LIDS Technical Report P-2510
Laboratory for Information & Decision Systems
MIT, Cambridge, MA 02139
Report available at http://ssg.mit.edu/group/mjwain/mjwain.shtml

Tree-based reparameterization for approximate estimation of stochastic processes on graphs with cycles

Martin Wainwright[†] mjwain@mit.edu
Tommi Jaakkola[*] tommi@ai.mit.edu
Alan Willsky[†] willsky@mit.edu

Lab for Information and Decision Systems[†]
Electrical Engineering & CS
MIT, Cambridge, MA

Lab for Artificial Intelligence[*]
Electrical Engineering & CS
MIT, Cambridge, MA

Original version: May 15, 2001
Revised version: August 25, 2001

Abstract

We present a tree-based reparameterization framework for the approximate estimation of stochastic processes on graphs with cycles. This framework provides a new conceptual view of a large class of iterative algorithms for computing approximate marginals in graphs with cycles. Among them is *belief propagation* (BP), otherwise known as the *sum-product* algorithm, which can be reformulated as a very local form of reparameterization. More generally, this class includes algorithms in which updates are more global, and involve performing exact computations over spanning trees of the full graph. On the practical side, we find that such tree reparameterization (TRP) algorithms typically converge more quickly than belief propagation with equivalent or lower cost per iteration; moreover, TRP often converges on problems for which BP fails.

The reparameterization perspective also provides new theoretical insights into approximate estimation. In particular, it leads to a new probabilistic characterization of fixed points in terms of a consistency condition over embedded trees. We develop the geometry of tree-based reparameterization, and use it to develop sufficient conditions for convergence in the case of two spanning trees. A fundamental property of reparameterization updates is that they leave invariant the distribution on the full graph. This invariance, in conjunction with our fixed point characterization, enables us to derive an exact relation between the true marginals on an arbitrary graph with cycles, and the approximations provided by TRP or BP. We also develop bounds on this approximation error, which illuminate the conditions that govern performance of such techniques. Our results also have natural extensions to approximations (e.g., Kikuchi) that involve clustering nodes.

Keywords: graphical models; belief propagation; sum-product algorithm; approximate inference; Bethe free energy; Kikuchi; decoding; codes on graphs.

MW was supported in part by a 1967 Fellowship from the Natural Sciences and Engineering Research Council of Canada; MW, AW and TJ were supported in part by ODDR&E MURI Grant DAAD19-00-1-0466 through the Army Research Office; by the Office of Naval Research through Grant N00014-00-1-0089; and by the Air Force Office of Scientific Research through Grant F49620-00-1-0362.

1 Introduction

Stochastic processes defined on graphs arise in a variety of fields, including coding theory [21, 30], artificial intelligence [38], statistical physics [37], system theory [7] and statistical image processing [33]. Given a graphical model, one important problem is computing marginal distributions of variables at each node on the graph. For acyclic graphs (i.e., trees), standard and highly efficient algorithms exist for this task. In contrast, exact solutions are prohibitively complex for more general graphs of any substantial size [13]. As a result, there has been considerable interest and effort aimed at developing approximate inference algorithms for large graphs with loops.

Perhaps the best-known and most widely studied [e.g., 2, 41, 48, 34, 52] approximation method is that known variously as *belief propagation* in the graphical model community [38], and the *sum-product algorithm* in coding theory [e.g., 34, 30]. The interest in this algorithm has been fueled in part by its use in fields such as artificial intelligence and computer vision [e.g., 36, 20], and also by the success of turbo codes and other compound codes, for which the decoding algorithm is a particular instantiation of belief propagation [e.g., 21, 34, 30]. While there are various equivalent forms for belief propagation [38], the best known formulation, which we refer to here as the BP algorithm, entails the exchange of statistical information among neighboring nodes via message-passing. If the graph is a tree, the resulting algorithm can be shown to produce exact solutions in a finite number of iterations. The message-passing formulation is thus equivalent to other techniques for optimal inference on trees, some of which involve more global and efficient computational procedures. On the other hand, if the graph contains cycles, then it is the local message-passing algorithm that is most generally applicable. It is well-known that the resulting algorithm may not converge; moreover, when it does converge, the quality of the resulting approximations varies substantially.

Recent work has yielded some insight into the dynamics and convergence properties of BP. For example, several researchers [5, 1, 28, 48] have analyzed the single loop case, where belief propagation can be reformulated as a matrix powering method. For Gaussian processes on arbitrary graphs, two groups [49, 43], using independent methods, have shown that when BP converges, then the conditional means are exact but the error covariances are generally incorrect. For the special case of graphs corresponding to turbo codes, Richardson [41] developed a geometric approach, through which he was able to establish the existence of fixed points, and give conditions for their stability. More recently, Yedidia et al. [52] showed that BP can be viewed as performing a constrained minimization of the so-called Bethe free energy associated with the graphical distribution,[1] which inspired other researchers [e.g., 50, 53] to develop more sophisticated algorithms for the minimization of the Bethe free energy. Yedidia et al. also proposed extensions to BP based on cluster variational methods [29]; related extensions using higher order structures have been proposed by Minka [35]. These advances notwithstanding, much remains to be understood about the behavior of this algorithm, and more generally about other (perhaps superior) approximation algorithms.

This important area constitutes the focus of this paper. In particular, the framework presented in this paper provides a new conceptual view of a large class of iterative algorithms that includes BP. Central to the framework is the idea of performing exact computations over acyclic subgraphs embedded within a graph with cycles. This idea was exploited in [46] to develop an iterative algorithm for exact estimation of Gaussian processes on graphs. One of the motivations for the work described here is to show how such tree-based updates can also be applied to discrete processes

[1] Several researchers have investigated the utility of Bethe tree approximations for graphical models; we refer the reader to [e.g., 45, 51].

on graphs with cycles.

A key idea in graphical models is the representation of a probability distribution as a product of factors, each of which involves variables only at a subset of nodes corresponding to a clique of the graph. Such factorized representations are far from unique, which suggests the goal of seeking a *reparameterization* of the distribution consisting of factors that correspond, either exactly or approximately, to the desired marginal distributions. If the graph is cycle-free (i.e., a tree), then there exists a unique reparameterization specified by exact marginal distributions over cliques. Indeed, such a parameterization is the cornerstone of the junction tree representation [e.g., 31, 25].

For a graph with cycles, on the other hand, exact factorizations exposing these marginals do not generally exist. Nevertheless, it is always possible to reparameterize certain *portions* of any factorized representation — namely, any subset of factors corresponding to a cycle-free subgraph of the original graph. We are thus led to consider iterative reparameterization of different subsets, each corresponding to a singly-connected subgraph. As we will show, BP can be interpreted in exactly this manner, in which each reparameterization takes place over the extremely simple subgraph consisting of a pair of neighboring nodes. One of the consequences of this interpretation is a more storage-efficient "message-free" implementation of BP.

More significantly, this interpretation suggests a more general class of updates where reparameterization is performed over arbitrary cycle-free subgraphs. Although the choice of cycle-free subgraphs is arbitrary, in this paper we will focus primarily on updates involving maximal cycle-free subgraphs — that is, spanning trees. We refer to this class as *tree-based reparameterization* (TRP) algorithms. Since each update on a spanning tree propagates information globally to each node of the graph, one might expect a TRP algorithm to have better convergence properties than the purely local two-node updates of BP. Indeed, experimentation with TRP supports this conclusion: when applied to problems for which BP converges, TRP typically converges at least as quickly, and for many problems much more quickly. More importantly, we find that TRP converges in cases where BP does not.

At one level, the more global updates of TRP can be viewed as a schedule for message-passing based on spanning trees (though with a more efficient implementation via reparameterization). Indeed, one of the practical contributions of this paper is to demonstrate that such tree-based updates have convergence properties superior to those of BP. At a more abstract level, however, the reparameterization perspective leads to a number of new conceptual insights, including a probabilistic characterization of fixed points; and an invariance intrinsic to the TRP or BP algorithms. These two properties, when applied in conjunction, allow us to characterize the approximation error. Many of our results, though not obvious from the more traditional message-passing viewpoint, follow in a natural way from the reparameterization framework.

In the next section, we introduce the background and notation that underlies our development. In the process, we illustrate how distributions over cycle-free graphs can be reparameterized in terms of local marginal distributions. In Section 3, we introduce the class of TRP algorithms. In this context, it is convenient to represent distributions in an exponential form using an overcomplete basis. Our choice of an overcomplete basis, though unorthodox, makes the idea of reparameterization more transparent, and easily stated. In this section, we also show an equivalent formulation of BP as a sequence of local reparameterizations. Moreover, we present some experimental results illustrating the benefits of more global TRP updates, which include a greater range of problems for which convergence is obtained, as well as increased speed of convergence.

Section 4 contains analysis of the fixed points of the TRP updates, as well as the question of

convergence. Central to the analysis is a geometric characterization of successive iterates, which reveals interesting links between tree-based reparameterization and successive projection algorithms for constrained minimization of Bregman distances [e.g., 11]. On this basis, we show that fixed points of the TRP algorithm satisfy the necessary conditions to be a local minimum of a certain cost function that arises as an approximation to the Kullback-Leibler divergence. This result allows us to make contact with the work of Yedidia et al. [52]. Specifically, we show that although the cost function minimized by our TRP algorithms is not the same as the Bethe free energy, TRP fixed points do coincide with extremal points of the Bethe free energy (i.e., with the fixed points of BP). An important benefit of our formulation is a new and intuitive characterization of the fixed points: in particular, any fixed point must be consistent, in a suitable sense to be defined, with respect to any singly-connected subgraph; and at least one such fixed point of this type is guaranteed to exist. In addition, the geometric viewpoint also allows us to formulate sufficient conditions for convergence in the case of TRP using two spanning trees.

A fundamental property of our reparameterization updates is that they leave invariant the distribution on the full graph. This result has a number of important consequences, which are also developed in Section 4. For example, by adapting this invariance to the Gaussian (as opposed to discrete) case, we obtain a short and elementary proof of the exactness of the means when BP/TRP converges.

The final topic in this paper is the analysis of the approximation error arising from application of TRP or BP. Previous results on this error have been obtained in certain special cases. For a single loop, Weiss [48] derived a relation between the exact marginals and the BP approximations, and for a binary processes showed how local corrections could be applied to compute the exact marginals. Empirically, he also observed that in certain cases, approximation accuracy is correlated with the convergence rate of BP. In the context of turbo decoding, Richardson [41] provided a heuristic analysis of the associated error. Despite these encouraging results, a deep and broadly applicable understanding of the approximation error remains a challenging and important problem. Our characterization of the TRP/BP fixed points, in conjunction with the invariance property, allows us to contribute to this goal by analyzing the approximation error for arbitrary graphs. In particular, our development in Section 5 begins with the derivation of an exact relation between the correct marginals and the approximate marginals computed by TRP or BP. We then exploit this exact relation to derive both upper and lower bounds on the approximation error. The interpretation of these bounds provides an understanding of the conditions that govern the performance of approximation techniques like TRP or BP. The paper concludes in Section 6 with a summary, as well as a discussion of extensions and open questions associated with the reparameterization framework.

2 Preliminaries

2.1 Basics of graphical models

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of nodes or vertices $\mathcal{V} = \{1, \ldots, n\}$ that are joined by a set of edges $\mathcal{E}$. Lying at each node $s \in \mathcal{V}$ is a random variable x_s. For the bulk of this paper, we focus on random variables that take values in the discrete alphabet $\{0, \ldots, m-1\}$. The underlying sample space $\mathcal{X}$ is the set of all n vectors $x = \{x_s \mid s \in \mathcal{V}\}$ over m symbols, so that $|\mathcal{X}| = m^n$. The edges in a graph represent probabilistic dependencies between variables, and come in two varieties —

directed or undirected. Although the probabilistic interpretation of directed and undirected edges is different, any directed graph can be converted to an equivalent[2] undirected graph [see, e.g., 38]. Herein we restrict our attention to undirected graphs.

The stochastic processes of interest are those which are Markov with respect to the underlying graph. To define this concept, let $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ be subsets of the vertex set $\mathcal{V}$. Let $x_{\mathcal{A}|\mathcal{B}}$ be the random variables in $\mathcal{A}$ conditioned on those in $\mathcal{B}$. The set $\mathcal{B}$ separates $\mathcal{A}$ and $\mathcal{C}$ if in the modified graph with $\mathcal{B}$ removed, there are no paths between nodes in the sets $\mathcal{A}$ and $\mathcal{C}$ (see Figure 1).

Definition 2.1. A stochastic process x is *Markov* with respect to the graph $\mathcal{G}$ if $x_{\mathcal{A}|\mathcal{B}}$ and $x_{\mathcal{C}|\mathcal{B}}$ are conditionally independent whenever $\mathcal{B}$ separates $\mathcal{A}$ and $\mathcal{C}$.

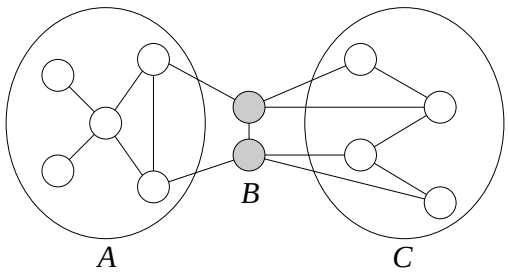


Figure 1. Illustration of the relation between conditional independence and graph separation. Here the set of nodes $\mathcal{B}$ separates $\mathcal{A}$ and $\mathcal{C}$, so that $x_{\mathcal{A}|\mathcal{B}}$ and $x_{\mathcal{C}|\mathcal{B}}$ are conditionally independent.

A graph strongly constrains the distribution of a Markov process. To express this fact, we introduce the notion of a graph *clique*, which is any fully connected subset of $\mathcal{V}$; a clique is maximal if it is not properly contained within any other clique. The *Hammersley-Clifford theorem* [9, 22] then guarantees that distributions of Markov processes over graphs can be expressed in factorized form as products of so-called *compatibility functions* defined over the cliques:

Theorem 2.1. Let $\mathcal{G}$ be a graph with a set of cliques $\mathbf{C}$. Suppose that a distribution[3] p is formed as a normalized product of nonnegative functions over the cliques:

$$p(x) = \frac{1}{Z} \prod_{\mathcal{C} \in \mathbf{C}} \psi_{\mathcal{C}}(x) \tag{1}$$

where $\psi_{\mathcal{C}}(x)$ is a compatibility function depending only on the subvector $x_{\mathcal{C}} = \{x_s \mid s \in \mathcal{C}\}$; and $Z \triangleq \sum_x \prod_{\mathcal{C} \in \mathbf{C}} \psi_{\mathcal{C}}(x)$ is the partition function. Then the underlying process x is Markov with respect to the graph. Conversely, the distribution p of any Markov random field over $\mathcal{G}$ that is strictly positive (i.e., $p(x) > 0$ for all $x \in \mathcal{X}$) can be represented in this factorized form.

Note that each individual node forms a singleton clique, so that some of the factors in (1) may involve functions of each individual variable. As a consequence, if we have independent measurements y_s of x_s at some (or all) of the nodes, then Bayes' rule implies that the effect of including these measurements — i.e., the transformation from the prior distribution $p(x)$ to the conditional distribution

[2]However, after converting to an undirected graph, it may no longer be possible to read certain independence properties directly from the graph.

[3]Strictly speaking, p is a probability mass function for discrete random variables; however, we will use distribution to mean the same thing.

$p(x \mid y)$ — is simply to modify the singleton factors of equation (1). As a result, throughout this paper, we suppress explicit mention of measurements, since the problem of computing marginals for either $p(x)$ or $p(x \mid y)$ are of identical structure and complexity.

Furthermore, primarily for simplicity of exposition, we assume that the maximal cliques of our graph have cardinality two. At a purely formal level, this assumption entails no loss of generality since it is always possible to cluster the nodes of any graph so as to form an equivalent graph with a maximal clique size of two [e.g., 19]. However, modifying the graph so as to create pairwise cliques can lead to aggregated nodes with excessively high state cardinalities, so that clustering may not be helpful in a practical sense. Nevertheless, we focus on the case of pairwise cliques both because of its importance in many problems of interest, and because it is straightforward to extend our reparameterization framework to include larger cliques, as in cluster variational methods [e.g., 29].

Given only singleton and pairwise cliques, the clique index set ranges over all edges $(s,t) \in \mathcal{E}$, as well as the singleton cliques $\{s\}$. Figure 2(a) gives an example of the assignment of pairwise compatibility functions ψ_{st} and single-node functions ψ_s. With a minor abuse of notation, the quantity ψ_{st} lying on the edges (s,t) can be thought of as a $m \times m$ matrix, where the (j,k) element $\psi_{st;jk}$ is equal to the function value of ψ_{st} for $\{x_s = j, x_t = k\}$. Similarly, the observation functions ψ_s can be thought of as an m-vector, where the j^{th} component $\psi_{s;j}$ equals the value of ψ_s for $\{x_s = j\}$.

2.2 Estimation in graphical models

Given a graphical model as in equation (1), the problem of central interest in this paper is the computation of the marginal probabilities $P_s = p(x_s)$ at each node of the graph. This task, which in this paper will be called *optimal estimation or inference*, is NP-hard [13], since it requires summations involving exponentially many terms. As indicated previously, for tree-structured graphs, there exist direct algorithms for optimal estimation. For graphs with cycles, suboptimal algorithms (such as BP) are used in an attempt to compute approximations to the desired marginals. In this section, we elaborate on both of these topics.

2.2.1 Exact estimation on trees as reparameterization

Algorithms for optimal estimation on trees have appeared in the literature of various fields, including coding theory [30], artificial intelligence [38], and system theory [7]. In broad overview, such algorithms consist of a recursive series of updates, in which "messages" are passed from node to node. The most efficient implementation of these algorithms follows a two-pass form, first sweeping upwards from the leaves to a node designated as the root, and then downwards from the root to leaves, with an overall computational complexity of $\mathcal{O}(m^2 n)$.

