Journal of Machine Learning Research 14 (2013) 2287-2314 Submitted 6/12; Revised 2/13; Published 8/13

Message-Passing Algorithms for Quadratic Minimization

Nicholas Ruozzi NICHOLAS.RUOZZI@EPFL.CH
Communication Theory Laboratory
École Polytechnique Fédérale de Lausanne
Lausanne 1015, Switzerland

Sekhar Tatikonda SEKHAR.TATIKONDA@YALE.EDU
Department of Electrical Engineering
Yale University
New Haven, CT 06520, USA

Editor: Martin Wainwright

Abstract

Gaussian belief propagation (GaBP) is an iterative algorithm for computing the mean (and variances) of a multivariate Gaussian distribution, or equivalently, the minimum of a multivariate positive definite quadratic function. Sufficient conditions, such as walk-summability, that guarantee the convergence and correctness of GaBP are known, but GaBP may fail to converge to the correct solution given an arbitrary positive definite covariance matrix. As was observed by Malioutov et al. (2006), the GaBP algorithm fails to converge if the computation trees produced by the algorithm are not positive definite. In this work, we will show that the failure modes of the GaBP algorithm can be understood via graph covers, and we prove that a parameterized generalization of the min-sum algorithm can be used to ensure that the computation trees remain positive definite whenever the input matrix is positive definite. We demonstrate that the resulting algorithm is closely related to other iterative schemes for quadratic minimization such as the Gauss-Seidel and Jacobi algorithms. Finally, we observe, empirically, that there always exists a choice of parameters such that the above generalization of the GaBP algorithm converges.

Keywords: belief propagation, Gaussian graphical models, graph covers

1. Introduction

Let $\Gamma \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix and $h \in \mathbb{R}^n$. The quadratic minimization problem is to find the $x \in \mathbb{R}^n$ that minimizes $f(x) = \frac{1}{2}x^T\Gamma x - h^T x$. Minimizing a positive definite quadratic function is equivalent to computing the mean of a multivariate Gaussian distribution with a positive definite covariance matrix, or equivalently, solving the positive definite linear system $\Gamma x = h$ for the vector x.

Because of the importance of solving linear systems, many different algorithms for the minimization of quadratic functions have been studied: Gaussian elimination, Gauss-Seidel iteration, Jacobi iteration, successive over-relaxation, etc. Distributed algorithms that can take advantage of the sparsity of the matrix Γ are particularly desirable for solving large-scale systems as even Gaussian elimination may be too time consuming in practice. In this work, we study Gaussian graphical models and an inference algorithm known as Gaussian belief propagation (GaBP): an iterative message-passing scheme that can be used to estimate the mean of a Gaussian distribution as well as individual variances. In the study of graphical models, Gaussian graphical models are especially

important because they represent one of the few continuous distributions for which exact message updates can be efficiently computed and more complicated distributions are often approximated by Gaussian distributions in practice.

In previous work, several authors have provided sufficient conditions for the convergence of GaBP. Weiss and Freeman (2001b) demonstrated that GaBP converges in the case that the covariance matrix is diagonally dominant. Malioutov et al. (2006) proved that the GaBP algorithm converges when the covariance matrix is walk-summable. Moallemi and Van Roy (2009, 2010) showed that scaled diagonal dominance was a sufficient condition for convergence and also characterized the rate of convergence via a computation tree analysis. The latter two sufficient conditions, walk-summability and scaled diagonal dominance, are known to be equivalent (Malioutov, 2008; Ruozzi et al., 2009).

While the above conditions are sufficient for the convergence of the GaBP algorithm they are not necessary: there are examples of positive definite matrices that are not walk-summable for which the GaBP algorithm still converges to the correct solution (Malioutov et al., 2006). A critical component of these examples is that the computation trees remain positive definite throughout the algorithm. Such behavior is guaranteed if the original matrix is scaled diagonally dominant, but arbitrary positive definite matrices can produce computation trees that are not positive definite (Malioutov et al., 2006). If this occurs, the standard GaBP algorithm fails to produce the correct solution. The purpose of this work is to understand why GaBP fails and to design iterative message-passing algorithms with improved convergence guarantees outside of the walk-summable case.

