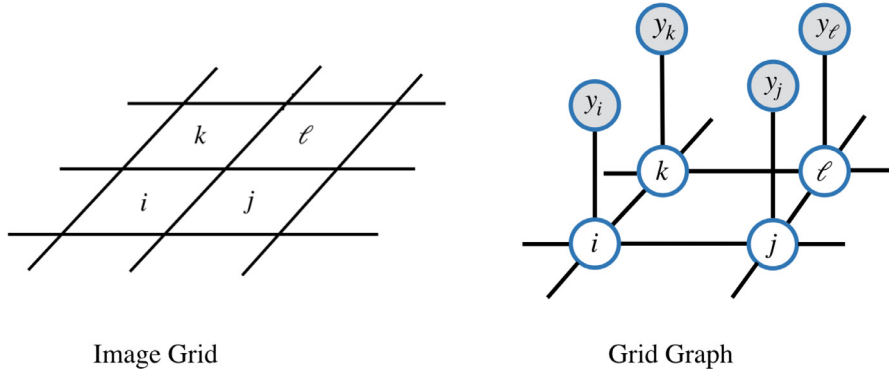


Fig. 9. On the left, we see a small region of an image with pixels i, j, k, ℓ in the image grid. On the right, we see the corresponding nodes in the grid graph. In addition, each node has an observed pixel value which is depicted by the gray colored node.

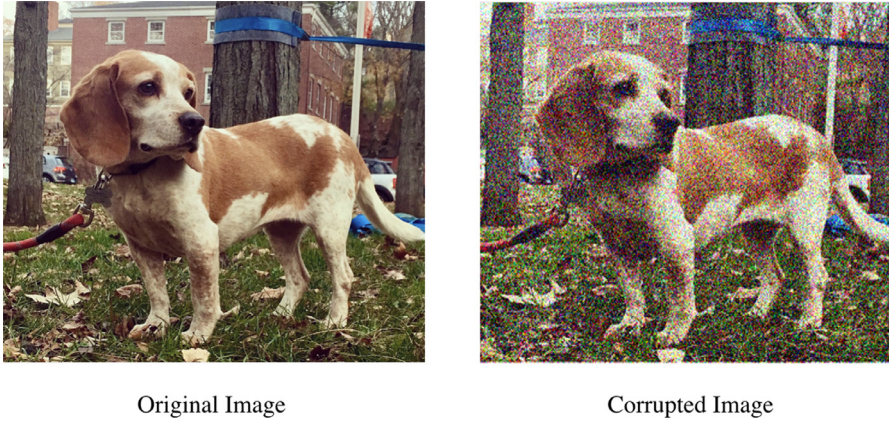


Fig. 10. Data used in the image restoration experiment.

The unary cost g_i enforces that pixels in the restored image are consistent with the observation. The pairwise cost h_{ij} enforces that the restored image is piecewise smooth with spatially coherent regions. The quadratic difference in h_{ij} is bounded by $\tau \in \Omega$ to allow for large differences between neighboring pixels, which occurs when two neighbors belong to different objects. The parameter $\lambda > 0$ controls how much weight is placed on the consistency versus smoothing terms in the energy, where large values of λ result in smoother images. Under this framework, an approximation to the clean image can be obtained by minimizing the energy given by

$$E(x) = \sum_{i \in V} |x_i - y_i|^2 + \lambda \sum_{\{i, j\} \in E} \min(|x_i - x_j|^2, \tau).$$

For our experiments we used the RGB image shown on the left hand side of Fig. 10. This image is composed of three color channels, where each channel is an 8-bit image with dimensions 400×466 . We restore the RGB image by separately restoring each of the three color channels.

We generated a corrupted version of the image by adding independent noise to each pixel, by sampling from a Gaussian distribution with mean $\mu = 0$ and standard deviation $\sigma = 50$. In Fig. 10 we show the original image on the left and a corrupted version of this image on the right. The weights used in the operator S were set uniformly and we set $\gamma = 0.99$. The parameters in our image model were set as $\tau = 100$ and we tried several values of λ . We applied CCBP to restore each channel of the corrupted image and the algorithm converged after at most 8 iterations.

A crucial component of this model is choosing a good value for the parameter λ . We obtained the best results with $\lambda = 3$ as shown in Fig. 11. The restored image is smooth almost everywhere. Importantly the result also preserves sharp discontinuities at the boundary of different objects.