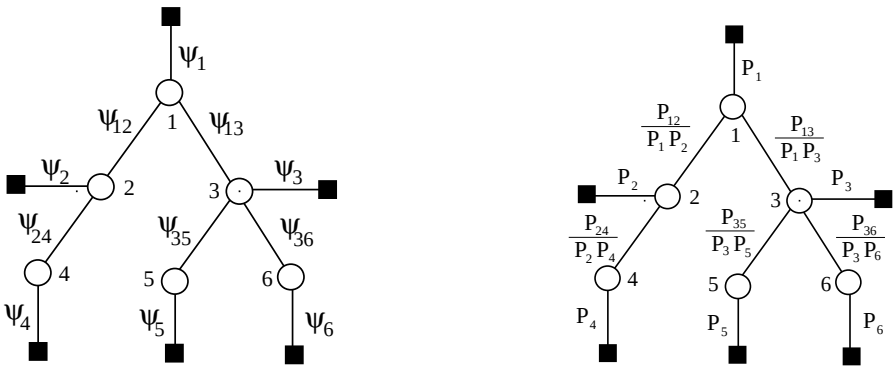
It is worth noting that tree inference algorithms can, in principle, be applied to any graph by clustering nodes so as to form a so-called *junction tree* [e.g., 31, 25]. However, in many cases of interest, the aggregated nodes of the junction tree have exponentially large state cardinalities, meaning that applying tree algorithms is prohibitively complex. This explosion in the state cardinality is another demonstration of the intrinsic complexity of exact computations for graphs with cycles.

An important observation that arises from the junction tree perspective is that any exact algorithm for optimal estimation on trees actually computes marginal distributions for pairs (s,t) of neighboring nodes (see, for example, the reformulation of BP given in Section 3.4). In doing so, it

produces an alternative factorization of the distribution $p(x)$, namely:

$$p(x) \quad = \quad \prod_{s \in \mathcal{V}} P_s \quad \prod_{(s,t) \in \mathcal{E}} \frac{P_{st}}{P_s P_t} \tag{2}$$

where $P_{st} = p(x_s, x_t)$. As an illustration, Figure 2(a) shows a simple example of a tree, labeled in terms of compatibility functions ψ_s and ψ_{st}, which leads to the factorization in equation (1). Figure 2(b) shows this same tree, now *reparameterized* in terms of the local marginal distributions P_s and P_{st}. The representation of equation (2) can be deduced from a more general factorization result on junction trees [e.g. 25, 31].[4] We thus arrive at an alternative interpretation of exact



(a) Original parameterization. $p(x) = \frac{1}{Z} \prod_{s \in \mathcal{V}} \psi_s \prod_{(s,t) \in \mathcal{E}} \psi_{st}$

(b) Final (desired) parameterization. $p(x) = \prod_{s \in \mathcal{V}} P_s \prod_{(s,t) \in \mathcal{E}} \frac{P_{st}}{P_s P_t}$

Figure 2. A simple example of a graphical model; circles correspond to state variables x_s, whereas squares correspond to observations. (a) Original parameterization of distribution on the tree in terms of compatibility functions ψ_{st} and ψ_s. (b) Final parameterization in terms of marginal probabilities P_s and joint probabilities P_{st}.

inference on trees: it entails computing a reparameterized factorization of the distribution $p(x)$ that explicitly exposes the local marginal distributions; and also does not require any additional normalization (i.e., with partition function $Z = 1$).

2.2.2 Belief propagation for graphs with cycles

As we have indicated, the message-passing form of BP, in addition to being exact in application to trees, yields an iterative message-passing algorithm for graphs with cycles. In this section, we

[4] Alternatively, equation (2) can be seen as a symmetrized generalization of the well-known factorization(s) of Markov chains. For example, the variables at the three nodes $\{1, 2, 4\}$ in Figure 2(b) form a simple Markov chain, meaning that the joint distribution can be written as

$$\begin{aligned} P_{124} &= P_1 \, (P_{2|1})(P_{4|2}) \\ &= P_1 \, (P_{12}/P_1)(P_{24}/P_2) \\ &= P_1 P_2 P_4 \, (P_{12}/P_1 P_2)(P_{24}/P_2 P_4) \end{aligned}$$

where the last equality is precisely the form of equation (2). Note that the final line removes the asymmetry present in those that precede it (which resulted from beginning the factorization from node 1, as opposed to node 2 or 4).

summarize for future reference the equations governing the BP dynamics. The message passed from node s to node t, denoted by M_{st}, is an m-vector in which element $M_{st;k}$ gives its value when $x_t = k$. Let $\mathcal{N}(s) = \{t \in \mathcal{V} \mid (s,t) \in \mathcal{E}\}$ be the set of neighbors of s in $\mathcal{G}$. With this notation, the message at iteration $(n+1)$ is updated based on the messages at the previous iteration n as follows:

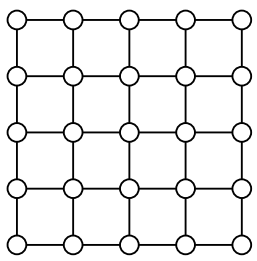
$$M_{st;k}^{n+1} \quad = \quad \eta \sum_{j=1}^{m} \psi_{st;jk} \psi_{s;j} \prod_{u \in \mathcal{N}(s)/t} M_{us;j}^{n} \tag{3}$$

where η denotes a normalization constant.[5] At any iteration, the "beliefs" — that is, approximations to the marginal distributions — are given by

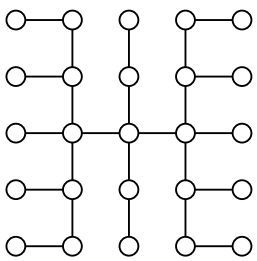
$$B_{s;j}^{n} \quad = \quad \eta \, \psi_{s;j} \prod_{u \in \mathcal{N}(s)} M_{us;j}^{n} \tag{4}$$

3 Tree-based reparameterization framework

Key to TRP is the concept of an *embedded tree* within an arbitrary graph $\mathcal{G}$ with cycles — i.e., a tree formed by removing edges from the graph. A *spanning tree* is an embedded tree that connects all nodes of the original graph. Figure 3 illustrates these definitions: panel (a) shows a nearest



(a) Original graph with cycles



(b) Example of a spanning tree

Figure 3. Illustration of spanning trees. (a) Original graph is a nearest neighbor grid. (b) One particular choice of spanning tree for the grid.

neighbor grid, whereas panel (b) illustrates an embedded spanning tree. Of course, panel (b) is just one example of a spanning tree embedded in the original graph (a). Indeed, a graph generally has a (large) number of spanning trees, and we exploit this fact in our work. Specifically, suppose that $\mathcal{S}^0, \ldots, \mathcal{S}^{L-1}$ (with corresponding edge sets $\mathcal{E}^0, \ldots, \mathcal{E}^{L-1} \subset \mathcal{E}$) is a given set of spanning trees for the graph $\mathcal{G}$. Then for any $i \in \{0, \ldots, L-1\}$, the distribution $p(x)$ of a stochastic process over $\mathcal{G}$ (as in equation (1)) can be factored as:

$$p(x) \quad = \quad p^i(x) \, q^i(x) \tag{5}$$

where $p^i(x)$ includes the factors in equation (1) corresponding to cliques of $\mathcal{S}^i$, and $q^i(x)$ absorbs the remaining terms, corresponding to edges in $\mathcal{E}/\mathcal{E}^i$ removed to form $\mathcal{S}^i$.

[5] Throughout this paper, we will use η to refer to an arbitrary normalization constant, the definition of which may change from line to line. In all cases, it is easy to determine η by local calculations.

Because $\mathcal{S}^i$ is a tree, the reparameterization operation in equation (2) can be applied to $p^i(x)$ in order to obtain an alternative factorization of the distribution $p^i(x)$. With reference to the full graph $\mathcal{G}$ and distribution $p(x)$, this operation simply modifies the compatibility functions for cliques in $\mathcal{G}$, but without modifying the actual distribution $p(x)$. In a subsequent update using this new set of functions and choosing a *different* tree $\mathcal{S}^j$, we can write $p(x) = p^j(x)q^j(x)$, where $p^j(x)$ includes compatibility functions over cliques in $\mathcal{S}^j$. We can then perform reparameterization for $p^j(x)$, and repeat the process, choosing one of the $\mathcal{S}^i$ at each step of the iteration.

Figure 4 illustrates the basic steps of this procedure for a simple graph with cycles. Panel (a) shows the original parameterization of $p(x)$ in terms of compatibility functions ψ_s and ψ_{st}, as in equation (1). A spanning tree, formed by removing edges $(4,5)$ and $(5,6)$, is shown in panel (b): that is, $q^i(x) = \psi_{45}\ \psi_{56}$ in this case. The tree distribution $p^i(x)$, corresponding to the product of all the

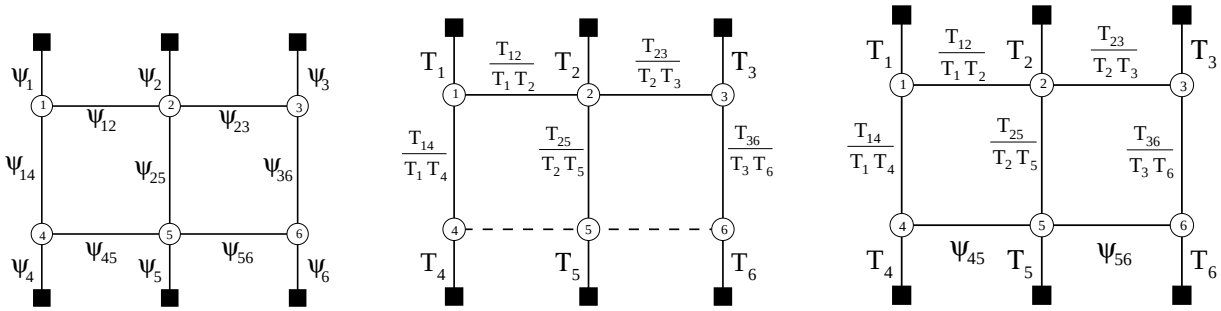


Figure 4. Illustration of TRP update. (a) Original parameterization in terms of compatibility functions ψ_s and ψ_{st}. (b) Tree reparameterization update on the spanning tree. (c) New parameterization after a single iteration.

other compatibility functions, is reparameterized in terms of marginals T_s and T_{st} computed from the tree $\mathcal{S}^i$. The quantities $\{T_s, T_{st}\}$ are exact marginals for the tree, but represent approximations to the true marginals $\{P_s, P_{st}\}$ of the full loopy graph. The graph compatibility functions after this first update are shown in panel (c). In a subsequent update, a different tree is chosen over which reparameterization is to be performed.

As should be clear from the preceding discussion, each step of the algorithm[6] reparameterizes the density $p(x)$ but *does not* modify it (aside from the normalization constant). To formalize this basic idea, in this section we introduce a particular parameterization of distributions $p(x;\theta)$, such that iterations of the type just described can be represented as explicit functional updates $\theta^n \mapsto \theta^{n+1}$ on these parameters. We also show that BP iterations can be interpreted as reparameterization operations using especially simple non-spanning embedded trees, and we present experimental results illustrating the potential advantages of TRP over BP.

[6]Here we have described an unrelaxed form of the updates; in the sequel, we present and analyze a suitably relaxed formulation.

3.1 Exponential families of distributions

Central to our work is the following exponential representation of a distribution on a graph

$$p(x;\theta) = \exp\Big\{\sum_{\alpha}\theta_\alpha\phi_\alpha(x) - \Phi(\theta)\Big\} \tag{6a}$$

$$\Phi(\theta) = \log\Big(\sum_{x}\exp\{\sum_{\alpha}\theta_\alpha\phi_\alpha(x)\}\Big) \tag{6b}$$

where Φ is the *log partition function* that serves to normalize the distribution. Equation (6) defines a family of Gibbs distributions on the graph, specified in terms of a set of functions $\phi_\alpha(x)$ and the vector of parameters $\theta = \{\theta_\alpha\}$. Such exponential families of distributions have been studied extensively in statistics [e.g., 12, 6, 3, 23], and also in the specific context of graphical models [e.g., 4], though to a lesser extent.

The log partition function Φ has a number of useful properties that we will exploit in subsequent analysis. In particular, by straightforward calculations, we obtain:

$$\frac{\partial\Phi}{\partial\theta_\alpha}(\theta) = \mathbb{E}_\theta[\phi_\alpha] \tag{7a}$$

$$\frac{\partial^2\Phi}{\partial\theta_\alpha\,\partial\theta_\beta}(\theta) \triangleq \mathrm{cov}_\theta\{\phi_\alpha,\,\phi_\beta\} = \mathbb{E}_\theta[\phi_\alpha\,\phi_\beta] - \mathbb{E}_\theta[\phi_\alpha]\mathbb{E}_\theta[\phi_\beta] \tag{7b}$$

where $\mathbb{E}_\theta$ denotes the expectation taken over x with respect to the distribution $p(x;\theta)$; that is, $\mathbb{E}_\theta[\phi_\alpha] = \sum_x p(x;\theta)\phi_\alpha(x)$. We note that the quantity in (7b) is an element of the Fisher information matrix $(-\mathbb{E}_\theta\{\frac{\partial^2\log p(x;\theta)}{\partial\theta^2}\})$. Therefore, the Hessian $\nabla^2\Phi$ is positive semi-definite, so that Φ is a convex function of θ.

In addition, the exponential parameterization of equation (6) induces a certain form for the Kullback-Leibler (KL) divergence [14] that will be useful in the sequel. Given two parameter vectors θ and θ^*, we denote by $D(\theta \parallel \theta^*)$ the KL divergence[7] between the distributions $p(x;\theta)$ and $p(x;\theta^*)$. This divergence can be written in the following form:

$$D(\theta \parallel \theta^*) = \sum_{\alpha}\mathbb{E}_\theta[\phi_\alpha]\big[\theta - \theta^*\big]_\alpha + \Phi(\theta^*) - \Phi(\theta) \tag{8}$$

It remains to specify a choice of functions $\{\phi_\alpha\}$. Let $s, t \in \mathcal{V}$ be indices parameterizing the nodes of the graph, and let the indices j, k run over the m possible states of the discrete random variables. We then take the index set for α, denoted by $\mathcal{A}$, to be the set of pairs $(s;j)$ or 4-tuples $(st;jk)$, and choose the potentials ϕ_α as indicator functions for x to take on the indicated value (or values) at the indicated node (or pair of nodes). That is,

$$\phi_\alpha(x) = \delta(x_s = j) \quad \text{for } \alpha = (s;j) \tag{9a}$$

$$\phi_\alpha(x) = \delta(x_s = j)\delta(x_t = k) \quad \text{for } \alpha = (st;\,jk) \tag{9b}$$

Here, the indicator or delta function $\delta(x_s = j)$ is equal to 1 when node x_s takes the state value j, and 0 otherwise. With this choice of $\{\phi_\alpha\}$, the length of θ is given by

$$d(\theta) = mn + m^2e \tag{10}$$

[7] This definition entails a minor abuse of notation, since the divergence applies to distributions $p(x;\theta)$ and $p(x;\theta^*)$, and not the parameters θ and θ^* themselves.

where e is the number of edges in $\mathcal{G}$.

It is typical to choose a linearly independent collection of functions $\{\phi_\alpha\}$, which gives rise to a so-called *minimal* representation [e.g., 6]. In this context, the exponential parameterization of equation (9) is unorthodox because it is *overcomplete* (i.e., there are linear relations among the functions $\{\phi_\alpha\}$).[8] As an example, for any edge $(s,t) \in \mathcal{E}$, we have the linear dependence $\sum_j \delta(x_s = j)\delta(x_t = k) = \delta(x_t = k)$ for all $k = 0, \ldots, m-1$. An important consequence of overcompleteness is the existence of distinct parameter vectors $\theta \neq \theta^*$ that induce the same distribution (i.e., $p(x;\theta) = p(x,\theta^*)$). This many-to-one correspondence between parameters and distributions is of paramount importance to our analysis because it permits reparameterization operations that leave the overall distribution unchanged.

3.2 Basic operators

Given a distribution $p(x;\theta)$ defined by a graph $\mathcal{G}$, the quantities that we wish to compute are elements of the $d(\theta)-$dimensional *marginal probability vector* $P = \{P_\alpha \mid \alpha \in \mathcal{A}\}$, i.e., the single node marginals $P_{s;j} = p(x_s = j;\theta)$ for $s \in \mathcal{V}$ (with P_s denoting the $m-$vector of values $\{P_{s;j},\ j = 1, \ldots, m\}$); and pairwise joint distributions $P_{st;jk} = p(x_s = j,\ x_t = k;\ \theta)$ for edges $(s,t) \in \mathcal{E}$ (with the m^2-vector P_{st} defined analogously). Such vectors can be viewed as an alternative set of parameters for a graphical distribution.[9] We will frequently need to consider mappings between these two parameterizations. In particular, the computation of the marginals can be expressed compactly as a map acting on the parameter vector θ:

$$P \quad = \quad \Lambda(\theta) \tag{11}$$

Note that the range of Λ, denoted $Ra(\Lambda)$, is a highly constrained set of vectors. First of all, any $T \in Ra(\Lambda)$ must belong to the unit hypercube $(0,1)^{d(\theta)}$. Secondly, there are normalization constraints (single-node and joint marginal probabilities must sum to one); and marginalization constraints (pairwise joint distributions, when marginalized, must be consistent with the single node marginals). That is, $Ra(\Lambda) \subseteq \mathcal{C}$, where

$$\mathcal{C} \quad = \quad \Big\{ T \;\Big|\; T \in (0,1)^{d(\theta)};\ \sum_j T_{s;j} = 1 \text{ for } s \in \mathcal{V};\ \sum_k T_{st;jk} = T_{s;j} \text{ for } (s,t) \in \mathcal{E} \Big\} \tag{12}$$

If the graph $\mathcal{G}$ is a tree, then the *local consistency* conditions defining $\mathcal{C}$ guarantee the existence of a unique distribution $p(x;\theta)$ that is *globally consistent*; that is, local consistency implies global consistency for trees [see, e.g., 25]. In fact, when $\mathcal{G}$ is a tree, then given any $T \in \mathcal{C}$, there exists a unique distribution $p(x;\theta)$ such that $T = \Lambda(\theta)$, so that $Ra(\Lambda) = \mathcal{C}$. For a graph with cycles, in contrast, $Ra(\Lambda)$ is strictly contained within $\mathcal{C}$.[10] This strict containment reflects the fact that for a graph with cycles, local consistency is no longer sufficient to guarantee the existence of a globally consistent distribution.