Related work studies the effect of preconditioning: the covariance matrix is preconditioned in order to force it to be scaled diagonally dominant and then the GaBP algorithm is applied to solve the preconditioned problem. Diagonal loading was proposed as one such useful preconditioner (Johnson et al., 2009). The key insight of diagonal loading is that scaled diagonal dominance can be achieved by sufficiently weighting the diagonal elements of the inverse covariance matrix. The diagonally loaded matrix can then be used as an input to a GaBP subroutine. The solution produced by GaBP is then used in a feedback loop to produce a new matrix, and the process is repeated until a desired level of accuracy is achieved. The performance is determined, in part, by the amount of diagonal loading needed to make the matrix scaled diagonally dominant. However, choosing the appropriate amount of diagonal loading that achieves the fastest rate of convergence remains an open question. As the approach in this work is to study reweighted versions of GaBP, our techniques can also be applied as part of this algorithmic scheme (or by themselves), and we provide experimental evidence that this technique results in performance gains over GaBP, allowing for less diagonal loading.

The GaBP algorithm can be seen as a special case of a standard message-passing algorithm known as the min-sum algorithm. Other recent work has studied provably convergent variants of the min-sum algorithm. The result has been the development of many different "convergent and correct" message-passing algorithms: MPLP (Globerson and Jaakkola, 2007), max-sum diffusion (Werner, 2007), norm-product belief propagation (Hazan and Shashua, 2010), and tree-reweighted belief propagation (Wainwright et al., 2005). Each of these algorithms can be viewed as a coordinate ascent/descent scheme for an appropriate lower/upper bound. Sontag and Jaakkola (2009) and Meltzer et al. (2009) provide a general overview of these techniques and their relationship to bound maximization. These algorithms guarantee convergence under an appropriate message-passing schedule, and they also guarantee correctness if a unique assignment can be extracted upon convergence. Such algorithms are plausible candidates in the search for convergent iterative algo-

rithms for the quadratic minimization problem, but as we show using the theory of graph covers, iterative schemes based on dual optimization techniques that guarantee the correctness of locally decodable beliefs cannot converge to the correct minimizing assignment outside of walk-summable models.

In this work, we investigate the behavior of reweighted message-passing algorithms for the quadratic minimization problem. The motivation for this study comes from the observation that belief propagation style algorithms typically do not explore all nodes in the factor graph with the same frequency (Frey et al., 2001). In many application areas, such uneven counting is undesirable and typically results in incorrect answers, but if we can use reweighting to overestimate the diagonal entries of the computation tree relative to the off diagonal entries, then we may be able to force the computation trees to be positive definite at each iteration of the algorithm. Although similar in spirit to diagonal loading, our approach obviates the need for preconditioning. We will show that there exists a choice of parameters for the reweighted algorithms that guarantees monotone convergence of the variance estimates on all positive definite models, even those for which the GaBP algorithm fails to converge. We empirically observe that there exists a choice of parameters that also guarantees the convergence of the mean estimates. In addition, we show that our graph cover analysis extends to other iterative algorithms for the quadratic minimization problem and that similar ideas can be used to reason about the min-sum algorithm for general convex minimization.

The outline of this paper is as follows. In Section 2 we review the min-sum algorithm, its reweighted generalizations, and the quadratic minimization problem. In Section 3 we discuss the relationship between pairwise message-passing algorithms and graph covers, and we show how to use graphs covers to characterize walk-summability. In Section 4, we examine the convergence of the means and the variances under the reweighted algorithm for the quadratic minimization problem, we explore the relationship between the reweighted algorithm and the Gauss-Seidel and Jacobi methods, and we compare the performance of the reweighted algorithm to the standard min-sum algorithm. Finally, in Section 5, we summarize the results and discuss extensions of this work to general convex functions as well as open problems. Detailed proofs of the two main theorems can be found in Appendices A and B.