5. Discussion

The purpose of this paper is to introduce an alternative to traditional belief propagation that we refer to as convex combination belief propagation. The primary advantage of this new method is that it converges to unique fixed points on graphs with arbitrary topology, independent of the initialization of messages. We provided a characterization of the beliefs



Fig. 11. Image restoration using CCBP with $\lambda = 4$.

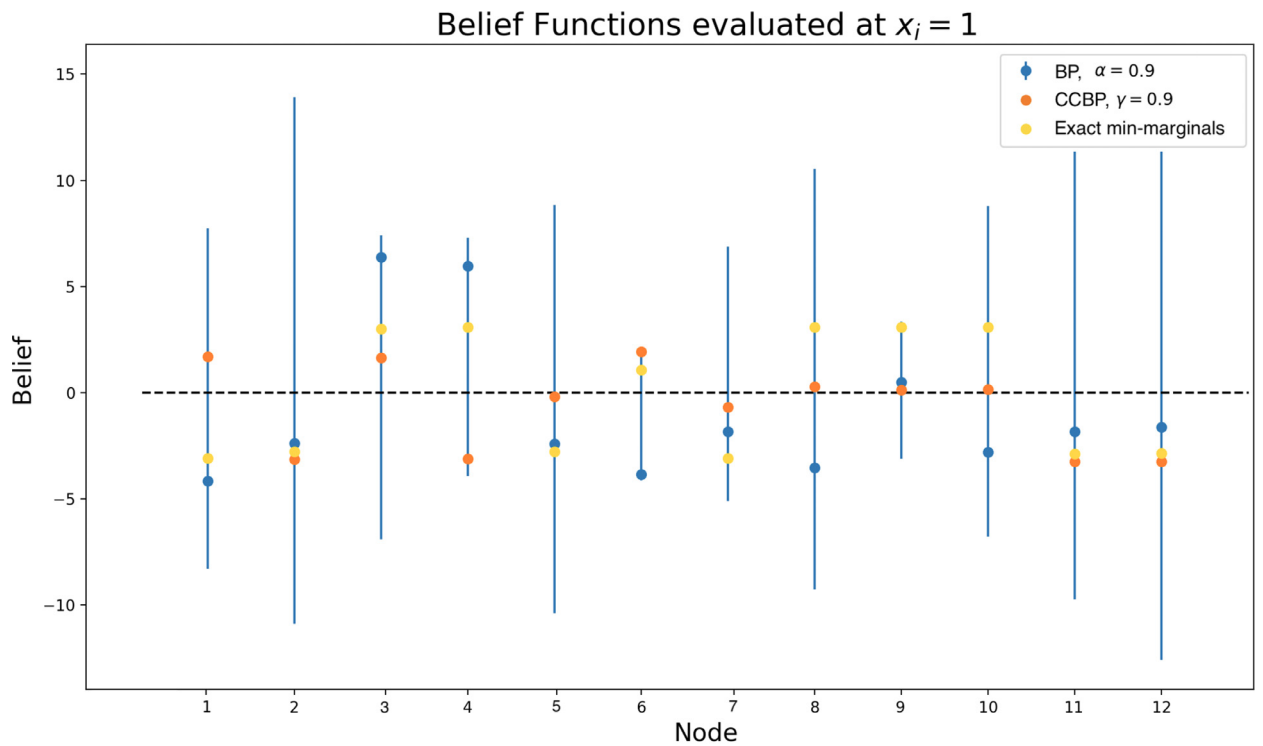


Fig. 12. Beliefs obtained from min-sum algorithms and exact min-marginals of the energy from Example 3. BP failed to converge and the vertical lines show the range of oscillation in the last 100 iterations before stopping the algorithm after 1000 iterations. CCBP converged as expected, but the resulting beliefs differ from the exact min-marginals.

obtained from the min-sum version of the algorithm in the case when the graph is acyclic. In addition, we demonstrated the practical application of this algorithm for image restoration.

Although CCBP has good theoretical properties, one drawback of the method is that the resulting beliefs may not provide a good approximation of the exact min-marginals (or marginal distributions) in the case of the min-sum (or sum-product) algorithm. For example, traditional BP is guaranteed to compute the exact min-marginals (or marginals) when the graph is acyclic. However, we showed the beliefs obtained with CCBP are the min-marginals of a weighted energy when the graph is acyclic. Although the beliefs may differ from the exact min-marginals of the unweighted energy, they still provide useful information. To illustrate this point, we provide an example of using CCBP to obtain a fixed point in the spin glass model from statistical physics.