[8] There are also nonlinear relations, but it is the linear dependencies that distinguish between minimal and overcomplete representations.

[9] In fact, the quantities P and θ are a dual set of parameters, related via the Legendre transform [42] applied to the log partition function.

[10] Indeed, for graphs with cycles, there exist elements of $\mathcal{C}$ that cannot be realized as the marginals of any distribution (Markov or otherwise).

For a general graph with cycles, of course, the computation of $\Lambda(\theta)$ in equation (11) is very difficult. Indeed, algorithms like BP and TRP can be formulated as iteratively generating approximations to $\Lambda(\theta)$. To make a sharp distinction from exact marginal vectors $P \in Ra(\Lambda) \subseteq \mathcal{C}$, we use the symbol T to denote such *pseudomarginal vectors* — i.e., vectors that belong the unit hypercube $(0,1)^{d(\theta)}$, but need not satisfy the marginal constraints (i.e., $\sum_k T_{st;jk} = T_{s;j}$) required for membership in $\mathcal{C}$.

We will also make use of the following mapping that is defined for any such T:

$$\big[\Theta(T)\big]_\alpha \quad = \quad \begin{cases} \log T_{s;j} & \text{if } \alpha = (s;j) \in \mathcal{A} \\ \log\Big[T_{st;jk} / (\sum_j T_{st;jk})(\sum_k T_{st;jk})\Big] & \text{if } \alpha = (st;jk) \in \mathcal{A} \end{cases} \tag{13}$$

The quantity $\Theta(T)$ can be viewed as an exponential parameter vector that indexes a distribution $p(x;\Theta(T))$ on the graph $\mathcal{G}$. In fact, consider a marginal vector $P \in Ra(\Lambda)$. If $\mathcal{G}$ is a tree, then not only is the computation of (11) simple, but we are also guaranteed $\Theta(P)$ indexes the same graphical distribution as that corresponding to the marginal vector P — that is:

$$\Lambda(\Theta(P)) = P \tag{14}$$

This equality is simply a restatement of the factorization of equation (2) for any tree-structured distribution in terms of its single-node and joint pairwise marginals. However, if $\mathcal{G}$ has loops, then in general the marginal distributions of $p(x;\Theta(P))$ need not agree with the original marginals P (i.e., the equality of equation (14) does not hold). In fact, determining the exponential parameters corresponding to P for a loopy graph is as difficult as the computation of $\Lambda(\theta)$ in equation (11). Thus, the composition of operators $\Lambda \circ \Theta$, mapping one marginal vector to another, is the identity for trees but not for general graphs.

Alternatively, we can consider composing Θ and Λ in the other order:

$$\mathcal{R}(\theta) \quad = \quad \big(\Theta \circ \Lambda\big)(\theta) \tag{15}$$

which defines a mapping from one exponential parameter vector to another. For a general graph, the operator $\mathcal{R}$ will alter the distribution (that is, $p(x;\theta) \neq p(x;\mathcal{R}(\theta))$). For a tree-structured graph, while $\mathcal{R}$ is not the identity mapping, it does leave the probability distribution unchanged; indeed, applying $\mathcal{R}$ corresponds to shifting from the original parameterization of the tree distribution in terms of θ to a new exponential parameter $\mathcal{R}(\theta)$ that corresponds directly to the factorization of equation (2). As a result, in application to trees, the operator $\mathcal{R}$ is idempotent (i.e., $\mathcal{R} \circ \mathcal{R} = \mathcal{R}$).

3.3 Tree-based reparameterization updates

The basic idea of TRP is to perform reparameterization updates on a set of spanning trees $\mathcal{S}^0, \ldots, \mathcal{S}^{L-1}$ in succession. The update on any given spanning tree $\mathcal{S}^i$ involves only a subset $\mathcal{A}^i = \{(s;j),\ (st;jk) \mid s \in \mathcal{V},\ (s,t) \in \mathcal{E}^i\}$ of all the elements of θ. To move back and forth between parameter vectors on the full graph and those on spanning tree $\mathcal{S}^i$, we define projection and injection operators

$$\Pi^i(\theta) = \{\theta_\alpha \mid \alpha \in \mathcal{A}^i\} \tag{16a}$$

$$\mathcal{I}^i(\Pi^i(\theta)) = \begin{cases} \theta_\alpha & \text{if } \alpha \in \mathcal{A}^i \\ 0 & \text{if } \alpha \notin \mathcal{A}^i \end{cases} \tag{16b}$$

We let Λ^i, Θ^i and $\mathcal{R}^i$ denote operators analogous to those in equations (11), (13) and (15) respectively, but as defined for $\mathcal{S}^i$.

Each TRP update acts on the full-dimensional vector θ, but changes only the lower-dimensional subvector $\Pi^i(\theta) = \{\theta_\alpha \mid \alpha \in \mathcal{A}^i\}$. For this reason, it is convenient to use the underbar notation to define operators of the following type:

$$\underline{\mathcal{R}}^i(\theta) = \mathcal{I}^i(\mathcal{R}^i(\Pi^i(\theta))) \tag{17a}$$

$$\underline{\Lambda}^i(\theta) = \mathcal{I}^i(\Lambda^i(\Pi^i(\theta))) \tag{17b}$$

For instance, $\underline{\Lambda}^i$ projects the exponential parameter vector θ onto spanning tree $\mathcal{S}^i$; computes the corresponding marginal vector for the distribution $p(x; \Pi^i(\theta))$ induced on the tree; and then injects back to the higher dimensional space by inserting zeroes for elements of edges not in $\mathcal{S}^i$ (i.e., for indices $\alpha \in \mathcal{A}/\mathcal{A}^i$). Moreover, analogous to $\mathcal{C}$, we define a constraint set $\mathcal{C}^i$ by imposing marginalization constraints only for edges in the spanning tree (i.e., as in equation (12) with $\mathcal{E}$ replaced by $\mathcal{E}^i$). Note that $\mathcal{C}^i \supseteq \mathcal{C}$, and since every edge is included in at least one spanning tree, we have that $\cap_i \mathcal{C}^i = \mathcal{C}$.

Using this notation, the operation of performing tree-reparameterization on spanning tree $\mathcal{S}^i$ can be written compactly as transforming a parameter vector θ into the new vector given by:

$$\mathcal{Q}^i(\theta) = \underline{\mathcal{R}}^i(\theta) + \left[I - \mathcal{I}^i \circ \Pi^i\right](\theta) \tag{18a}$$

$$= \theta + \left[\underline{\mathcal{R}}^i(\theta) - \mathcal{I}^i(\Pi^i(\theta))\right] \tag{18b}$$

where I is the identity operator. The two terms in equation (18a) parallel the decomposition of equation (5): namely, the operator $\underline{\mathcal{R}}^i$ performs reparameterization of the distribution $p^i(x)$, whereas the operator $[I - \mathcal{I}^i \circ \Pi^i]$ corresponds to leaving the residual term $q^i(x)$ unchanged. Thus, equation (18a) is a precise statement of a spanning tree update (as illustrated in Figure 4), specified in terms of the exponential parameter θ.

Given a parameter vector θ, computing $\mathcal{Q}^i(\theta)$ is straightforward, since it only involves operations on the spanning tree $\mathcal{S}^i$. The tree-based reparameterization algorithm generates a sequence of parameter vectors $\{\theta^n\}$ by successive application of these operators $\mathcal{Q}^i$. The sequence is initialized[11] at θ^0 using the original set of graph functions $\{\psi_s\}$ and $\{\psi_{st}\}$ as follows:

$$\theta^0_\alpha = \begin{cases} \log \psi_{s;j} & \text{if } \alpha = (s;j) \\ \log \psi_{st;jk} & \text{if } \alpha = (st;jk) \end{cases}$$

At each iteration n, we choose some spanning tree index $i(n)$ from $\{0, \ldots, L-1\}$, and then update using the operator on spanning tree $\mathcal{S}^i$:

$$\theta^{n+1} = \mathcal{Q}^{i(n)}(\theta^n) \tag{19}$$

In the sequel, we will also consider a relaxed iteration, involving a step size $\lambda^n \in (0, 1]$ for each iteration:

$$\theta^{n+1} = \lambda^n \mathcal{Q}^{i(n)}(\theta^n) + (1 - \lambda^n)\theta^n \tag{20}$$

[11] Other initializations are also possible. More generally, θ^0 can be chosen as any exponential parameter that induces the same distribution as the original compatibility functions $\{\psi_s\}$ and $\{\psi_{st}\}$.

where $\lambda^n = 1$ recovers the unrelaxed version.

The only restriction that we impose on the set of spanning trees is that each edge of the full graph $\mathcal{G}$ is included in at least one spanning tree (i.e., $\cup_i \mathcal{A}^i = \mathcal{A}$). It is also necessary to specify an order in which to apply the spanning trees — that is, how to choose the index $i(n)$. An obvious choice, and the one of interest for this paper, is the *cyclic ordering*, in which we set $i(n) \equiv n \pmod{L}$. More generally, any ordering — possibly random — in which each spanning tree occurs infinitely often is acceptable. A variety of possible orderings for successive projection algorithms are discussed in [11].

3.4 Belief propagation as reparameterization

In this section, we show that BP can be reformulated in a "message-free" manner as a sequence of local rather than global reparameterization operations. Specifically, in each step, new compatibility functions are determined by performing exact calculations over extremely simple (non-spanning) trees formed of two nodes and the corresponding edge joining them.

We denote by M_{st}^0 the $m-$vector corresponding to the chosen initialization of the messages.[12] The message-free version of BP iteratively updates approximations to the exact marginals $P = \{P_s, P_{st}\}$. Initial values of the approximations $T = \{T_s, T_{st}\}$ are determined from the initial messages M_{st}^0 and the original compatibility functions of the graphical model as follows:

$$T_s^0 = \eta\, \psi_s \prod_{u \in \mathcal{N}(s)} M_{us}^0 \quad \text{for all } s \in \mathcal{V} \tag{21a}$$

$$T_{st}^0 = \eta\, \psi_{st}\psi_s\psi_t \prod_{u \in \mathcal{N}(s)/t} M_{us}^0 \prod_{u \in \mathcal{N}(t)/s} M_{ut}^0 \quad \text{for all } (s,t) \in \mathcal{E} \tag{21b}$$

where η denotes a normalization factor.

At iteration n, these quantities are updated according to the following recursions:

$$T_{s;j}^n = \eta\, T_{s;j}^{n-1} \prod_{t \in \mathcal{N}(s)} \frac{1}{T_{s;j}^{n-1}} \sum_{k=1}^m T_{st;jk}^{n-1} \tag{22a}$$

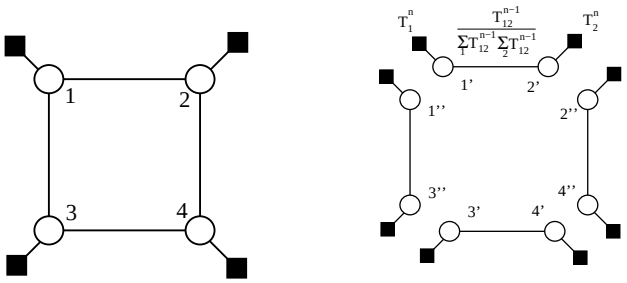
$$T_{st;jk}^n = \eta\, \frac{T_{st;jk}^{n-1}}{\big(\sum_{j=1}^m T_{st;jk}^{n-1}\big)\big(\sum_{k=1}^m T_{st;jk}^{n-1}\big)}\, T_{s;j}^n T_{t;k}^n \tag{22b}$$

The update in equation (22b) is especially noteworthy: it corresponds to performing optimal estimation on the very simple two-node tree formed by edge (s,t). As an illustration, Figure 5(b) shows the decomposition of a single-loop graph into such two-node trees. This simple reparameterization algorithm operates by performing optimal estimation on this set of non-spanning trees, one for each edge in the graph, as in equation (22b). The single-node marginals from each such tree are merged via equation (22a).

We now claim that this reparameterization algorithm is equivalent to belief propagation, summarizing the result as follows:

Proposition 3.1. The reparameterization algorithm specified by equations (21) and (22) is equivalent to the message-passing form of BP given in equations (3) and (4). In particular, for each

[12] This choice is often the vector of all ones, but any initialization with strictly positive components is permissible.



(a) (b)

Figure 5. (a) Toy example of original graph. (b) Two node trees used for updates in message-free version of belief propagation. Computations are performed exactly on each two-node tree formed by a single edge and the two associated observation potentials as in equation (22b). The node marginals from each two-node tree are merged via equation (22a).

iteration $n = 0, 1, \ldots$ and initial message vector M^0_{st}, we have the following relations:

$$M^{n+1}_{st;k} = \eta\, M^0_{st;k} \prod_{i=0}^{n} \frac{1}{T^i_{t;k}} \sum_{j=1}^{m} T^i_{st;jk} \quad \text{for all } (s,t) \in \mathcal{E} \tag{23a}$$

$$B^n_{s;j} = T^n_{s;j} \quad \text{for all } s \in \mathcal{V} \tag{23b}$$

where η denotes a normalization factor.

Proof. These equations can be established by induction on iteration n, using the BP equations (3) and (4), and the reparameterization equations (21) and (22). □

Despite the equivalence of this alternative form of BP with the message-passing version, the two schemes differ in their implementation; in particular, the reparameterization form permits in-place computation wherein new potentials overwrite the old ones. As a consequence, the reparameterization form requires only half of the storage used by the standard message-passing updates.

3.5 Empirical comparisons of BP versus TRP

Given the more global nature of a TRP iteration (in contrast with the local updates of BP), one might expect TRP to have superior convergence properties. Indeed, this has proven to be the case in various experiments that we have performed. In this section, we present results from simulations of a binary process ($m = 2$) on three different graphs: a dense 5-node graph, a 15-node single loop, and a 7×7 grid.

3.5.1 Convergence rates

At first sight, the more global nature of TRP might suggest that each TRP iteration is more complex computationally than the corresponding BP iteration. In fact, the opposite statement is true. Each TRP update involves $\mathcal{O}(m^2(n-1))$ operations, whereas each iteration of BP requires $\mathcal{O}(m^2 e)$ operations, where $e \geq n-1$ is the number of edges in the graph. Consequently, each TRP iteration is slightly cheaper than a BP iteration for tree-like graphs (e.g., a single loop); and considerably

cheaper for denser graphs (such as the grid used in this section) where $e \gg n$. Therefore, in order to make comparisons fair in terms of actual computation required, whenever we report iteration numbers, they are rescaled in terms of relative cost (i.e., for each graph, TRP iterations are rescaled by the ratio $(n-1)/e < 1$).

For each graph, we performed simulations under three conditions: edge potentials that are *repulsive* (i.e., that encourage neighboring nodes to take opposite values); *attractive* (that encourage neighbors to take the same value); and *mixed* (in which some potentials are attractive, while others are repulsive). For each of these experimental conditions, each run involved a random selection of the initial parameter vector θ^0 defining the distribution $p(x;\theta^0)$. In all experiments reported here, we generated the single node parameters $\theta_{s;j}$ as follows:[13] for each node $s \in \mathcal{V}$, sample $a_s \sim \mathcal{N}(0,(0.25)^2)$, and set $[\theta_{s;0} \quad \theta_{s;1}] = [a_s \quad -a_s]$. To generate the edge potential components $\theta_{st;jk}$, we began by sampling $b_{st} \sim \mathcal{N}(0,1)$ for each edge (s,t), With δ_{jk} denoting the Kronecker delta for j,k, we set the edge potential components in one of three ways depending on the experimental condition:

(a) repulsive condition: $\theta_{st;jk} = -(2\delta_{jk}-1)\,|b_{st}|$.

(b) attractive condition: $\theta_{st;jk} = (2\delta_{jk}-1)\,|b_{st}|$.

(c) mixed condition: $\theta_{st;jk} = (2\delta_{jk}-1)\,b_{st}$.

For each graph and experimental condition, we ran a total of 500 trials, comparing the performance of TRP to BP. On any given run, an algorithm was deemed to converge when the mean L^2 difference between successive node elements $(\frac{1}{n}\sum_s \|\theta_s^{n+1}-\theta_s^n\|^2)$ reached a threshold[14] of $\epsilon = 1\times 10^{-16}$. A run

Graph	Single 15-loop						7×7 grid					
	R		A		M		R		A		M	
BP	500	23.2	500	23.4	500	23.6	455	62.3	457	65.8	267	310.1
TRP	500	8.1	500	8.0	500	8.2	500	30.5	500	30.8	282	103.2

Table 1. Comparison of convergence behavior of TRP versus BP for a single loop of 15 nodes; and a 7×7 grid. Potentials were chosen randomly in each of the three conditions: repulsive potentials (R); attractive potentials (A); mixed potentials (M). First and second numbers in each box denote the number of convergent runs out of 500; and the mean number of iterations (rescaled by relative cost and computed using only runs where both TRP and BP converged) respectively.

in which which a given algorithm failed to reach this threshold within 3000 iterations was classified as a failure to converge. In each condition, we report the total number of convergent trials (out of 500); and also the mean number of iterations required to converge, rescaled by the ratio $(n-1)/e$ and based only on trials where both TRP and BP converged.