2. Preliminaries

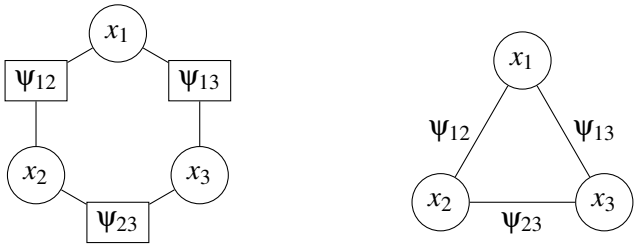
In this section, we review the min-sum algorithm and a reweighted variant over pairwise factor graphs. Of particular importance for later proofs will be the computation trees generated by each of these algorithms. We also review the quadratic minimization problem, and explain the closed-form message updates for this problem.

2.1 The Min-Sum Algorithm

The min-sum algorithm attempts to compute the minimizing assignment of an objective function $f : \prod_i \mathcal{X}_i \to \mathbb{R}$ that, given a graph $G = (V,E)$, can be factorized as a sum of self-potentials and edge potentials as follows.

$$f(x_1,...,x_n) = \sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E} \psi_{ij}(x_i,x_j)$$

We assume that this minimization problem is well-defined: f is bounded from below and there exists an $x \in \prod_i \mathcal{X}_i$ that minimizes f.



(a) Factor graph. (b) The graph G.

Figure 1: The factor graph corresponding to $f(x_1,x_2,x_3) = \phi_1 + \phi_2 + \phi_3 + \psi_{12} + \psi_{23} + \psi_{13}$ and the graph G. The functions ϕ_1,ϕ_2, and ϕ_3 each depend on only one variable, and are typically omitted from the factor graph representation for clarity.

To each factorization, we associate a bipartite graph known as the factor graph. In general, the factor graph consists of a node for each of the variables $x_1,...,x_n$, a node for each of the ψ_{ij}, and for all $(i,j) \in E$, an edge joining the node corresponding to x_i to the node corresponding to ψ_{ij}. Because the ψ_{ij} each depend on exactly two factors, we often omit the factor nodes from the factor graph construction and replace them with a single edge. This reduces the factor graph to the graph G. See Figure 1 for an example of this construction.

We can write the min-sum algorithm as a local message-passing algorithm over the graph G. During the execution of the min-sum algorithm, messages are passed back and forth between adjacent nodes of the graph. On the t^{th} iteration of the algorithm, messages are passed along each edge of the factor graph as

$$m^t_{i\to j}(x_j) \quad = \quad \kappa + \min_{x_i}\left[\psi_{ij}(x_i,x_j) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} m^{t-1}_{k\to i}(x_i)\right],$$

where ∂i denotes the set of neighbors of the node i in G and $\partial j \setminus i$ is abusive notation for the set-theoretic difference $\partial j \setminus \{i\}$. When the factor graph is a tree, these updates are guaranteed to converge, but understanding when these updates converge to the correct solution for an arbitrary graph is a central question underlying the study of the min-sum algorithm.

Each message update has an arbitrary normalization factor κ. Because κ is not a function of any of the variables, it only affects the value of the minimum and not where the minimum is located. As such, we are free to choose it however we like for each message and each time step. In practice, these constants are used to avoid numerical issues that may arise during the execution of the algorithm.

We will think of the messages as a vector of functions indexed by the edge over which the message is passed. Any vector of real-valued messages is a valid choice for the vector of initial messages m^0, and the choice of initial messages can greatly affect the behavior of the algorithm. A typical assumption, that we will use in this work, is that the initial messages are chosen such that $m^0_{i\to j} \equiv 0$ for all i and j.

Given any vector of messages, m^t, we can construct a set of beliefs that are intended to approximate the min-marginals of f as

$$\begin{aligned}\tau^t_i(x_i) &= \kappa + \phi_i(x_i) + \sum_{j\in\partial i} m^t_{j\to i}(x_i),\\ \tau^t_{ij}(x_i,x_j) &= \kappa + \psi_{ij}(x_i,x_j) + \tau^t_j(x_j) - m^t_{i\to j}(x_j) + \tau^t_i(x_i) - m^t_{j\to i}(x_i).\end{aligned}$$

Additionally, we can approximate the optimal assignment by computing an estimate of the argmin,

$$x_i^t \quad \in \quad \arg\min_{x_i} \tau_i^t(x_i).$$

If the beliefs are equal to the true min-marginals of f (i.e., $\tau_i^t(x_i) = \min_{x' : x'_i = x_i} f(x')$), then for any $y_i \in \arg\min_{x_i} \tau_i^t(x_i)$ there exists a vector x^* such that $x_i^* = y_i$ and x^* minimizes the function f. If $|\arg\min_{x_i} \tau_i^t(x_i)| = 1$ for all i, then we can take $x^* = y$, but, if the objective function has more than one optimal solution, then we may not be able to construct such an x^* so easily.