Example 3. Let $G = (V, E)$ be a complete graph with twelve nodes and let $\Omega = \{-1, 1\}$ be a set of possible outcomes for each random variable. Consider the energy given by

$$E(x) = \sum_{i \in V} y_i x_i + \sum_{\{i, j\} \in E} \lambda_{ij} x_i x_j$$

Each y_i was independently sampled from a uniform distribution over Ω and each λ_{ij} was independently sampled from a standard normal distribution.

Let $\gamma = 0.9$ and $w_{ki} = 1/(d(i) - 1)$ in CCBP. Let $\alpha = 0.9$ be the damping factor in traditional BP. We use the initialization $\mu_{ij}^{(0)} = (1, 1)$. For each algorithm, we computed the resulting beliefs along with the exact min-marginals of the energy. In Fig. 12, we show the value of these functions when $x_i = 1$. (Note: each function was shifted to have mean zero.)

As seen in Fig. 12 the beliefs obtained with traditional BP oscillate despite the damping factor being quite large. In contrast, we see that the beliefs obtained with CCBP converge. We have seen CCBP obtains good approximations to min-marginals across many problem instances, but there is no theoretical bound on the quality of this approximation. This is an issue that is shared by both convex combination and damped belief propagation (when damped BP converges).

In damped BP, the message passing operator can have multiple fixed points. Some fixed points may be attractive while others may be repulsive which makes the dynamics of fixed point iteration unstable. CCBP is always guaranteed to converge. Although the resulting beliefs may differ from the exact min-marginals, the beliefs provide useful information to obtain a good configuration.

Consider the problem of obtaining an optimal (minimum energy) configuration in Example 3. The beliefs and min-marginals in Fig. 12 were centered to have mean zero. Since the state space consists of exactly two elements each belief is positive for one label and negative for the other. In this framework, the value 0 functions as a decision boundary where favorable states correspond to a negative belief.

The beliefs obtained with damped BP in Example 3 are problematic because they oscillate. In some special cases of oscillations, the beliefs can be useful as long as they choose the same state for the entire period of the oscillation. However, the beliefs in Example 3 do not fall into this category. Instead the beliefs are uninformative because they oscillate about the decision boundary. Given these circumstances, it is impossible to confidently choose a labelling with these beliefs.

Although the beliefs obtained with CCBP differ from the exact min-marginals, they agree on the optimal state for all but one node (see Fig. 12). This highlights a strength of this algorithm, namely that it converges on difficult inference tasks and the beliefs tend to align with the exact min-marginals. Thus, one conclusion to draw from this example is that CCBP provides useful information in decision-based applications.

We conclude by noting that CCBP is not only guaranteed to converge but also converges quickly. As discussed above the resulting beliefs provide useful information in the context of decision making, a central component of an intelligence system. Many applications require powerful algorithms that can handle complex and large scale data sets. CCBP is a natural fit for these applications because it is designed to converge on the most challenging problems.

Acknowledgement

We would like to thank the reviewer who provided very helpful comments that improved the overall structure and content in this work.

Appendix A. Message passing in a metric space

In this appendix we prove the space \mathcal{M} is complete with respect to the metric d from Proposition 1. In the next lemma, we define a distance function over the set of positive reals and prove that this pair is a complete metric space. Then we extend this distance function into a metric that is defined over the product space \mathcal{M} .

Lemma 3. Let $f : \mathbb{R}_+ \times \mathbb{R}_+ \rightarrow \mathbb{R}$ be the distance function given by

$$f(x_i, x_j) = |\log x_i - \log x_j|$$

for any $x_i, x_j \in \mathbb{R}_+$, then the pair (\mathbb{R}_+, f) is a complete metric space.