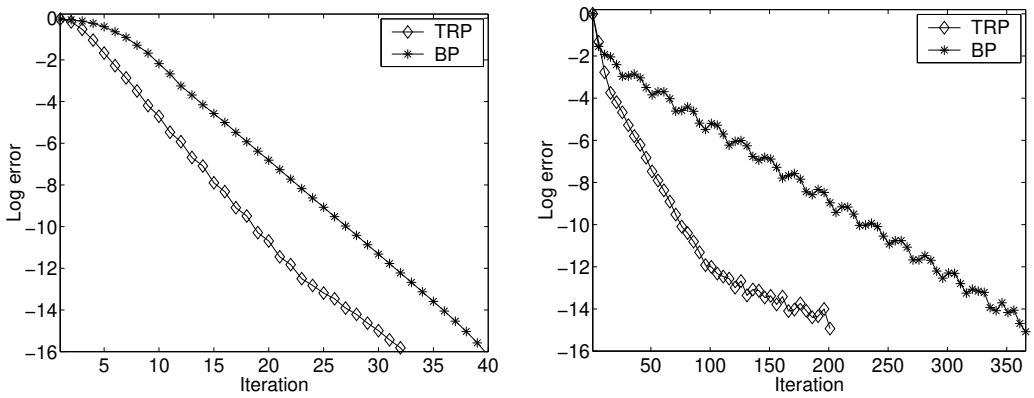
Table 1 shows some summary statistics for the two graphs used in these experiments. For the single loop, we implemented TRP with two spanning trees, whereas we used four spanning trees for the grid. Although both algorithms converged on all trials for the single loop, the rate of TRP convergence was significantly (roughly 3 times) faster. The superiority of TRP is easily

[13] The notation $\mathcal{N}(0,\sigma^2)$ denotes a zero-mean Gaussian with variance σ^2.

[14] This value was chosen by examining the behavior of each algorithm, and the successive differences in iterate values over a number of runs.

understandable in the single loop case. A single iteration of TRP suffices to transmit information from each node to every other node in the graph, whereas the local updates of BP will require as many iterations as the graph diameter ($\lfloor n/2 \rfloor$ for a loop of n nodes).

For the grid, algorithm behavior depends more on the experimental condition. The repulsive and attractive conditions are relatively easy, though still difficult enough for BP that it failed to converge on roughly 10% of the trials, in contrast to the perfect convergence percentage of TRP. In terms of mean convergence rates, TRP converged more than twice as quickly as BP. The mixed condition is difficult for suitably strong edge potentials on a grid: in this case both algorithms failed to converge on almost half the trials, although TRP converged more frequently than BP. Moreover, on runs where both algorithms converged, the TRP mean rate of convergence was three times faster than BP.



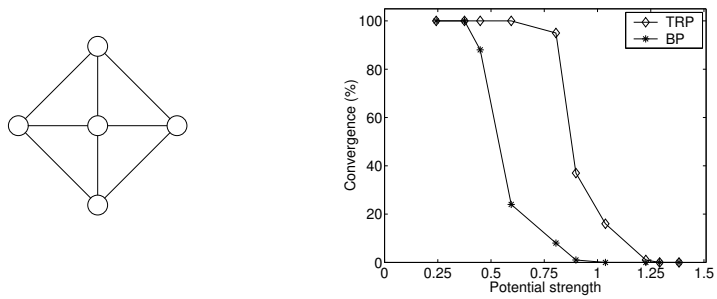
(a) Attractive potentials (b) Mixed potentials

Figure 6. Convergence plots of log error ($\log \sum_{s,j} \|\theta^n_{s;j} - \theta^*{}_{s;j}\|^2$) versus iteration number n for the 7×7 grid under two conditions. Here both BP and TRP converge to the same fixed point θ^*.

Figure 6 compares the convergence behavior of BP and TRP. Plotted is the log error between the single-node elements of θ^n and θ^* at each iteration, where θ^* is a fixed point common to BP and TRP, versus the iteration number. Panel (a) illustrates the attractive condition, in which the error in TRP begins decreasing earlier than BP, with the asymptotic rate being similar. Examples in the repulsive condition show similar convergence behavior. On the other hand, in the mixed condition shown in (b), the global updates of the tree-reparameterization updates take effect after a few iterations, leading to much swifter convergence for the first 80 iterations or so. For later iterations, the convergence of TRP slows down, which may be due to numerical precision issues or other effects related to the choice of trees in TRP iterations. For example, each TRP update ignores some local interactions corresponding to the edges removed to form the spanning tree. These edges are covered by other spanning trees in the set used; however, it remains an open question how to choose trees so as to maximize the rate of convergence. In this context, one could imagine a hybrid algorithm (in which BP iterations are interspersed with TRP iterations). If one considers the example of Figure 6(b), a switch from TRP to BP after iteration 80 would lead to faster convergence than either algorithm alone. The exploration of such issues remains for future research.

3.5.2 Domain of convergence

The dense 5-node graph shown in Figure 7(a) serves to illustrate how TRP updates tend to converge for a wider range of potentials than BP. We simulated a binary process over a range of potential strengths μ ranging from -0.3 to -1.0. Explicitly, for each value of μ, we made a deterministic assignment of the potential for each edge (s,t) of the graph as $\theta_{st;jk} = (2\delta_{jk} - 1)\mu$. For each potential strength we conducted 100 trials, where on each trial the single-node potentials were set randomly by sampling $a_s \sim \mathcal{N}(0, (0.25)^2)$ and setting $[\theta_{s;0} \quad \theta_{s;1}] = [a_s \quad -a_s]$. On any given trial, the convergence of a given algorithm was assessed as in Section 3.5.1. Plotted in



(a) (b)

Figure 7. (a) Simple 5-node graph. (b) Comparison of BP and TRP convergence percentages versus function of potential strength on graph in (a). Plotted along the abscissa as a measure of potential strength is the multi-information $D(p(x_1, \ldots, x_n) \,\|\, \prod_{s=1}^{n} p(x_s))$. Both TRP and BP exhibit a threshold phenomenon, with TRP converging for a wider range of potentials.

Figure 7(b) is the percentage of successfully converged trials versus potential strength for TRP and BP. Both algorithms exhibit a type of threshold behavior, in which they converge with 100% success up to a certain potential strength, after which their performance degrades rapidly. However, TRP updates extend the effective range of convergence, even on such a small graph. Moreover, there is a range of potential settings for which BP almost never converges, whereas TRP converges for almost every trial.[15]

4 Analysis of fixed points and convergence

In this section, we present a number of results related to the fixed points and convergence of TRP and BP. In Section 4.1, we define and develop properties of a cost function G that arises as an approximation to the Kullback-Leibler divergence. In Section 4.2, we show that fixed points of the TRP algorithm satisfy necessary conditions to be a constrained minimum of this cost function. At the heart of our analysis is a geometric characterization of successive iterates; it is of independent interest because it establishes links to successive projection techniques for constrained minimization of Bregman distances [e.g., 11]. By combining our results with those of Yedidia et al. [52], we are able conclude that fixed points of the TRP algorithm coincide with those of BP. Moreover, our analysis allows us to formulate sufficient conditions for convergence of the TRP algorithm in the case of

[15] This result is not dependent on the symmetry of the problem induced by our choice of edge potentials; for instance, the results are similar if edge potentials are perturbed from their nominal strengths by small random quantities.

two spanning trees, which we present in Section 4.3. In Section 4.4, we formalize the fundamental property of reparameterization updates — namely, that they leave unchanged the distribution on the full graph. Some of the consequences of this invariance can be found in an investigation of the geometry of TRP in Section 4.4, and in the elementary proof in Section 4.5 of the result originally developed in [43, 49] concerning the behavior of BP for jointly Gaussian distributions.

4.1 Approximation to the Kullback-Leibler divergence

The cost function G central to our analysis arises as an approximation to the Kullback-Leibler (KL) divergence [14], one which is exact for a tree. Let $T \in (0,1)^{d(\theta)}$ be a pseudomarginal vector, and let θ be a parameter vector for the original graph $\mathcal{G}$ with cycles. We then define

$$G(T;\theta) \triangleq \sum_{\alpha \in \mathcal{A}} T_\alpha \big[\Theta(T) - \theta\big]_\alpha \tag{24}$$

To see how this cost function is related to the KL divergence as defined in equation (8), consider the analogous function defined on spanning tree $\mathcal{S}^i$ for a vector $T \in \mathcal{C}^i$:

$$G^i(\Pi^i(T);\Pi^i(\theta)) = \sum_{\alpha \in \mathcal{A}^i} T_\alpha \big[\Theta^i(\Pi^i(T)) - \theta\big]_\alpha \tag{25}$$

where $\Pi^i(\theta)$ and $\Pi^i(T)$ are exponential parameter vectors and marginal vectors, respectively, defined on $\mathcal{S}^i$. With the exponential parameterization of equation (9) applied to any tree, we have $T_\alpha = \mathbb{E}_{\Theta^i(\Pi^i(T))}[\phi_\alpha]$ for all indices $\alpha \in \mathcal{A}^i$. As a result, the function G^i is related to the KL divergence as follows:

$$D(\Theta^i(\Pi^i(T)) \parallel \Pi^i(\theta)) = G^i(\Pi^i(T);\Pi^i(\theta)) + \Phi(\Pi^i(\theta)) \tag{26}$$

In establishing this equivalence, we have used the fact that the partition function of the factorization in equation (2) is unity, so that the corresponding log partition function is zero (i.e., $\Phi(\Theta^i(\Pi^i(T))) = 0$). Therefore, aside from an additive constant $\Phi(\Pi^i(\theta))$ independent of T, the quantity $G^i(\Pi^i(T);\Pi^i(\theta))$ viewed as a function of $\Pi^i(T)$, is equivalent to the KL divergence.

Now consider the problem of minimizing the KL divergence as a function of T, subject to the constraint $T \in \mathcal{C}^i$. The KL divergence in equation (26) assumes its minimum value of zero at the vector of correct marginals on the spanning tree — namely, $\widehat{P} = \underline{\Lambda}^i(\Pi^i(\theta)) \in \mathcal{C}^i$. By the equivalence shown in equation (26), minimizing the function $G^i(\Pi^i(T);\Pi^i(\theta))$ over $T \in \mathcal{C}^i$ will also yield the same minimizing argument $\widehat{P}$.

For the original graph $\mathcal{G}$ with cycles, the cost function G of equation (24) is not equivalent to the KL divergence. The argument leading up to equation (26) cannot be applied because $\Lambda(\Theta(T)) \neq T$ for a general graph with cycles. Nevertheless, this cost function lies at the core of our analysis of TRP. Indeed, we show in Section 4.2 how the TRP algorithm can be viewed as a successive projection technique for constrained minimization of the cost function G, in which the reparameterization update on spanning tree $\mathcal{S}^i$ as in equation (19) corresponds to a projection onto constraint set $\mathcal{C}^i$. Moreover, we will see that G and the Bethe free energy, though different for points outside the set $\mathcal{C}$ defined in equation (12), agree on this constraint set. This allows us to use recent results of Yedidia et al. [52] to link the fixed points of TRP with those of BP.

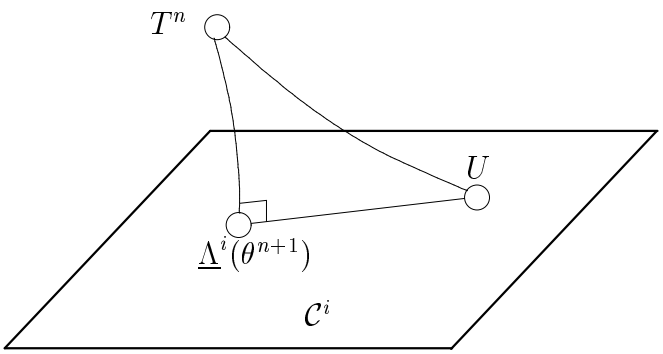


Figure 8. Illustration of the geometry of Proposition 4.1. The pseudomarginal vector T^n is projected onto the linear constraint set $\mathcal{C}^i$. This yields the point $\underline{\Lambda}^i(\theta^{n+1})$ that minimizes the cost function G^i over the constraint set $\mathcal{C}^i$.

4.2 Tree reparameterization as a successive projection technique

The first result of this section is of a geometric nature: it shows how successive iterates θ^n and θ^{n+1} are related via the cost function G.

Proposition 4.1. Assume that the sequence $\{\theta^n\}$ generated by equation (20) with step sizes λ^n remains bounded. Let $i = i(n)$ be the tree index used at iteration n. Then for all $U \in \mathcal{C}^i$:

$$G(U;\theta^n) = G(U;\theta^{n+1}) + \lambda^n G(\underline{\Lambda}^i(\mathcal{Q}^i(\theta^n));\theta^n) \tag{27}$$

where $\underline{\Lambda}^i$ is defined in equation (17b).

Proof. See Appendix A. □

An important special case of Proposition 4.1 is the unrelaxed update ($\lambda^n = 1$), in which case equation (27) simplifies to

$$G(U;\theta^n) = G(U;\theta^{n+1}) + G(\underline{\Lambda}^i(\theta^{n+1});\theta^n) \tag{28}$$

Figure 8 illustrates the geometry of Proposition 4.1 in this unrelaxed setting, for which we are guaranteed the existence[16] of a pseudomarginal T^n such that $\theta^n = \Theta(T^n)$. We project the point T^n onto the constraint set $\mathcal{C}^i$, where the function G^i serves as the distance measure. This projection yields the point $\underline{\Lambda}^i(\theta^{n+1}) \in \mathcal{C}^i$, and we have depicted its relation to an arbitrary U also in $\mathcal{C}^i$.

We note that a result analogous to equation (28) holds for the minimum of a Bregman distance over a linear constraint set [e.g., 11]. Well-known examples of Bregman distances include the L^2 norm, and the KL divergence. Choosing the KL divergence as the Bregman distance leads to the I-projection in information geometry [e.g., 4, 12, 15]. Even when the distance is not the L^2 norm, results of the form in equation (28) are still called Pythagorean. Indeed, the function G plays the role of the squared Euclidean distance, with the three points T^n, $\underline{\Lambda}^i(\theta^{n+1})$ and U analogous to the

[16] The image of the unit hypercube $(0,1)^{d(\theta)}$ under the map Θ is not all of $\mathbb{R}^{d(\theta)}$, since, for example, given any pseudomarginal $T \in (0,1)^{d(\theta)}$, we have $[\Theta(T)]_{s;j} = \log T_{s;j} < 0$. Nonetheless, for unrelaxed updates producing iterates θ^n, it can be seen that the inverse image of a point θ^n under Θ will be non-empty as soon as each edge has been updated at least once.

vertices of a right triangle, as illustrated in Figure 8. In addition, a wide class of algorithms can be formulated as successive projection techniques for minimizing a Bregman distance over a set formed by an intersection of linear constraints (e.g., generalized iterative scaling [16]). The Pythagorean relation is instrumental in establishing the convergence of such techniques [15, 11].

The problem at hand is similar, since we are interested in minimizing a function over a constraint set formed as an intersection of linear constraint sets (i.e., $\mathcal{C} = \cap_i \mathcal{C}^i$). However, the function G is certainly not a Bregman distance since, for instance, it can assume negative values. Nonetheless, Proposition 4.1 allows us to show that any fixed point θ^* of the TRP algorithm satisfies the necessary conditions for it to be a local minimum of $G(T;\theta^0)$ over the constraint set $\mathcal{C}$. Although the result extends to other orderings, for concreteness we state it here for a *cyclic ordering* of spanning trees $\mathcal{S}^0, \ldots, \mathcal{S}^{L-1}$: i.e., the tree index for iteration n is chosen as $i(n) = n(\mathrm{mod}\, L)$.

Theorem 4.1. Consider a sequence of iterates $\{\theta^n\}$ generated by equation (20) with a cyclic tree ordering, and using step sizes $\lambda^n \in [\epsilon, 1]$ for some $\epsilon > 0$. Suppose that the sequence $\{\theta^n\}$ remains bounded and converges to some θ^*. Then

(a) The point θ^* is a fixed point of all the tree operators $\mathcal{Q}^i$. I.e., $\theta^* = \mathcal{Q}^i(\theta^*)$ for all indices $i = 0, \ldots, L-1$. Therefore, each fixed point θ^* is associated with a unique psuedomarginal vector $T^* \in \mathcal{C}$.

(b) The pseudomarginal vector T^* satisfies the necessary conditions for it to be a local minimum of $G(T;\theta^0)$ over the constraint set $\mathcal{C}$:

$$\sum_\alpha \frac{\partial G}{\partial T_\alpha}(T^*;\theta^0)\, \left[U - T^*\right]_\alpha \;\; = \;\; 0$$

for all U in the constraint set $\mathcal{C}$.

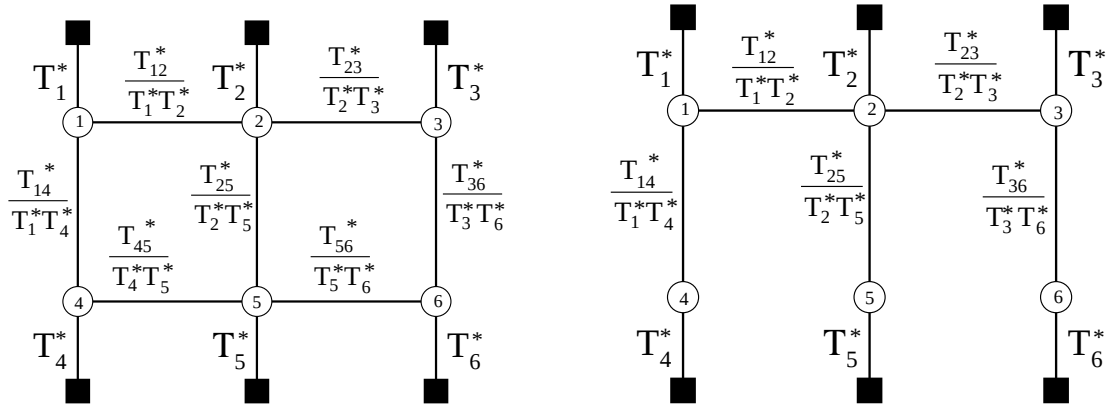
(c) The TRP algorithm always possesses at least one fixed point.