Definition 1 *A vector, $\tau = (\{\tau_i\}, \{\tau_{ij}\})$, of beliefs is **locally decodable** to x^* if $\tau_i(x_i^*) < \tau_i(x_i)$ for all i, $x_i \neq x_i^*$. Equivalently, for each $i \in V$, τ_i is uniquely minimized at x_i^*.*

If the algorithm converges to a vector of beliefs that are locally decodable to x^*, then we hope that the vector x^* is a global minimum of the objective function. This is indeed the case when the factor graph contains no cycles (Wainwright et al., 2004) but need not be the case for arbitrary graphical models.

2.1.1 COMPUTATION TREES

An important tool in the analysis of the min-sum algorithm is the notion of a computation tree. Intuitively, the computation tree is an unrolled version of the original graph that captures the evolution of the messages passed by the min-sum algorithm needed to compute the belief at time t at a particular node of the factor graph. Computation trees describe the evolution of the beliefs over time, which, in some cases, can help us prove correctness and/or convergence of the message-passing updates. For example, the convergence of the min-sum algorithm on graphs containing a single cycle can be demonstrated by analyzing the computation trees produced by the min-sum algorithm at each time step (Weiss, 2000).

The depth t computation tree rooted at node i contains all of the length t non-backtracking walks in the factor graph starting at node i. A walk is non-backtracking if it does not go back and forth successively between two vertices. For any node v in the factor graph, the computation tree at time t rooted at i, denoted by $T_i(t)$, is defined recursively as follows. $T_i(0)$ is just the node i, the root of the tree. The tree $T_i(t)$ at time $t > 0$ is generated from $T_i(t-1)$ by adding to each leaf of $T_i(t-1)$ a copy of each of its neighbors in G (and the corresponding edge), except for the neighbor that is already present in $T_i(t-1)$. Each node of $T_i(t)$ is a copy of a node in G, and the potentials on the nodes in $T_i(t)$, which operate on a subset of the variables in $T_i(t)$, are copies of the potentials of the corresponding nodes in G. The construction of a computation tree for the graph in Figure 1 is pictured in Figure 2. Note that each variable node in $T_i(t)$ represents a distinct copy of some variable x_j in the original graph.

Given any initialization of the messages, $T_i(t)$ captures the information available to node i at time t. At time $t = 0$, node i has received only the initial messages from its neighbors, so $T_i(0)$ consists only of i. At time $t = 1$, i receives the round one messages from all of its neighbors, so i's neighbors are added to the tree. These round one messages depend only on the initial messages, so the tree terminates at this point. By construction, we have the following lemma.

Lemma 2 *The belief at node i produced by the min-sum algorithm at time t corresponds to the exact min-marginal at the root of $T_i(t)$ whose boundary messages are given by the initial messages.*

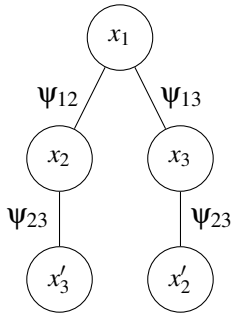


Figure 2: The computation tree at time $t = 2$ rooted at the variable node x_1 of the graph in Figure 1. The self-potentials corresponding to each variable node are given by the subscript of the variable.

Proof Tatikonda and Jordan (2002) and Weiss and Freeman (2001a) both provide a proof of this lemma. ■

Computation trees provide a dynamic view of the min-sum algorithm. After a finite number of time steps, we hope that the beliefs at the root of the computation trees stop changing and that the message vector converges to a fixed point of the message update equations (in practice, when the beliefs change by less than some small amount, we say that the algorithm has converged). For any real-valued objective function f (i.e., $|f(x)| < \infty$ for all x), there always exists a fixed point of the message update equations (see Theorem 2 of Wainwright et al. 2004).