Proof. It is clear that f is non-negative, symmetric, and that $d(x_i, x_j) = 0$ if and only if $x_i = x_j$. The triangle inequality holds for any $x_i, x_j, x_k \in \mathbb{R}_+$ by

$$\begin{aligned} f(x_i, x_j) &= |\log x_i - \log x_j| \\ &\leq |\log x_i - \log x_k| + |\log x_k - \log x_j| \\ &= f(x_i, x_k) + f(x_k, x_j). \end{aligned}$$

To show that this space is complete, choose any Cauchy sequence $\{x_n\} \subset \mathbb{R}_+$ and note that this sequence can be written as $\{x_n\} = \{e^{y_n}\}$ with $y_n = \log x_n$. The sequence $\{y_n\} \subset \mathbb{R}$ must be Cauchy with respect to the Euclidean metric because for any $\epsilon > 0$ there exists an $N > 0$ such that $f(x_n, x_m) < \epsilon$ for all $n, m > N$, which implies that

$$|y_n - y_m| = |\log x_n - \log x_m| = f(x_n, x_m) < \epsilon.$$

Since $\{y_n\}$ is a Cauchy sequence in a complete space, there exists some $y \in \mathbb{R}$ such that $y_n \rightarrow y$ and hence $x_n = e^{y_n} \rightarrow e^y$. \square

Proposition 3. Let $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ be the distance function given by

$$d(\mu, \nu) = \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}(x_j) - \log \nu_{ij}(x_j)|,$$

then the pair (\mathcal{M}, d) is a complete metric space.

Proof. It is clear that d is non-negative, symmetric, and that $d(\mu, \nu) = 0$ if and only if $\mu = \nu$. To show the triangle inequality, choose any $\mu, \nu, \lambda \in \mathcal{M}$ and observe that

$$\begin{aligned} d(\mu, \nu) &= \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}(x_j) - \log \nu_{ij}(x_j)| \\ &\leq \max_{i \in V} \max_{j \in N(i)} \max_{x_j} \left(|\log \mu_{ij}(x_j) - \log \lambda_{ij}(x_j)| + |\log \lambda_{ij}(x_j) - \log \nu_{ij}(x_j)| \right) \\ &\leq \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}(x_j) - \log \lambda_{ij}(x_j)| + \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \lambda_{ij}(x_j) - \log \nu_{ij}(x_j)| \\ &= d(\mu, \lambda) + d(\lambda, \nu). \end{aligned}$$

Now choose any Cauchy sequence $\{\mu^{(n)}\} \subset \mathcal{M}$, then $\{\mu_{ij}^{(n)}(x_j)\} \subset \mathbb{R}_+$ is a Cauchy sequence in (\mathbb{R}_+, f) because for any $\epsilon > 0$ there exists an $N > 0$ such that for all $n, m > N$

$$\begin{aligned} f(\mu_{ij}^{(n)}(x_j), \mu_{ij}^{(m)}(x_j)) &= |\log \mu_{ij}^{(n)}(x_j) - \log \mu_{ij}^{(m)}(x_j)| \\ &\leq \max_{i \in V} \max_{j \in N(i)} \max_{x_j} |\log \mu_{ij}^{(n)}(x_j) - \log \mu_{ij}^{(m)}(x_j)| \\ &= d(\mu^{(n)}, \mu^{(m)}) \\ &< \epsilon \end{aligned}$$

Given that the pair (\mathbb{R}_+, f) is a complete metric space by Lemma 3, there exists some μ such that $\mu_{ij}^{(n)}(x_j) \rightarrow \mu_{ij}(x_j)$. Thus, the space (\mathcal{M}, d) is complete because $\mu^{(n)} \rightarrow \mu \in \mathcal{M}$ by

$$d(\mu^{(n)}, \mu) = \max_{i \in V} \max_{j \in N(i)} \max_{x_j} f(\mu_{ij}^{(n)}(x_j), \mu_{ij}(x_j)) \rightarrow 0.$$

\square

Appendix B. Sum-product algorithm

In this appendix, we present the sum-product version of CCBP. The operator in the sum-product algorithm is analogous to the one derived from the max-product equations.

Definition 7. The operator $S : \mathcal{M} \rightarrow \mathcal{M}$ in the CCBP sum-product algorithm is

$$(S\mu)_{ij}(x_j) = \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \left(\prod_{k \in N(i) \setminus j} \mu_{ki}(x_i)^{w_{ki}} \right)^\gamma,$$

where the weights must be non-negative with $\sum_{k \in N(i) \setminus j} w_{ki} \leq 1$ and $\gamma \in (0, 1)$

Similar to the case of the max-product operator we show the sum-product CCBP operator is contractive.