(d) Fixed points of the TRP algorithm coincide with those of BP.

Proof. See Appendix A. □

A few remarks about Theorem 4.1 are in order. First of all, to clarify the result stated in (a), the unique pseudomarginal vector T^* associated with θ^* can be constructed explicitly as follows. For an arbitrary index α, pick a spanning tree $\mathcal{S}^i$ such that $\alpha \in \mathcal{A}^i$. Then define $T^*_\alpha = [\Lambda^i(\Pi^i(\theta^*))]_\alpha$; that is, T^*_α is the value of this (single node or pairwise) marginal for the tree distribution on $\mathcal{S}^i$ specified by $\Pi^i(\theta^*)$. Note that this is a consistent definition of T^*_α, because the condition of part (a) means that $[\Lambda^i(\Pi^i(\theta^*))]_\alpha$ is the same for all spanning tree indices $i \in \{0, \ldots, L-1\}$ such that $\alpha \in \mathcal{A}^i$. Moreover, this construction ensures that $T^* \in \mathcal{C}$, since it must satisfy the normalization and marginalization constraints associated with every node and edge.

Figure 9 illustrates this characterization of fixed points in terms of T^*. Shown in panel (a) is an example of a graph with cycles, parameterized according to the approximate marginals T^*_{st} and T^*_s. The consistency condition implies that if edges are removed from the full graph to form a spanning tree, as shown in panel (b), then the quantities T^*_{st} and T^*_s correspond to *exact* marginal distributions over the tree. This statement holds for *any* singly-connected substructure embedded within the full graph with cycles — not just the spanning trees used to implement the algorithm.



(a) Fixed point on full graph (b) Tree consistency condition.

Figure 9. Illustration of fixed point consistency condition. (a) Fixed point $\{T_s^*,\ T_{st}^*\}$ on the full graph with cycles. (b) Illustration of consistency condition on an embedded tree. The quantities $\{T_s^*,\ T_{st}^*\}$ must be *exact* marginal probabilities for any tree embedded within the full graph.

Thus, Theorem 4.1 provides an alternative and very intuitive view of BP or TRP: such algorithms attempt to reparameterize a distribution on a loopy graph so that it is consistent with respect to each embedded tree. In this regard, part (c) of Theorem 4.1 is noteworthy, in that it guarantees that any positive distribution on a graph can be reparameterized in a form that satisfies the tree-based consistency condition of part (a). It is interesting that such a result, though obvious for any tree, should also hold for a positive distribution on an arbitrary loopy graph.[17]

Part (d) of Theorem 4.1 follows because our cost function G and the Bethe free energy coincide on the constraint set $\mathcal{C}$. It is also possible to establish this equivalence in a more constructive manner. In particular, any fixed point $\{M_{st}^*\}$ of the message updates in equation (3) can be transformed into a pseudomarginal vector $T^* = \{T_s^*,\ T_{st}^*\}$ as follows:

$$T_s^* = \eta\, \psi_s \prod_{u \in \mathcal{N}(s)} M_{us}^* \tag{29a}$$

$$T_{st}^* = \eta\, \psi_{st} \psi_s \psi_t \prod_{u \in \mathcal{N}(s)/t} M_{us}^* \prod_{u \in \mathcal{N}(t)/s} M_{ut}^* \tag{29b}$$

The message fixed point condition of equation (3) guarantees that the corresponding T^* belongs to the constraint set $\mathcal{C}$. Membership in $\mathcal{C}$ guarantees that T^* is locally consistent with respect to all the simple two-node trees formed by single edges $(s,t) \in \mathcal{E}$, as illustrated in Figure 5. This local consistency on two-node trees then implies that the vector T^* must be consistent on any singly-connected embedded substructure (using the equivalence of local and global consistency for trees). That is, T^* satisfies the tree-based consistency condition of part (a), as illustrated in Figure 9. Even more generally, a similar line of reasoning establishes that any constrained local minimum of the Bethe free energy, whether obtained by TRP/BP or an alternative minimization

[17] It is also interesting to note the link between this fixed point characterization, and Freeman and Weiss' [19] analysis of the max-product algorithm. This is a technique for computing an approximate MAP assignment, in contrast with the approximate marginal distributions computed by BP/TRP. In particular, they show that the approximate MAP assignment of max-product is guaranteed to be correct on local neighborhoods consisting of a tree plus a single cycle.

technique [e.g., 53, 50, 18], can be identified with a pseudomarginal vector T^* satisfying the condition of Theorem 4.1.

4.3 Sufficient conditions for convergence for two spanning trees

Proposition 4.1 can also be used to derive a set of conditions that are sufficient to guarantee the convergence in the case of two spanning trees. To convey the intuition of the proof, suppose that it were possible to interpret the cost function G as a distance function. Moreover, suppose U were an arbitrary element of $\mathcal{C} = \cap_i \mathcal{C}^i$, so that we could apply Proposition 4.1 for each index i. Then equation (27) would show that the "distance" between θ^n and an arbitrary element $U \in \mathcal{C}$, as measured by G, decreases at each iteration. As with proofs on the convergence of successive projection techniques for Bregman distances [e.g., 11, 15], this property would allow us to establish convergence of the algorithm.

Of course, there are two problems with the use of G as a type of distance: it is not necessarily non-negative, and it is possible that $G(\Lambda^i(\mathcal{Q}^i(\theta)); \theta) = 0$ for some $\theta \neq \mathcal{Q}^i(\theta)$. With respect to the first issue, we are able to show (Appendix C) in general that an appropriate choice of step size will ensure the non-negativity of $G(\Lambda^i(\mathcal{Q}^i(\theta)); \theta)$. The following result then states sufficient conditions (including assuming that the second problem does not arise along TRP trajectories) for convergence in the case of two spanning trees:

Theorem 4.2. Consider the application of TRP with two spanning trees $\mathcal{S}^0$ and $\mathcal{S}^1$. Suppose that the sequence of iterates $\{\theta^n\}$ remains bounded, and that:
(a) for $i = 0,\ 1$, the condition $G(\underline{\Lambda}^i(\mathcal{Q}^i(\theta^n)); \theta^n) \to 0$ implies that $[\mathcal{Q}^i(\theta^n) - \theta^n] \to 0$.
(b) there exists some integer L such that the condition $G(\underline{\Lambda}^0(\mathcal{Q}^1(\theta^n)); \theta^n) G(\underline{\Lambda}^1(\mathcal{Q}^0(\theta^n)); \theta^n) > 0$ holds for all $n \geq L$.
Then there exist choices of the step sizes λ^n such that the sequence θ^n converges to some θ^* in the desired constraint set. I.e., $\theta^* = \mathcal{Q}^i(\theta^*)$ for $i = 0, 1$.

Proof. See Appendix C, which includes a characterization of the step size choices that ensure convergence. □

This result, though its hypotheses cannot be checked a priori, provides some insight into the factors that cause failures of convergence when applying TRP/BP. In particular, the proof of Theorem 4.2 shows that assumption (a) is analogous to the *gradient-relatedness* condition of standard descent algorithms for nonlinear optimization [8].

4.4 Geometry and invariance of TRP updates

In this section, we establish a fundamental property of TRP updates— namely, that they leave invariant the full distribution on the graph with cycles. We then exploit this invariance to develop the geometry of TRP updates.

Recall that a crucial feature of the exponential $\theta-$parameterization of equation (9) is its overcompleteness. For this reason, given a fixed exponential parameter $\widetilde{\theta}$, it is interesting to consider the following subset of $\mathbb{R}^{d(\theta)}$:

$$\mathcal{M}(\widetilde{\theta}) \triangleq \left\{ \theta \in \mathbb{R}^{d(\theta)} \ \mid \ p(x;\theta) \equiv p(x;\widetilde{\theta}) \right\} \tag{30}$$

where $d(\theta)$ denotes the length of θ as defined in equation (10). This set can be seen to be a closed submanifold of $\mathbb{R}^{d(\theta)}$— in particular, note that it is the inverse image of the point $\widetilde{\theta}$ under the continuous mapping $\theta \mapsto p(x;\theta)$. To further understand the structure of $\mathcal{M}(\widetilde{\theta})$, we need to link the overcomplete θ-parameterization to a minimal parameterization, specified by a linearly independent collection of functions.

To illustrate the basic intuition, we begin with the special case of binary-valued nodes ($m = 2$). In this case, the indicator functions of the θ-representation can be expressed as linear combinations of the functions x_s and $x_s x_t$. For example, we have $\delta(x_s = 0)\delta(x_t = 1) = (1 - x_s)\, x_t$. Thus, in the binary case, a minimal parameterization is given by

$$p(x;\gamma) = \exp\Big\{ \sum_s \gamma_s x_s + \sum_{s,t\in\mathcal{E}} \gamma_{st} x_s x_t - \Phi(\gamma)\Big\} \tag{31}$$

Such a model is known as a Boltzmann machine in the inference literature [e.g., 27].

These ideas are extended readily to discrete processes with $m > 2$ states. In general, for a graph with pairwise cliques, it can be seen that the following collection of functions constitute a minimal representation:

$$\mathcal{R}(s) \triangleq \{x_s^a \mid a = 1, \ldots, m-1\} \qquad \text{for } s \in \mathcal{V} \tag{32a}$$

$$\mathcal{R}(s,t) = \{x_s^a x_t^b \mid a, b = 1, \ldots, m-1\} \qquad \text{for } (s,t) \in \mathcal{E} \tag{32b}$$

As in the binary case illustrated above, we let γ be a parameter vector of weights on these functions.

In contrast to the overcomplete case, the minimal representation induces a one-to-one correspondence between parameter vectors γ and distributions $p(x;\gamma)$. Therefore, associated with the distribution $p(x;\widetilde{\theta})$ is a unique vector $\widetilde{\gamma}$ such that $p(x;\widetilde{\theta}) \equiv p(x;\widetilde{\gamma})$. The dimension of the exponential family [see 3] is given by the length of γ, which we denote by $d(\gamma)$. From equation (32), we see that this dimension is $d(\gamma) = [(m-1)\, n + (m-1)^2\, e]$, where e is the number of edges in the graph. On the basis of these equivalent representations, the set $\mathcal{M}(\widetilde{\theta})$ can be characterized as follows:

Proposition 4.2. The set $\mathcal{M}(\widetilde{\theta})$ of equation (30) is a linear submanifold of $\mathbb{R}^{d(\theta)}$ of codimension $d(\gamma)$. It has the form $\{\theta \in \mathbb{R}^{d(\theta)} \mid A\,\theta = \widetilde{\gamma}\}$, where A is an appropriately defined $d(\gamma) \times d(\theta)$ matrix of constraints.

Proof. See Appendix D. □

Based on this proposition, we now prove a fundamental property of TRP updates:

Theorem 4.3. Consider a sequence of TRP iterates $\{\theta^n\}$ generated by the relaxed updates:

$$\theta^{n+1} \;=\; \lambda^n \mathcal{Q}^{i(n)}(\theta^n) + (1 - \lambda^n)\theta^n \tag{33}$$

Then the distribution on the full graph with cycles is invariant under the updates: that is, $\theta^n \in \mathcal{M}(\theta^0) = \{\theta \in \mathbb{R}^{d(\theta)} \mid p(x;\theta) \equiv p(x;\theta^0)\}$ for all $n = 1, 2, \ldots$. Moreover, any limit point θ^* of the sequence also belongs to $\mathcal{M}(\theta^0)$.
In addition, the same statements hold for the reparameterization form of BP presented in Section 3.4.

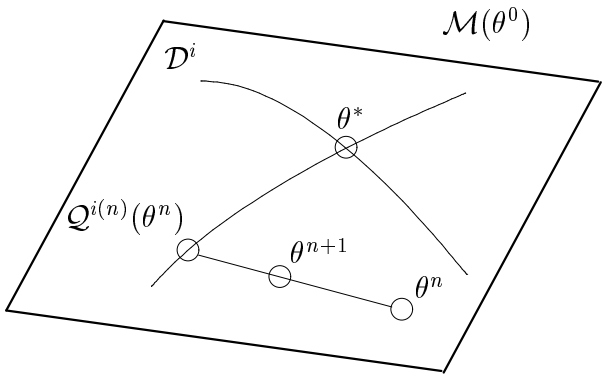


Figure 10. Geometry of tree-reparameterization updates in the exponential domain. Iterates are confined to the linear manifold $\mathcal{M}(\theta^0)$. Curved lines within $\mathcal{M}(\theta^0)$ correspond to the intersection $\mathcal{D}^i \cap \mathcal{M}(\theta^0)$, for a particular spanning tree constraint set $\mathcal{D}^i$. Each update entails moving along the line between θ^n and the point $\mathcal{Q}^{i(n)}(\theta^n)$ on $\mathcal{D}^{i(n)}$. Any fixed point θ^* belongs to $\mathcal{D} = \cap_i \mathcal{D}^i$.

Proof. As previously described, the unrelaxed TRP update of equation (19) does indeed leave the distribution unchanged, so that $\mathcal{Q}^i(\theta) \in \mathcal{M}(\theta)$ for all θ. The relaxed update of equation (33) is nothing more than a convex combination of two exponential vectors (θ^n and $\mathcal{Q}^{i(n)}(\theta^n)$) that parameterize the same distribution, so that by recourse to Proposition 4.2, the proof of the first statement is complete.As noted earlier, $\mathcal{M}(\theta^0)$ is a closed submanifold, so that any limit point of the sequence $\{\theta^n\}$ must also belong to $\mathcal{M}(\theta^0)$. An inductive argument establishes that the reparameterization form of BP also leaves invariant the distribution on the full graph.

□

These results together also lead to a geometric understanding of the TRP updates in the exponential domain (i.e., in terms of the parameter vector θ). In order to describe this geometry, we begin by defining an exponential analog of the constraint set $\mathcal{C}$ as follows:

$$\mathcal{D} \;=\; \{\theta \mid \theta = \Theta(T) \text{ for some } T \in \mathcal{C}\} \;=\; \Theta(\mathcal{C}) \tag{34}$$

If a vector θ belongs to the set $\mathcal{D}$, then it must satisfy certain nonlinear convex constraints (e.g., $\log[\sum_j \exp(\theta_{s;j})] = 0$ for all $s \in \mathcal{V}$; and $\log[\sum_j \exp(\theta_{st;jk} + \theta_{s;j})] = 0$ for all $(s,t) \in \mathcal{E}$). For each spanning tree constraint set $\mathcal{C}^i$, we also define the set $\mathcal{D}^i$ in an analogous manner, and note that by construction $\mathcal{D} = \cap_i \mathcal{D}^i$.

Figure 10 illustrates the geometry of the TRP updates in the exponential domain. First of all, in agreement with Theorem 4.3, all iterates θ^n are confined to the linear manifold $\mathcal{M}(\theta^0)$. The intersection $\mathcal{D}^i \cap \mathcal{M}(\theta^0)$ will trace out some curved set within the linear manifold. Each step of TRP entails moving along the line between the current iterate θ^n and the projection $\mathcal{Q}^{i(n)}(\theta^n)$. This latter point belongs to the spanning tree constraint set $\mathcal{D}^{i(n)}$. When the sequence θ^n converges to some θ^*, then we are guaranteed that $\theta^* \in \mathcal{D} \cap \mathcal{M}(\theta^0)$.

4.5 Implications for continuous processes

The reparameterization framework can be extended and has important implications for continuous processes as well. In particular, by extension to the Gaussian case, we obtain an elementary proof of

a generalization to TRP of the result [43, 49] that when BP converges, the means are, in fact, correct. To establish this result, let us consider the Gaussian analog of TRP. For simplicity in notation, we treat the case of scalar Gaussian random variables at each node (though the ideas extend easily to the vector case). In the scalar Gaussian case, the approximate marginal distribution $T_s(x_s)$ at each node $s \in \mathcal{V}$ is parameterized by a mean μ_s and variance σ_s^2. Similarly, the approximate joint distribution $T_{st}(x_s, x_t)$ can be parameterized by a mean vector $\nu_{st} \triangleq [\nu_{st;s} \quad \nu_{st;t}]'$, and a covariance matrix. At each iteration, the edge (s,t) is labeled with the edge function $T_{st}/\widetilde{T}_{st;s}\widetilde{T}_{st;t}$, where $\widetilde{T}_{st;s}(x_s) = \int_{-\infty}^{\infty} T_{st}(x_s, x_t)dx_t$ is the marginal distribution over x_s induced by T_{st}. This edge function is parameterized by the mean vector ν_{st}, and a quadratic form $Q_{st} = [q_{st;s} \;\; q_{st} \; ; \; q_{st} \;\; q_{st;t}]$. With this set-up, we have:

Proposition 4.3. Consider the Gaussian analog of TRP or BP, and suppose that it converges. Then the computed means are exact, whereas in general the error covariances are incorrect.

Proof. From the original problem specification, we have

$$-\log p(x) \quad = \quad 1/2 \; (x - \widehat{\mu})^T \; P^{-1} \; (x - \widehat{\mu}) + C \tag{35}$$

where P^{-1} is the inverse covariance; C is a constant independent of x; and $\widehat{\mu}$ are the correct means on the loopy graph.