Lemma 4. The operator S is contractive with Lipschitz constant γ .

Proof. Choose any $\mu, \nu \in \mathcal{M}$, then

$$\begin{aligned} (S\mu)_{ij}(x_j) &= \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \left(\prod_{k \in N(i) \setminus j} \mu_{ki}(x_i)^{w_{ki}} \right)^\gamma \\ &= \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \nu_{ki}(x_i)^{\gamma w_{ki}} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \\ &\leq \left(\sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} \nu_{ki}(x_i)^{\gamma w_{ki}} \right) \left(\max_{x_i} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \right) \\ &= (S\nu)_{ij}(x_j) \max_{x_i} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}}. \end{aligned}$$

Taking the logarithm of both sides yields that

$$\begin{aligned}
 \log (S\mu)_{ij}(x_j) &\leq \log \left((S\nu)_{ij}(x_j) \max_{x_i} \prod_{k \in N(i) \setminus j} \frac{\mu_{ki}(x_i)^{\gamma w_{ki}}}{\nu_{ki}(x_i)^{\gamma w_{ki}}} \right) \\
 &= \log (S\nu)_{ij}(x_j) + \gamma \max_{x_i} \sum_{k \in N(i) \setminus j} w_{ki} \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \\
 &\leq \log (S\nu)_{ij}(x_j) + \gamma \max_{x_i} \sum_{k \in N(i) \setminus j} w_{ki} \left| \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \right| \\
 \Rightarrow \log (S\mu)_{ij}(x_j) - \log (S\nu)_{ij}(x_j) &\leq \gamma \max_{x_i} \sum_{k \in N(i) \setminus j} w_{ki} \left| \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \right|.
 \end{aligned}$$

Since this inequality holds when μ and ν are interchanged, we can take the absolute value of the left hand side. Moreover, given that the above inequality holds for any $x_j \in \Omega$, it must hold for the maximum over x_j the left hand side.

$$\begin{aligned}
 \max_{x_j} \left| \log \frac{(S\mu)_{ij}(x_j)}{(S\nu)_{ij}(x_j)} \right| &\leq \gamma \max_{x_i} \sum_{k \in N(i) \setminus j} w_{ki} \left| \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \right| \\
 &\leq \gamma \max_{i \in V} \max_{j \in N(i)} \max_{x_i} \sum_{k \in N(i) \setminus j} w_{ki} \left| \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \right| \\
 &\leq \gamma \max_{i \in V} \max_{j \in N(i)} \max_{k \in N(i) \setminus j} \max_{x_i} \left| \log \frac{\mu_{ki}(x_i)}{\nu_{ki}(x_i)} \right| \\
 &= \gamma \max_{i \in V} \max_{j \in N(i)} \max_{x_i} \left| \log \frac{\mu_{ij}(x_i)}{\nu_{ij}(x_i)} \right|.
 \end{aligned}$$

Since the edge $\{i, j\} \in E$ was chosen arbitrarily from the beginning, the inequality holds for any $\{i, j\} \in E$. The final result is obtained by taking the maximum of the left hand side over all the edges in the graph.

$$\begin{aligned}
 \max_{i \in V} \max_{j \in N(i)} \max_{x_i} \left| \log \frac{(S\mu)_{ij}(x_j)}{(S\nu)_{ij}(x_j)} \right| &\leq \gamma \max_{i \in V} \max_{j \in N(i)} \max_{x_i} \left| \log \frac{\mu_{ij}(x_i)}{\nu_{ij}(x_i)} \right| \\
 \Rightarrow d(S\mu, S\nu) &\leq \gamma d(\mu, \nu)
 \end{aligned}$$

□

Theorem 3. The operator S has a unique fixed point $\mu^* \in \mathcal{M}$ and the sequence defined by $\mu^{(n+1)} := S\mu^{(n)}$ converges to μ^* . Furthermore, after n iterations

$$d(S^{(n)}\mu^{(0)}, \mu^*) \leq \gamma^n d(\mu^{(0)}, \mu^*).$$

Proof. The pair (\mathcal{M}, d) is a complete metric space by Proposition 1 and S is a contraction by Lemma 4. The result holds by applying Banach's fixed point theorem. □

References

- [1] G. Cooper, The computational complexity of probabilistic inference using bayesian belief networks, *Artif. Intell.* 42 (1990) 393–405.
- [2] J. Kim, J. Pearl, A computational model for causal and diagnostic reasoning in inference engines, *Proc. Int. Joint Conf. Artif. Intell.* (1983).
- [3] J. Chen, M. Fossorier, Near optimum universal belief propagation based decoding of low-density parity check codes, *IEEE Trans. Commun.* 50 (3) (2002) 406–414.
- [4] M. Fossorier, M. Mihaljevic, H. Imai, Reduced complexity iterative decoding of low-density parity check codes based on belief propagation, *IEEE Trans. Commun.* 47 (5) (1999) 673–680.
- [5] B. Frey, D. MacKay, A revolution: Belief propagation in graphs with cycles, *Advances in Neural Information Processing Systems*, volume 10, MIT Press, 1997.
- [6] K. Murphy, Y. Weiss, M. Jordan, Loopy belief propagation for approximate inference: An empirical study, in: *UAI*, 1999.
- [7] J. Sun, N. Zheng, H. Shum, Stereo matching using belief propagation, *IEEE Trans. Pattern Anal. Mach. Intell.* 25 (7) (2003) 787–800.
- [8] P. Felzenszwalb, B. Svaite, Diffusion methods for classification with pairwise relationships, *Q. Appl. Math.* 77 (2019) 793–810.
- [9] H. Georgii, *Gibbs measures and phase transitions*, de Gruyter, 2011.
- [10] S. Tatikonda, Convergence of the sum-product algorithm, in: *Proceedings 2003 IEEE Information Theory Workshop*, 2003, pp. 222–225.
- [11] S. Tatikonda, M. Jordan, Loopy belief propagation and Gibbs measures, in: *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers Inc., 2002, p. 493500.
- [12] A. Ihler, J. Fisher III, A. Willsky, D. Chickering, Loopy belief propagation: convergence and effects of message errors, *J. Mach. Learn. Res.* 6 (5) (2005).
- [13] T. Heskes, Stable fixed points of loopy belief propagation are local minima of the bethe free energy, *NIPS*, 2002.
- [14] T. Heskes, On the uniqueness of loopy belief propagation fixed points, in: *Neural Computations*, volume 16, 2004, pp. 2397–2413.
- [15] Y. Watanabe, K. Fukumizu, Graph zeta function in the bethe free energy and loopy belief propagation, *NIPS*, 2009.
- [16] J. Yedidia, W. Freeman, Y. Weiss, Constructing free-energy approximations and generalized belief propagation algorithms, *IEEE Trans. Inf. Theory* 51 (7) (2005) 2282–2312.

- [17] V. Martin, M. Lasgouttes, C. Furtlehner, Local stability of belief propagation algorithm with multiple fixed points, in: *Frontiers in Artificial Intelligence and Applications*, volume 241, 2012, pp. 180–191.
- [18] J. Yedidia, W. Freeman, Y. Weiss, Generalized belief propagation, *Adv. Neural Inf. Process Syst.* 13 (2000).
- [19] M. Wainwright, T. Jaakkola, A. Willsky, Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching, *AISTATS*, 2003.
- [20] V. Kolmogorov, Convergent tree-reweighted message passing for energy minimization, *IEEE Trans. Pattern Anal. Mach. Intell.* 28 (10) (2006) 1568–1583.
- [21] T. Roosta, M. Wainwright, S. Sastry, Convergence analysis of reweighted sum-product algorithms, *IEEE Trans. Signal Process.* 56 (9) (2008) 4293–4305.
- [22] C. Knoll, A. Weller, F. Pernkopf, Self-guided belief propagation—a homotopy continuation method, *arXiv preprint arXiv:1812.01339* (2018).
- [23] A. Granas, J. Dugundji, *Fixed Point Theory*, Springer, 2003.
- [24] F. Kschischang, B. Frey, H. Loeliger, Factor graphs and the sum-product algorithm, *IEEE Trans. Inf. Theory* 47 (2) (2001) 498–519.
- [25] A. Blake, A. Zisserman, *Visual Reconstruction*, The MIT Press, 1987.
- [26] P. Felzenszwalb, D. Huttenlocher, Efficient belief propagation for early vision, in: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, 2004.
- [27] S. Geman, D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Trans. Pattern Anal. Mach. Intell.* 6 (6) (1984) 721–741.