We begin by noting that the Gaussian analog of Theorem 4.3 guarantees that this distribution will remain invariant under the reparameterization updates of TRP (or BP). At any iteration, the distribution is reparameterized in terms of T_s and the edge functions as follows:

$$\begin{aligned} -\log p(x) \quad = \quad & \frac{1}{2} \sum_{(s,t) \in \mathcal{E}} \left\{ q_{st;s}(x_s - \nu_{st;s})^2 + 2q_{st}(x_s - \nu_{st;s})(x_t - \nu_{st;t}) + q_{st;t}(x_t - \nu_{st;t})^2 \right\} \\ & + \frac{1}{2} \sum_s (x_s - \mu_s)^2 / \sigma_s^2 + C \end{aligned} \tag{36}$$

Note that the pseudomarginal vector $\{T_s, T_{st}\}$ need not be consistent so that, for example, $\widetilde{T}_{st;s}(x_s)$ need not equal $T_s(x_s)$. However, suppose that TRP (or BP) converges so that these quantities are equal, which, in particular, implies that $\mu_s = \nu_{st;s}$ for all (s,t) such that $t \in \mathcal{N}(s)$. That is, the means parameterizing the edge functions must agree with the means at the node marginals. In this case, equations (35) and (36) are two alternative representations of the same quadratic form, so that we must have $\widehat{\mu}_s = \mu_s$ for each node $s \in \mathcal{V}$. Therefore, the means computed by TRP or BP must be exact. In contrast to the means, there is no reason to expect that the error covariances in a graph with cycles need be exact. □

It is worth remarking that there exist highly efficient techniques from numerical linear algebra (e.g., conjugate gradient [17]) for computing the means of a linear-Gaussian problem on a graph. Therefore, although TRP and BP compute the correct means (if they converge), there is little reason to apply them in practice. There remains, however, the interesting problem of computing correct error covariances at each node: we refer the reader to [47] for description of an embedded spanning tree method that efficiently computes both means and error covariances for a linear-Gaussian problem on a graph with cycles.

5 Analysis of the approximation error

An important but very difficult problem is analysis of the errors that arise from approximate algorithms such as BP and TRP. In fact, to date very few results are available on characterizing the error. Empirical simulations show that BP gives good approximations for certain graphs (e.g., those with long cycles relative to potential strength); in other cases, however, the approximations can be very poor. In the absence of error analysis, it is impossible to gauge the validity of the approximation. In this section, we provide a contribution to this important problem.

Our analysis of the approximation error is based on two fundamental properties of a fixed point θ^*. First of all, part (a) of Theorem 4.1 dictates that for an arbitrary spanning tree $\mathcal{S}^i$, the single node elements ${\theta^*}_{s;j} = \log T^*_{s;j}$ correspond to exact marginal distributions on the spanning tree. That is, the quantities $T^*_{s;j}$ have two distinct interpretations:

(a) as the BP or TRP approximations to the exact marginals on the loopy graph.

(b) as the exact single node marginals of a distribution defined by the spanning tree $\mathcal{S}^i$.

Secondly, by the invariance stated in Theorem 4.3, the distribution $p(x;\theta^*)$ induced by the fixed point θ^* is equivalent to the original distribution $p(x;\theta^0)$. In conjunction, these two properties imply that the exact marginals on the full loopy graph are related to the approximations $T^*_{s;j}$ by a relatively simply perturbation — namely, removing edges to form a spanning tree. On this basis, we first derive an exact expression relating expectations under two different distributions, from which we proceed to derive lower and upper bounds on the approximation error.

5.1 Exact expression

Our treatment begins at a slightly more general level, before specializing to the case of marginal distributions and the TRP algorithm. Consider a function $f : \mathcal{X} \to \mathbb{R}$, and two distributions $p(x;\widetilde{\theta})$ and $p(x;\theta)$. Suppose that we wish to express the expectation $\mathbb{E}_{\widetilde{\theta}}[f]$ in terms of an expectation over $p(x;\theta)$. Using the exponential representation of equation (6), it is straightforward to show that

$$\mathbb{E}_{\widetilde{\theta}}[f] \quad = \quad \mathbb{E}_{\theta}\left[\exp\Big\{ \sum_{\alpha} (\widetilde{\theta} - \theta)_{\alpha} \phi_{\alpha} + \Phi(\theta) - \Phi(\widetilde{\theta}) \Big\} \; f \right] \tag{37}$$

Note that this is a change of measure formula, where the exponentiated quantity can be viewed as the Radon-Nikodym derivative.

We now specialize equation (37) to the problem at hand. Let us denote the true single node marginal $p(x_s = j;\theta^0)$ on the loopy graph by

$$P_{s;j} \;\triangleq\; \mathbb{E}_{\theta^0}[\delta(x_s = j)] \;=\; \mathbb{E}_{\theta^*}[\delta(x_s = j)] \tag{38}$$

where we have used the invariance property (i.e., $p(x;\theta^0) = p(x;\theta^*)$) of Theorem 4.3. Assume that TRP (or BP) converges to some exponential parameter θ^*, with associated pseudomarginal vector T^*. In this case, Theorem 4.1 guarantees that the single-node elements of T^* can be interpreted as the following expectations:[18]

$$T^*_{s;j} \triangleq \mathbb{E}_{\Pi^i(\theta^*)}[\delta(x_s = j)] \tag{39}$$

[18] The tree-based consistency condition of Theorem 4.1 ensures that $T^*_{s;j} = \mathbb{E}_{\Pi^i(\theta^*)}[\delta(x_s = j)]$ independent of the choice of spanning tree index $i \in \{0, \ldots, L-1\}$.

We now make the assignments $\widetilde{\theta} = \theta^*$; $\theta = \Pi^i(\theta^*)$; and $f(x) = \delta(x_s = j)$ in equation (37) and re-arrange to obtain

$$P_{s;j} - T^*_{s;j} = \mathbb{E}_{\Pi^i(\theta^*)}\Big[\big(\exp\big\{\sum_{\alpha \notin \mathcal{A}^i} \theta^*{}_\alpha \phi_\alpha(x) - \Phi(\theta^*)\big\} - 1\big)\ \delta(x_s = j)\Big] \tag{40}$$

where we have used the fact that $\Phi(\Pi^i(\theta)) = 0$. Equation (40) is an exact expression for the error $(P_{s;j} - T^*_{s;j})$ in terms of an expectation over the tree-structured distribution $p(x; \Pi^i(\theta^*))$. Note that equation (40) holds for all spanning tree indices $i \in \{0, \ldots, L-1\}$.

5.2 Error bounds

It is important to observe that equation (40), though conceptually interesting, is of limited practical use. The problem stems from the presence of the term $\exp\{\sum_{\alpha \notin \mathcal{A}^i} \theta^*{}_\alpha \phi_\alpha\}$ within the expectation on the right-hand side. For most problems,[19] this computation will not be tractable, since it involves all the edges removed to form spanning tree $\mathcal{S}^i$.

This intractability motivates the idea of bounding the approximation error. In order to do so, we begin by considering the following problem: given distributions $p(x; \widetilde{\theta})$ and $p(x; \theta)$, and a function f on the state space, give a bound for the expectation $\mathbb{E}_{\widetilde{\theta}}[f]$ in terms of quantities computed using the distribution $p(x; \theta)$. The linearity of expectation allows us to assume without loss of generality that $f(x) \in [0, 1]$ for all $x \in \mathcal{X}$. If g is another function on the sample space $\mathcal{X}$, the covariance of f and g under $p(x; \theta)$ is defined as $\mathrm{cov}_\theta\{f,\, g\} = \mathbb{E}_\theta[f\, g] - \mathbb{E}_\theta[f]\mathbb{E}_\theta[g]$. With these definitions, we have the following result:

Proposition 5.1. Let $\widetilde{\theta}$ and θ be two arbitrary parameter vectors. Let $f \neq 0$ be a function on the state space such that $f(x) \in [0, 1]$ for all $x \in \mathcal{X}$. Then we have

$$\mathbb{E}_{\widetilde{\theta}}[f] \;\geq\; \mathbb{E}_\theta[f] \exp\Big\{ - D(\theta \parallel \widetilde{\theta}) + \frac{1}{\mathbb{E}_\theta[f]} \sum_{\alpha \in \mathcal{A}} \big(\widetilde{\theta} - \theta\big)_\alpha \,\mathrm{cov}_\theta\{f,\, \phi_\alpha\}\Big\} \tag{41}$$

Proof. See Appendix E. □

Applying Proposition 5.1 allows us to bound the approximation error in the single node marginals as follows:

Theorem 5.1. Let θ^* be a consistent fixed point of TRP/BP, giving rise to approximate marginal distributions $T^*_{s;j}$, and let $P_{s;j}$ be the true marginal distributions on the loopy graph. Then for each of the spanning trees $\mathcal{S}^i$, we have

$$\log T^*_{s;j} - \log P_{s;j} \;\leq\; D(\mathcal{I}^i(\Pi^i(\theta^*)) \parallel \theta^*) - \frac{1}{T^*_{s;j}} \sum_{\alpha \notin \mathcal{A}^i} \theta^*{}_\alpha \;\mathrm{cov}_{\Pi^i(\theta^*)}\{\delta(x_s = j),\, \phi_\alpha\} \tag{42}$$

$$\log T^*_{s;j} - \log P_{s;j} \;\geq\; \log T^*_{s;j} - \log\big[1 - (1 - T^*_{s;j})\exp(\Delta)\big] \tag{43a}$$

$$\Delta \;\triangleq\; -D(\mathcal{I}^i(\Pi^i(\theta^*)) \parallel \theta^*) - \frac{1}{1 - T^*_{s;j}} \sum_{\alpha \notin \mathcal{A}^i} \theta^*{}_\alpha \,\mathrm{cov}_{\Pi^i(\theta^*)}\{\delta(x_s = j),\, \phi_\alpha\} \tag{43b}$$

[19] Indeed, if the computation of equation (40) were easy for a particular graph, this would imply that we could compute the *exact* marginals, thereby obviating the need for an approximate algorithm such as BP/TRP.

Proof. We first make the identifications $\widetilde{\theta} = \theta^*$ and $\theta = \mathcal{I}^i(\Pi^i(\theta^*))$, and then set $f(x) = \delta(x_s = j)$, a choice which satisfies the assumptions of Proposition 5.1. Equation (42) then follows by applying Proposition 5.1, followed by some algebraic manipulation. The lower bound follows via the same argument applied to $f(x) = 1 - \delta(x_s = j)$, which also satisfies the restrictions of Proposition 5.1. □

A number of remarks about Theorem 5.1 are in order. For practical purposes, the primary consideration is the cost of computing these lower and upper bounds. The summations appearing in equations (42) and (43) are tractable. In particular, each of the covariances can be calculated by taking expectations over tree-structured distributions, and their weighted summation is even simpler. On the other hand, within the KL divergence $D(\mathcal{I}^i(\Pi^i(\theta^*)) \parallel \theta^*)$ lurks a log partition function $\Phi(\theta^*)$ associated with the loopy graph. In general, computing this quantity is as costly as performing inference on the original graph. However, there do exist techniques for either computing it approximately or bounding it [e.g., 26, 44, 24, 39]. With this caveat, the bounds given in Theorem 5.1 represent (approximately) computable bounds on the performance of the BP and TRP algorithms.

On the conceptual side, Theorem 5.1 highlights three factors that control the accuracy of the TRP/BP approximation. For the sake of concreteness, consider the upper bound of equation (42).

(a) the KL divergence $D(\mathcal{I}^i(\Pi^i(\theta^*)) \parallel \theta^*)$ measures the discrepancy between the tree-structured distribution $p(x; \Pi^i(\theta^*))$ and the distribution $p(x; \theta^*)$ on the graph with cycles. It will be small when the distribution $p(x; \theta^*)$ is well-approximated by a tree. This term reflects the empirical finding that BP performs well on graphs that are approximately tree-like (e.g., graphs with fairly long cycles).

(b) the covariance terms in the second summation measure the strength of the interaction, as measured under the tree distribution $p(x; \Pi^i(\theta))$, between the delta function at node s and clique potential ϕ_α. When the removed clique potential ϕ_α interacts only weakly with the delta function, then this covariance term will be small and so have little effect.

(c) the weight ${\theta^*}_\alpha$ on each covariance term measures the strength of the clique potentials $\{\phi_\alpha\}$ that were removed to form the spanning tree.

A number of extensions to the bounds presented in Theorem 5.1 are possible. First of all, it is worthwhile emphasizing that Theorem 5.1 provides L bounds on the single-node marginal $P_{s;j}$ — one for each of the L spanning trees used in the algorithm.[20] This allows us to choose the *tightest* of all the spanning tree bounds for a given index $(s; j)$. Point (b) above suggests that one should expect tighter bounds when using a tree formed by removing edges "relatively far" away from node s of interest. Indeed, the covariance between $\delta(x_s = j)$ and the removed clique potential ϕ_α captures this notion in a precise fashion.

Secondly, equation (37) as well as Proposition 5.1 both hold for arbitrary choices of the function $f : \mathcal{X} \to [0, 1]$. Different choices will allow us to derive bounds on the error of other approximate marginals. For instance, choosing $f(x) = \delta(x_s = j)\ \delta(x_t = k)$ will lead to bounds on the pairwise marginal $P_{st;jk}$. Thirdly, note that the bound of Proposition 5.1 (and the derived bounds in Theorem 5.1) are first-order, because they account for the interaction between the function f and clique potentials ϕ_α only up to first order (i.e., $\mathrm{cov}_\theta\{f, \phi_\alpha\}$). On the basis of equation (40), it is

[20] More generally, results of the form of Theorem 5.1 hold for any singly-connected subgraph embedded within the graph, not just the spanning trees used to implement the algorithm.

possible to derive stronger bounds by including higher order terms (as in [e.g., 32]), though with an associated price of increased computation. A thorough analysis of various bounds and the inherent tradeoffs is open for future work.

5.3 Illustrative examples of bounds

The tightness of the bounds given in Theorem 5.1 varies, depending on the graph topology, the choice of clique potentials, and the choice of spanning tree. Figure 11 illustrates the behavior for a binary-valued process on a simple 5-node graph under two conditions. The top panel of each subfigure shows the exact marginals $P_{s;0} = p(x_s = 0; \theta^*)$ versus the TRP approximations $T^*_{s;0}$ on a node-by-node basis, whereas the bottom part of each panel shows the error $\log T^*_{s;0} - \log P_{s;0}$, along with the associated bounds of Theorem 5.1. Subfigure (a) illustrates the case of

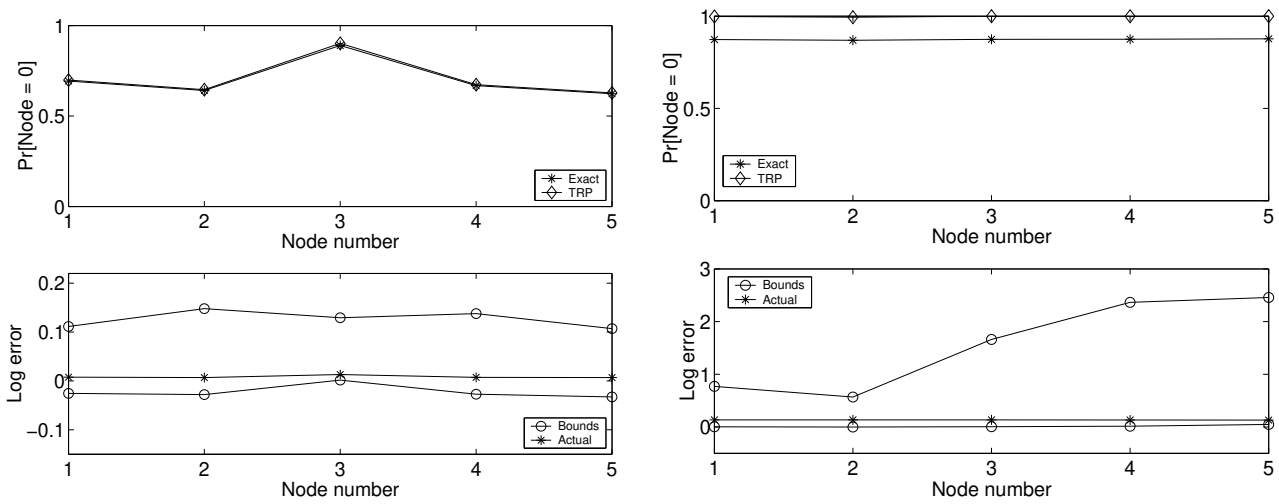


Figure 11. Behavior of the bounds of Theorem 5.1 for a simple 5-node graph under two conditions. The top panel in each subfigure (a) and (b) shows the exact marginals $P_{s;0}$ versus the TRP approximations $T^*_{s;0}$, whereas the bottom panel shows bounds on the corresponding error $\log T^*_{s;0} - \log P_{s;0}$. (a) For weak potentials, TRP gives a good approximation, so that the log error lies in a narrow band around zero. (b) For stronger potentials, the approximation is poor, and the lower bound on the log error is fairly tight.

weak potentials, so that BP or TRP leads to the very accurate approximation shown in the top panel. The corresponding bounds on the log error, plotted in the bottom panel, are also quite tight. Subfigure (b), in contrast, displays the more interesting choice of strong attractive clique potentials, for which the BP/TRP approximations tend to be skewed towards an extreme value (one in this case). The corresponding bounds on the log error are loose in comparison to those shown in subfigure (b). However, note that the lower bound is relatively tight. This shows that a *much* better approximation to the exact marginals could be obtained by adding in a correction term to the BP/TRP approximation. We are currently investigating conditions under which information from our error analysis can be incorporated efficiently to yield more accurate approximations.

6 Discussion

The contributions of this paper are both practical and theoretical. On the practical side, we have introduced a large class of iterative algorithms for approximate estimation of stochastic processes on graphs with cycles. Common to algorithms of this class is the operation of reparameterizing distributions defined on embedded trees. We showed, for example, that belief propagation (BP) is a special case of such an algorithm and derived a "message-free" implementation of BP that requires less computational storage than the traditional message-passing formulation. More generally, we considered updates that involve global operations over spanning trees of the graph. The convergence properties of these tree-based reparameterization (TRP) algorithms were shown to be superior to those of BP. There is great freedom in the detailed specification of TRP algorithms — in particular, in the choice of spanning trees (where the only constraint is that each edge in the loopy graph belong to at least one tree). This freedom suggests open questions of a graph-theoretic nature: how to choose a set of spanning trees so as to make the bounds of Section 5 as tight as possible, to guarantee convergence, or to optimize the rate of convergence. One possibility is to choose spanning trees randomly from the full graph, a task for which there exist well-known algorithms [e.g., 40].

The reparameterization framework also led to a number of important theoretical results. In particular, we obtained a new and intuitive characterization of the fixed points of any TRP algorithm (including BP as a special case). Even more generally, this fixed point characterization holds for any constrained local minimum of the Bethe free energy, whether obtained by a TRP algorithm, or an alternative minimization technique [e.g., 18, 50, 53]. We also proved a result which, though obvious from a reparameterization perspective, is nonetheless fundamental — namely, any TRP algorithm does not alter the full distribution on the loopy graph. This invariance has a number of important consequences, a subset of which we explored in this paper. In particular, in conjunction with the fixed point characterization, this invariance enabled us to derive an exact expression for the error between the TRP/BP approximations, and the exact marginals on an arbitrary loopy graph. We also derived upper and lower bounds on this error, which illuminate the conditions governing performance of such approximation methods. Note that the invariance property places severe restrictions on cases in which TRP or BP can be exact. A detailed treatment of this issue is left for future work.

This paper focused exclusively on reparameterization algorithms which involved only singleton and pairwise cliques. However, the reparameterization idea can be extended in a straightforward manner to include larger cliques (as in cluster variational methods [e.g., 29]). The analysis and results of this paper can be adapted to the case of higher-order cliques with little more than notational changes. For instance, the Möbius inversion formula [10] can be used to show how cost functions (analogous to the G of equation (24)) involving higher-order cliques correspond to more refined approximations of the KL divergence. The same fixed point characterization, invariance and error analysis will apply to these higher-order TRP algorithms.

A Proof of Proposition 4.1

We begin with some preliminary definitions and lemmas. For a closed and convex set X, we say that x^* is an *algebraic interior point* [11] if for all $x \neq x^*$ in X there exists $x' \in X$, and $\lambda \in (0,1)$ such that $x^* = \lambda x + (1-\lambda)x'$. Otherwise, x is an exterior point. The following lemma characterizes the nature of a constrained local minimum over X.

Lemma A.1. Let $f : X \to \mathbb{R}$ be a C^1 function, where X is a closed, convex and nonempty set. Suppose that x^* is a local minimum of f over X. Then

$$\nabla f(x^*)^T(x - x^*) \geq 0$$

for all $x \in X$. Moreover, if x^* is an algebraic interior point, then $\nabla f(x^*)^T(x - x^*) = 0$ for all $x \in X$.

Proof. See [8] for a proof of the first statement. To prove the second statement, assume that x^* is an algebraic interior point, so that for an arbitrary $x \in X$ we can write $x^* = \lambda x + (1-\lambda)x'$ for some x' and $\lambda \in (0,1)$. Then:

$$\begin{aligned} \nabla f(x^*)^T(x^* - x') &= \lambda \nabla f(x^*)^T(x - x') \geq 0 \\ \nabla f(x^*)^T(x^* - x) &= (1-\lambda)\nabla f(x^*)^T(x' - x) \geq 0 \end{aligned}$$

Since $\lambda \in (0,1)$, this establishes that $\nabla f(x^*)^T(x' - x) = 0$, and hence also $\nabla f(x^*)^T(x - x^*) = 0$. □

Lemma A.2. Let $U \in \mathcal{C}^i$ be arbitrary. Then for any θ such that $\widetilde{\theta} = \underline{\mathcal{R}}^i(\theta)$ is bounded:

$$\sum_{\alpha \in \mathcal{A}^i} \left\{ U_\alpha - \Lambda^i(\Pi^i(\theta))_\alpha \right\} [\widetilde{\theta} - \theta]_\alpha = 0 \tag{44}$$

Proof. We established in Section 4.1 that the point $\Lambda^i(\Pi^i(\theta))$ is the minimizing argument of the function G^i defined in equation (25) over the linear and hence convex set $\mathcal{C}^i$. This point will be an exterior point only if some element is equal to zero or one, a possibility that is prevented by the assumption that $\underline{\mathcal{R}}^i(\theta) = \mathcal{I}^i(\Theta^i(\Lambda^i(\Pi^i(\theta))))$ is bounded. Therefore, $\Lambda^i(\Pi^i(\theta))$ is an algebraic interior point, meaning that we can apply Lemma A.1 to conclude that for all $U \in \mathcal{C}^i$, we have

$$\sum_{\alpha \in \mathcal{A}^i} \left\{ U_\alpha - \Lambda^i(\Pi^i(\theta))_\alpha \right\} \frac{\partial G^i}{\partial T_\alpha}(\Lambda^i(\Pi^i(\theta)); \theta) = 0 \tag{45}$$

It remains to calculate the necessary partial derivatives of G^i. We begin with the decomposition $G^i(T;\theta) = \sum_{(s,t)\in\mathcal{E}^i} G^i_{st}(T_{st};\theta_{st}) + \sum_{s\in\mathcal{V}} G^i_s(T_s;\theta_s)$ where

$$\begin{aligned} G^i_{st}(T_{st}) &= \sum_{j,k} T_{st;jk} \left\{ \log \left[T_{st;jk} / (\sum_j T_{st;jk})(\sum_k T_{st;jk}) \right] - \theta_{st;jk} \right\} \\ G^i_s(T_s) &= \sum_j T_{s;j} \left[\log T_{s;j} - \theta_{s;j} \right] \end{aligned}$$

Using this decomposition, we calculate:

$$\frac{\partial G^i}{\partial T_\alpha}(T;\theta) = \begin{cases} \Theta(T)_{s;j} - \theta_{s;j} + 1 & \text{for } \alpha = (s;j) \\ \Theta(T)_{st;jk} - \theta_{st;jk} - 1 & \text{for } \alpha = (st;jk) \end{cases}$$

Substituting these quantities into equation (45), evaluated at $\widetilde{T} = \mathcal{I}^i(\Lambda^i(\Pi^i(\theta)))$, we obtain:

$$\sum_{s\in\mathcal{V}}\sum_{j}\left\{U-\widetilde{T}\right\}_{s;j}\left[\widetilde{\theta}-\theta+1\right]_{s;j}+\sum_{(s,t)\in\mathcal{E}^i}\sum_{j,k}\left\{U-\widetilde{T}\right\}_{st;jk}\left[\widetilde{\theta}-\theta-1\right]_{st;jk} \;=\; 0 \qquad (46)$$

where by definition $\widetilde{\theta}_\alpha = \Theta^i(\widetilde{T})_\alpha$ for each $\alpha \in \mathcal{A}^i$. Now since both U and $\widetilde{T}$ belong to $\mathcal{C}^i$, we have $\sum_j [U - \Lambda(\mathcal{Q}^i(\theta))]_{s;j} = 0$ for all $s \in \mathcal{V}$ and $\sum_{j,k}[U-\widetilde{T}]_{st;jk} = 0$ for all $(s,t) \in \mathcal{E}^i$. As a result, the constants 1 or -1 in equation (46) vanish in the sums over j or $\{j,k\}$, and we are left with the desired statement in equation (44). □

Equipped with these lemmas, we now establish equation (27) for $\lambda^n = 1$ by writing:

$$\begin{aligned} G(U;\theta) - G(U;\mathcal{Q}^i(\theta)) - G(\underline{\Lambda}^i(\mathcal{Q}^i(\theta));\theta) &= \sum_{\alpha\in\mathcal{A}}\left[U-\underline{\Lambda}^i(\mathcal{Q}^i(\theta))\right]_\alpha\left[\mathcal{Q}^i(\theta)-\theta\right]_\alpha \\ &= \sum_{\alpha\in\mathcal{A}^i}\left[U-\Lambda^i(\mathcal{Q}^i(\theta))\right]_\alpha\left[\Theta(\Lambda^i(\Pi^i(\theta)))-\theta\right]_\alpha \end{aligned}$$

where we used the fact that $\mathcal{Q}^i(\theta)_\alpha = \begin{cases} \theta_\alpha & \text{for all } \alpha \in \mathcal{A}/\mathcal{A}^i \\ \underline{\mathcal{R}}^i(\theta)_\alpha & \text{for all } \alpha \in \mathcal{A}^i. \end{cases}$
Since $\underline{\mathcal{R}}^i(\theta)$ is bounded by assumption, we can apply Lemma A.2 to conclude that

$$G(U;\theta) - G(U;\mathcal{Q}^i(\theta)) - G(\underline{\Lambda}^i(\mathcal{Q}^i(\theta));\theta) = 0 \qquad (47)$$

thereby establishing equation (27) for $\lambda^n = 1$, with the identifications $\theta \equiv \theta^n$ and $i = i(n)$.

To extend the result to $\lambda^n \in [0,1]$, we use the definition of θ^{n+1} given in equation (20) to write:

$$\begin{aligned} G(U;\theta^n) - G(U;\theta^{n+1}) &= \sum_\alpha U_\alpha\left[\theta^{n+1}-\theta^n\right]_\alpha \\ &= \lambda^n\sum_\alpha U_\alpha\left[\mathcal{Q}^{i(n)}(\theta^n)-\theta^n\right]_\alpha \\ &= \lambda^n\Big\{G(U;\theta^n) - G(U;\mathcal{Q}^{i(n)}(\theta^n))\Big\} \\ &= \lambda^n G(\underline{\Lambda}^{i(n)}(\mathcal{Q}^{i(n)}(\theta^n));\theta^n) \end{aligned}$$

where we have obtained the final line using equation (47).

B Proof of Theorem 4.1

(a): By the cyclic tree ordering, we have $\theta^{Ln+i+1} - \theta^{Ln+i} = \lambda^{Ln+i}[\mathcal{Q}^i(\theta^{Ln+i}) - \theta^{Ln+i}]$ where i is arbitrary in $\{0,\ldots,L-1\}$. Since the sequence $\{\theta^n\}$ converges to θ^*, it is Cauchy so that the left side tends to zero. Since $\lambda^{Ln+i} \geq \epsilon$, this implies that $\mathcal{Q}^i(\theta^{Ln+i}) - \theta^{Ln+i} \to 0$. I.e. $\mathcal{Q}^i(\theta^*) = \theta^*$ for all $i \in \{0,\ldots,L-1\}$.

We now construct the unique $T^* \in \mathcal{C}$ such that $\Pi^i(T^*) = \Lambda^i(\Pi^i(\theta^*))$. For an arbitrary index $\alpha \in \mathcal{A}$, pick a spanning tree $\mathcal{S}^i$ such that $\alpha \in \mathcal{A}^i$. This is always possible since $\cup_i \mathcal{A}^i = \mathcal{A}$ by construction. Define $T^*_\alpha = [\Lambda^i(\Pi^i(\theta^*))]_\alpha$, which is a consistent definition because

$$[\Lambda^i(\Pi^i(\theta^*))]_\alpha = [\Lambda^j(\Pi^j(\theta^*))]_\alpha$$

for any spanning tree indices i, j such that $\alpha \in \mathcal{A}^i \cap \mathcal{A}^j$. By construction, it is clear that $T^* \in \mathcal{C}$, and that $\Pi^i(T^*) = \Lambda^i(\Pi^i(\theta^*))$.

(b): Let $U \in \mathcal{C} = \cap_i \mathcal{C}^i$ be arbitrary. By applying Proposition 4.1 repeatedly, we obtain

$$G(U;\theta^0) = G(U;\theta^*) + \sum_{n=1}^{\infty} W_n \tag{48}$$

where $W_n \triangleq \lambda^n G(\underline{\Lambda}^{i(n)}(\mathcal{Q}^{i(n)}(\theta^n)); \theta^n)$. By part (a), the parameter θ^* induces a unique pseudo-marginal vector $T^* \in \cap_i \mathcal{C}^i$. We then apply equation (48) with $U = T^*$ and use the fact that $G(T^*;\theta^*) = 0$ by construction to obtain $G(T^*;\theta^0) = \sum_{n=1}^{\infty} W_n$. Substituting this result back into equation (48), we find that

$$G(U;\theta^0) = G(U;\theta^*) + G(T^*;\theta^0)$$

for all $U \in \cap_i \mathcal{C}^i$. To prove that T^* satisfies the necessary conditions to be a local minimum, we note that

$$\begin{aligned} G(U;\theta^0) - G(U;\theta^*) - G(T^*;\theta^0) &= \sum_{\alpha} \frac{\partial G}{\partial T_\alpha}(T^*;\theta^0)\big[U - T^*\big]_\alpha \\ &= 0 \end{aligned}$$

where we have used a sequence of steps similar to the proof of Proposition 4.1.

(c) Since the cost function G is bounded below and the constraint set is non-empty, the problem has at least one minimum. Moreover, because the constraint sets are linear, the existence of Lagrange multipliers is guaranteed for any local minimum [8]. By applying Farkas' lemma [8], the condition stated in (b) must be satisfied by any local minimum.

(d) Part (b) establishes that any fixed point T^* satisfies the necessary conditions to be a local minimum of G over the constraint set $\mathcal{C}$. The cost function G agrees with the Bethe free energy on this constraint set. Moreover, Yedidia et al. [52] have shown that BP fixed points correspond the points that satisfy the Lagrangian conditions for an extremum of the Bethe free energy over $\mathcal{C}$. By recourse to Farkas' lemma [8], these Lagrangian conditions are equivalent to the condition stated in (b). Therefore, we conclude that fixed points of the two algorithms coincide.

C Proof of Theorem 4.2

Throughout this appendix, we will use the notation $\Phi^i(\theta)$ as a shorthand for the quantity $\Phi(\Pi^i(\theta))$. With this notation, we begin with some preliminary lemmas.

Lemma C.1. For all indices $i = 0, \ldots, L-1$, we have $G^i(\Lambda^i(\Pi^i(\theta)); \Pi^i(\theta)) = -\Phi^i(\theta)$.

Proof. Note that $\Theta^i(\Lambda^i(\Pi^i(\theta)))$ and $\Pi^i(\theta)$ induce the same distribution on spanning tree $\mathcal{S}^i$ so that $D(\Theta(\Lambda^i(\Pi^i(\theta))) \,\|\, \Pi^i(\theta)) = 0$. The statement of the lemma then follows from equation (26). □

Lemma C.2. Let $l \in \{0, \ldots, L-1\}$ be arbitrary, and let $i(n)$ the tree index used at the n^{th} iteration. Then:

$$\Phi^l(\theta^{n+1}) \;\leq\; (1-\lambda^n)\Phi^l(\theta^n) + \lambda^n \Phi^l(\mathcal{Q}^{i(n)}(\theta^n)) \tag{49}$$

Moreover, in the special case $l = i(n)$, we have

$$\Phi^{i(n)}(\theta^{n+1}) \;=\; (1-\lambda^n)\Phi^{i(n)}(\theta^n) \tag{50}$$

Proof. Recall that θ^{n+1} is formed as the convex combination $\theta^{n+1} = (1-\lambda^n)\theta^n + \lambda^n \mathcal{Q}^{i(n)}(\theta^n)$. This combination remains convex if we apply the linear projection operator Π^l to both sides, so that equation (49) follows from the well-known convexity of the log partition function Φ.

In the special case $l = i(n)$, we have $\Phi^l(\mathcal{Q}^l(\theta^n)) = 0$, so that equation (49) reduces to $\Phi^l(\theta^{n+1}) \leq (1-\lambda^n)\Phi^l(\theta^n)$. Moreover, by the convexity of Φ:

$$\begin{aligned}
\Phi^l(\theta^{n+1}) &\geq \Phi^l(\theta^n) + \sum_{\alpha \in \mathcal{A}^l} \mathbb{E}_{\Pi^l(\theta^n)}[\phi_\alpha]\big[\theta^{n+1} - \theta^n\big]_\alpha && \text{(51a)}\\
&= \Phi^l(\theta^n) + \lambda^n \sum_{\alpha \in \mathcal{A}^l} \mathbb{E}_{\Pi^l(\theta^n)}[\phi_\alpha]\big[\mathcal{Q}^l(\theta^n) - \theta^n\big]_\alpha && \text{(51b)}\\
&= (1-\lambda^n)\Phi^l(\theta^n) && \text{(51c)}
\end{aligned}$$

where we have used the fact that $\frac{\partial \Phi}{\partial \theta_\alpha}(\Pi^l(\theta)) = \mathbb{E}_{\Pi^l(\theta)}[\phi_\alpha]$ to obtain equation (51a); the definition of θ^{n+1} in equation (51b); and Lemma C.1 in equation (51c). □

With these preliminary lemmas, we can begin the proof of the theorem. Let $U \in \mathcal{C}$ be arbitrary. By applying Proposition 4.1 repeatedly, for any iteration $M = 1, 2, \ldots$, we obtain

$$G(U;\theta^0) - G(U;\theta^M) = \sum_{n=0}^{M-1} W_n \tag{52}$$

where $W_n = \lambda^n G(\underline{\Lambda}^{i(n)}(\mathcal{Q}^{i(n)}(\theta^n)); \theta^n)$. Lemma C.1 and the definition of $\underline{\Lambda}^i$ in equation (17b) lead to the equivalent expressions:

$$\begin{aligned}
\Phi^i(\theta^n) &= G^i(\Lambda^i(\Pi^i(\theta^n)); \Pi^i(\theta^n)) && \text{(53a)}\\
&= G(\underline{\Lambda}^i(\mathcal{Q}^i(\theta^n)); \theta^n) && \text{(53b)}
\end{aligned}$$

meaning that we can write $W_n = -\lambda^n \Phi^{i(n)}(\theta^n)$.

From hereon, our goal is to establish that

$$\lim_{n\to\infty} \Phi^i(\theta^n) \;=\; 0 \quad \text{for} \;\; i = 0,\, 1 \tag{54}$$

Indeed, if equation (54) holds, then using equation (53), we see that assumption (a) implies that $\mathcal{Q}^i(\theta^n) \to \theta^n$ as well, which is the statement of the theorem. The essence of establishing (54) is to choose a sequence $\{\lambda^n\}$ of positive step sizes such that $W_n > 0$ is guaranteed for all $n = 0, 1, 2, \ldots$. This condition ensures that for some fixed $U \in \mathcal{C}$, the LHS of equation (52) — namely, the sequence $A_M \triangleq G(U;\theta^0) - G(U;\theta^M)$ — is non-decreasing in M. Moreover, since the sequence $\{\theta^M\}$ remains bounded by assumption, the sequence A_M is also bounded above. That is, the sequence

$\{A_M\}$ is both non-decreasing and bounded above, and so must converge. Using equation (52), the convergence of $\{A_M\}$ will allow us to conclude that $W_n \to 0$. Finally, we will use this fact to establish equation (54).

Without loss of generality, we assume that $\Phi^i(\theta^0) < 0$ for $i = 0, 1$, a condition that can be guaranteed by subtracting a constant from the full vector θ^0 if necessary. We formalize the step size choice in the following lemma:

Lemma C.3. At each iteration n, define:

$$\mu^n \triangleq \begin{cases} \left[\frac{-\Phi^i(\theta^n)}{\Phi^i(\mathcal{Q}^{i(n)}(\theta^n)) - \Phi^i(\theta^n)}\right] & \text{if } \Phi^i(\mathcal{Q}^{i(n)}(\theta^n)) > 0 \\ 1/(n+1) & \text{otherwise} \end{cases}$$

where $i(n) \equiv n (\mathrm{mod}\, 2)$ and $i \equiv (n+1) (\mathrm{mod}\, 2)$. Provided that $\Phi^i(\theta^0) < 0$ for $i = 0, 1$, then choosing the step sizes

$$\lambda^n = \frac{1}{2}\mu^n \tag{55}$$

will guarantee that $\Phi^i(\theta^n) < 0$ for all n, $i = 0, 1$.

Proof. The proof is by induction; the case $n = 0$ is given, and so we assume it holds for an even iteration n so that $i(n) = 0$. From equation (50) in Lemma C.2, if $\Phi^0(\theta^n) < 0$, then any step size in $(0, 1)$ will ensure that $\Phi^0(\theta^{n+1}) < 0$. Note that by construction $0 < \lambda^n < 1$, so that it is a valid step size choice.

Now considering $i = 1$: if $\Phi^1(\mathcal{Q}^0(\theta^n)) \leq 0$, then again any choice $\lambda^{n+1} < 1$ will suffice. On the other hand, if $\Phi^1(\mathcal{Q}^0(\theta^n)) > 0$, with the step size

$$0 \;<\; \mu^n \;=\; \frac{-\Phi^1(\theta^n)}{\Phi^1(\mathcal{Q}^0(\theta^n)) - \Phi^1(\theta^n)} \;<\; 1$$

the right hand side (i.e., upper bound) of equation (49) is zero. Since the upper bound of equation (49) decreases for smaller λ^n, the step size choice of equation (55) will ensure that $\Phi^1(\theta^{n+1}) < 0$.

A similar argument can be applied for odd n, where $i(n) = 1$. Therefore, we have established that our step size choice ensures that $\Phi^i(\theta^{n+1}) < 0$ for all n, and $i = 0, 1$. □

We now prove equation (54). By the step size choice of Lemma C.3 and our earlier reasoning, we are guaranteed that the infinite sum $\sum_{n=0}^{\infty} W_n$ exists, and that $W_n \to 0^+$. So as to exploit assumption (b) of the theorem statement, we now split our analysis into two cases. Note that for $a, b = 0, 1$, we have $\Phi^a(\mathcal{Q}^b(\theta^n)) = G(\underline{\Lambda}^a(\mathcal{Q}^b(\theta^n)); \mathcal{Q}^b(\theta^n))$ by definition of $\underline{\Lambda}^a$ in equation (17b). Therefore, assumption (b) means that the quantities $\Phi^1(\mathcal{Q}^0(\theta^n))$ and $\Phi^0(\mathcal{Q}^1(\theta^n))$ are eventually (i.e., for $n \geq L$) of the same sign.

Case 1: Suppose first that for $a, b = 0, 1$, we have $\Phi^a(\mathcal{Q}^b(\theta^n)) \leq 0$ for all $n \geq L$. This implies that $\lambda^n = \frac{1}{2(n+1)}$ for all $n \geq L$, so that the infinite sum $\sum_{n \geq L} W_n = -\sum_{n \geq L} \Phi^{i(n)}(\theta^n)/[2\,(n+1)]$ exists. Since $-\Phi^{i(n)}(\theta^n) > 0$ for all n by construction, this implies that $\Phi^i(\theta^n) \to 0$ for $i = 0, 1$.

Case 2: Otherwise for $a, b = 0, 1$ and $a \neq b$,[21] we have $\Phi^a(\mathcal{Q}^b(\theta^n)) > 0$ for all $n \geq L$. Let $\{n_k\}$ be the even integers for which $i(n_k) = 0$. Then we have:

$$\begin{aligned} \sum_{n_k \geq L}^{\infty} W_{n_k} &= -\sum_{n_k \geq L}^{\infty} \lambda^{n_k} \Phi^0(\theta^{n_k}) \\ &= \frac{1}{2} \sum_{n_k \geq L} \frac{\Phi^0(\theta^{n_k}) \Phi^1(\theta^{n_k})}{\Phi^1(\mathcal{Q}^0(\theta^{n_k})) - \Phi^1(\theta^{n_k})} \end{aligned} \tag{56}$$

Since the sequence $\{\theta^n\}$ remains bounded by assumption, the denominator of W_{n_k} is bounded in absolute value. Therefore, the fact that $W_{n_k} \to 0$ implies that the numerator — namely, $\Phi^0(\theta^{n_k})\Phi^1(\theta^{n_k})$ — must converge to zero. This condition does not necessarily imply that one of these two log partition functions converges to zero; for example, we could have $\Phi^0(\theta^{n_k})$ tending to zero for even k, and $\Phi^1(\theta^{n_k})$ tending to zero for odd k.

With the additional constraints of our problem, we shall now prove that, in fact we have $\lim_{n_k \to \infty} \Phi^1(\theta^{n_k}) = 0$. We proceed via proof by contradiction: if this were not the case, then there would exist some infinite subsequence (say $\{n_j\}$) of the even indices $\{n_k\}$ such that $\Phi^1(\theta^{n_j})$ is bounded away from zero. From the condition $\Phi^1(\theta^{n_j})\Phi^0(\theta^{n_j}) \to 0$, this implies that $\Phi^0(\theta^{n_j}) \to 0$. By assumption (a) and the equivalence of Lemma C.1, this implies that $[\mathcal{Q}^0(\theta^{n_j}) - \theta^{n_j}] \to 0$. Since Φ^1 is a C^2 function and $\{\theta^n\}$ is bounded, we can apply the mean value theorem to conclude that $\lim_{n_j \to \infty} \inf \left[\Phi^1(\theta^{n_j}) - \Phi^1(\mathcal{Q}^0(\theta^{n_j}))\right] = 0$. Moreover, since $\Phi^1(\mathcal{Q}^0(\theta^{n_j})) > 0$ for all $n_j \geq L$ by assumption, we have

$$\begin{aligned} \lim_{n_j \to \infty} \inf \Phi^1(\theta^{n_j}) &\geq \lim_{n_j \to \infty} \inf \left[\Phi^1(\theta^{n_j}) - \Phi^1(\mathcal{Q}^0(\theta^{n_j}))\right] \\ &= 0 \end{aligned}$$

Moreover, by our step size choice, we have $\Phi^1(\theta^{n_j}) < 0$ for all n_j, thereby ensuring the relation $\lim_{n_j \to \infty} \sup \Phi^1(\theta^{n_j}) \leq 0$. In conjunction, these two relations imply that $\lim_{n_j \to \infty} \Phi^1(\theta^{n_j})$ exists, and is equal to zero, so that we have reached a contradiction. Therefore, our initial assumption must have been false, and we can conclude that $\lim_{n_k \to \infty} \Phi^1(\theta^{n_k}) = 0$.

On the other hand, to analyze the behavior of $\Phi^0(\theta^n)$, consider the sequence formed by the odd indices $\{m_k\}$. This leads to equation (56), with the roles of 0 and 1 interchanged. Thus, a similar argument allows us to establish that $\Phi^0(\theta^{m_k}) \to 0$.

Therefore, we have proved that $\Phi^1(\theta^{n_k}) \to 0$ and $\Phi^0(\theta^{m_k}) \to 0$. These conditions in conjunction imply that the step sizes λ^n are tending to zero for the infinite subsequences formed by even indices $\{n_k\}$ and odd indices $\{m_k\}$. Therefore, we can conclude that the overall sequence θ^n converges to some θ^* such that $\Phi^i(\theta^*) = 0$, and hence equation (54) is proved, which establishes the theorem.

D Proof of Proposition 4.2

We begin by expressing the delta function $\delta(x_s = j)$ as a linear combination of the monomials in the set $\mathcal{R}(s)$ defined in equation (32a) as follows:

$$\delta(x_s = j) = \prod_{k \neq j} \frac{(k - x_s)}{(k - j)} \tag{57}$$

[21] Note that by definition, $\Phi^a(\mathcal{Q}^a(\theta^n)) = 0$ for $a = 0, 1$.

This decomposition is extended readily to pairwise delta functions, which are defined by products $\delta(x_s = j)\delta(x_t = k)$; in particular, they can be written as linear combinations of elements in the sets $\mathcal{R}(s)$ and $\mathcal{R}(s,t)$, as defined in equation (32a) and equation (32b), respectively. Now suppose that $\theta \in \mathcal{M}(\theta^0)$, so that $\log p(x;\gamma^0) = \log p(x;\theta)$ for all $x \in \mathcal{X}$. By construction, both the LHS and RHS are linear combination of the elements $\mathcal{R}(s)$ and $\mathcal{R}(s,t)$. Equating the coefficients of these terms yields a set of $d(\gamma) = (m-1)n + (m-1)^2 e$ linear equations. We write these equations compactly in matrix form as $A\theta = \gamma^0$.

This establishes the necessity of the linear manifold constraints. To establish their sufficiency, we need only check that the linear constraints ensure the constant terms in $\log p(x;\gamma^0)$ and $\log p(x;\theta)$ (i.e., $\Phi(\gamma^0)$ and $\Phi(\theta)$ respectively) are also equal. This is a straightforward verification.

E Proof of Proposition 5.1

Consider the function

$$F(\theta) \triangleq \log \sum_{x \in \mathcal{X}} \left[\exp\left\{ \sum_{\alpha} \theta_\alpha \phi_\alpha \right\} f(x) \right]$$

It can be seen that F is the log partition function corresponding to the distribution $q(x;\theta) \propto \exp\left\{\sum_\alpha \theta_\alpha \phi_\alpha\right\} f(x)$, and therefore is convex. For any parameter vectors $\widetilde{\theta}$ and θ, this convexity allows to write:

$$F(\widetilde{\theta}) \quad \geq \quad F(\theta) + \sum_{\alpha} \frac{\partial F}{\partial \theta_\alpha} (\widetilde{\theta} - \theta)_\alpha \tag{58}$$

The partial derivative can be calculated as $\frac{\partial F}{\partial \theta_\beta} = \mathbb{E}_\theta[f\ \phi_\beta]/\mathbb{E}_\theta[f]$. We also observe the relation $F(\theta) = \log \mathbb{E}_\theta[f] + \Phi(\theta)$. Substituting these relations into equation (58), rearranging and taking exponentials yields the bound of equation (41).

References

[1] S. Aji, G. Horn, and R. McEliece. On the convergence of iterative decoding on graphs with a single cycle. In *Proceedings IEEE Intl. Symp. on Information Theory*, page 276, Cambridge, MA, 1998.

[2] S.M. Aji and R.J. McEliece. The generalized distributive law. *IEEE Trans. Info. Theory*, 46:325–343, March 2000.

[3] S. Amari. Differential geometry of curved exponential families — curvatures and information loss. *Annals of Statistics*, 10(2):357–385, 1982.

[4] S. Amari, K. Kurata, and H. Nagaoka. Information geometry of Boltzmann machines. *IEEE Trans. on Neural Networks*, 3(2):260–271, 1992.

[5] J. B. Anderson and S. M. Hladnik. Tailbiting map decoders. *IEEE Sel. Areas Comm.*, 16:297–302, February 1998.

[6] O. E. Barndorff-Nielson. *Information and exponential families.* Wiley, Chichester, 1978.

[7] M. Basseville, A. Benveniste, K. Chou, S. Golden, R. Nikoukhah, and A. Willsky. Modeling and estimation of multiresolution stochastic processes. *IEEE Transactions on Information Theory*, 38(2):766–784, March 1992.

[8] D.P. Bertsekas. *Nonlinear programming.* Athena Scientific, Belmont, MA, 1995.

[9] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Roy. Stat. Soc. Series B*, 36:192–236, 1974.

[10] P. Brémaud. *Markov chains, Gibbs fields, Monte Carlo simulation, and queues.* Springer, 1991.

[11] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications.* Numerical Mathematics and Scientific Computation. Oxford University Press, 1988.

[12] N. N. Chentsov. *Statistical Decision Rules and Optimal Inference*, volume 53 of *Translations of Mathematical Monographs.* American Mathematical Society, 1982.

[13] G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.

[14] T.M. Cover and J.A. Thomas. *Elements of Information Theory.* John Wiley and Sons, New York, 1991.

[15] I. Csiszar. I-divergence geometry of probability distributions and minimization problems. *Annals of Probability,*, 3(1):146–158, Feb. 1975.

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

[17] J.W. Demmel. *Applied numerical linear algebra.* SIAM, Philadelphia, 1997.

[18] D. Forney and M. Chiang. Statistical physics, convex optimization, and the sum-product algorithm. Preprint, March 2001.

[19] W. Freeman and Y. Weiss. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Trans. Info. Theory*, 47:736–744, 2001.

[20] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Intl. J. Computer Vision*, 40(1):25–47, 2000.

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

[22] G. R. Grimmett. A theorem about random fields. *Bulletin of the London Mathematical Society*, 5:81–84, 1973.

[23] S. S. Gupta, editor. *Differential geometry in statistical inference*, volume 10 of *Lecture notes – Monograph series.* Institute of Mathematical Statistics, Hayward, CA, 1987.

[24] T. Jaakkola and M. Jordan. Recursive algorithms for approximating probabilities in graphical models. In *Advances in Neural Information Processing Systems*, volume 9, 1996.

[25] F. V. Jensen. *An introduction to Bayesian networks.* UCL Press, London, 1996.

[26] M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal Comput.*, 22:1087–1116, 1993.

[27] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. In *Learning in graphical models*, pages 105–161. MIT Press, 1999.

[28] G. D. Forney Jr., F. R. Kschischang, B. Marcus, and S. Tuncel. *Codes, systems and graphical models*, chapter Iterative decoding of tail-biting trellises and connections with symbolic dynamics, pages 239–264. Springer, 2001.

[29] R. Kikuchi. The theory of cooperative phenomena. *Physical Review*, 81:988–1003, 1951.

[30] F. Kschischang and B. Frey. Iterative decoding of compound codes by probability propagation in graphical models. *IEEE Sel. Areas Comm.*, 16(2):219–230, February 1998.

[31] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society B*, 50:155–224, January 1988.

[32] M.A.R. Leisink and H.J. Kappen. A tighter bound for graphical models. In *NIPS 13*, pages 266–272. MIT Press, 2001.

[33] M. Luettgen, W. Karl, and A. Willsky. Efficient multiscale regularization with application to optical flow. *IEEE Trans. Im. Proc.*, 3(1):41–64, Jan. 1994.

[34] R.J. McEliece, D.J.C. McKay, and J.F. Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Jour. Sel. Communication*, 16(2):140–152, February 1998.

[35] T. P. Minka. *A family of algorithms for approximate Bayesian inference.* PhD thesis, MIT Media Lab, 2001.

[36] K. Murphy, Y. Weiss, and M. Jordan. Loopy-belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence*, volume 9, 1999.

[37] G. Parisi. *Statistical field theory.* Addison-Wesley, 1988.

[38] J. Pearl. *Probabilistic reasoning in intelligent systems.* Morgan Kaufman, San Mateo, 1988.

[39] G. Potamianos and J. Goutsias. Stochastic approximation algorithms for partition function estimation of Gibbs random fields. *IEEE Trans. Info. Theory*, 43(6):1948–1965, November 1997.

[40] J. G. Propp and D. B. Wilson. How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *Journal of Algorithms*, 27:170–217, 1998.

[41] T. Richardson. The geometry of turbo-decoding dynamics. *IEEE Trans. Info. Theory*, 46(1):9–23, January 2000.

[42] G. Rockafeller. *Convex Analysis.* Princeton University Press, Princeton, 1970.

[43] P. Rusmevichientong and B. Van Roy. An analysis of turbo decoding with Gaussian densities. In *NIPS 12*, pages 575–581. MIT Press, 2000.

[44] L. Saul, T. Jaakkola, and M. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

[45] K. Tanaka and T. Morita. Cluster variation method and image restoration problem. *Physics Letters A*, 203:122–128, 1995.

[46] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree-based reparameterization for approximate estimation on graphs with cycles. LIDS Tech. report P-2510: available at `http://ssg.mit.edu/group/mjwain/mjwain.shtml`, May 2001.

[47] M. J. Wainwright, E. B. Sudderth, and A. S. Willsky. Tree-based modeling and estimation of Gaussian processes on graphs with cycles. In *Advances in Neural Information Processing Systems 13*, pages 661–667. MIT Press, 2001. Paper available at http://ssg.mit.edu/group/mjwain/mjwain.shtml.

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

[49] Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. In *NIPS 12*, pages 673–679. MIT Press, 2000.

[50] M. Welling and Y. Teh. Belief optimization: A stable alternative to loopy belief propagation. Preprint, May 2001.

[51] C. H. Wu and Peter C. Doerschuk. Tree approximations to Markov random fields. *IEEE Trans. on PAMI*, 17(4):391–402, April 1995.

[52] J. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *NIPS 13*, pages 689–695. MIT Press, 2001.

[53] A. Yuille. A double-loop algorithm to minimize the Bethe and Kikuchi free energies. Preprint, March 2001.