2.2 Reweighted Message-Passing Algorithms

Because the min-sum algorithm is not guaranteed to converge and, even it does, is not guaranteed to compute the correct minimizing assignment, recent research has focused on the design of alternative message-passing schemes that do not suffer from these drawbacks. Efforts to produce provably convergent message-passing schemes have resulted in the reweighted message-passing algorithm described in Algorithm 1. This algorithm is parameterized by a vector of non-zero real weights for each edge of the graph. Notice that if we set $c_{ij} = 1$ for all i and j, then we obtain the standard min-sum algorithm. Wainwright et al. (2005) choose the c_{ij} in a specific way in order to guarantee correctness of the algorithm (which they call TRMP in this special case). In this work, we will focus on choices of these weights that will guarantee convergence of the algorithm for the quadratic minimization problem. These choices will, surprisingly, not coincide with those of the TRMP algorithm. In fact, the choice of weights that guarantees correctness of the TRMP algorithm must necessarily cause the algorithm to either not converge or converge to the incorrect solution whenever the given matrix is not walk-summable.

The beliefs for the reweighted algorithm are defined analogously to those for the standard min-sum algorithm.

$$\begin{aligned}
\tau_i^t(x_i) &= \kappa + \phi_i(x_i) + \sum_{j \in \partial i} c_{ji} m_{j\to i}^t(x_i), \\
\tau_{ij}^t(x_i, x_j) &= \kappa + \frac{\psi_{ij}(x_i, x_j)}{c_{ij}} + \tau_j^t(x_j) - m_{i\to j}^t(x_j) + \tau_i^t(x_i) - m_{j\to i}^t(x_i)
\end{aligned}$$

Algorithm 1 Synchronous Reweighted Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: For iteration $t = 1, 2, \ldots$ update the the messages as follows:

$$m_{i\to j}^t(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + \phi_i(x_i) + (c_{ij}-1)m_{j\to i}^{t-1}(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}^{t-1}(x_i)\right].$$

The vector of messages at any fixed point of the message update equations has two important properties. First, the beliefs corresponding to these messages provide an alternative factorization of the objective function f. Second, the beliefs correspond to approximate marginals.

Lemma 3 *For any vector of messages m^t with corresponding beliefs τ^t,*

$$f(x_1,\ldots,x_{|V|}) = \kappa + \sum_{i\in V}\tau_i^t(x_i) + \sum_{(i,j)\in E} c_{ij}\left[\tau_{ij}^t(x_i,x_j) - \tau_i^t(x_i) - \tau_j^t(x_j)\right].$$

Lemma 4 *If τ is a set of beliefs corresponding to a fixed point of the message updates in Algorithm 1, then*

$$\min_{x_j}\tau_{ij}(x_i,x_j) = \kappa + \tau_i(x_i)$$

for all $(i,j)\in G$ and all x_i.

The proof of these two lemmas is a straightforward exercise in applying the definitions. Wainwright et al. (2004) provide similar results for the special case of the max-product algorithm.

2.2.1 COMPUTATION TREES

The computation trees produced by Algorithm 1 are different from their predecessors. Again, the computation tree captures the messages that would need to be passed in order to compute $\tau_i^t(x_i)$. However, the messages that are passed in the new algorithm are multiplied by a non-zero constant. As a result, the potential at a node u in the computation tree corresponds to some potential in the original graph multiplied by a constant that depends on all of the nodes above u in the computation tree. We summarize the changes as follows.

1. The message passed from i to j may now depend on the message from j to i at the previous time step. As such, we now form the time $t+1$ computation tree from the time t computation tree by taking any leaf u, which is a copy of node v in the factor graph, of the time t computation tree, creating a new node for every $w\in\partial v$, and connecting u to these new nodes. As a result, the new computation tree rooted at node u of depth t contains at least all of the non-backtracking walks of length t in the factor graph starting from u and, at most, all walks of length t in the factor graph starting at u.

2. The messages are weighted by the elements of c. This changes the potentials at the nodes in the computation tree. For example, suppose the computation tree was rooted at variable node i and that τ_i depends on the message from j to i. Because m_{ji} is multiplied by c_{ij} in τ_i,

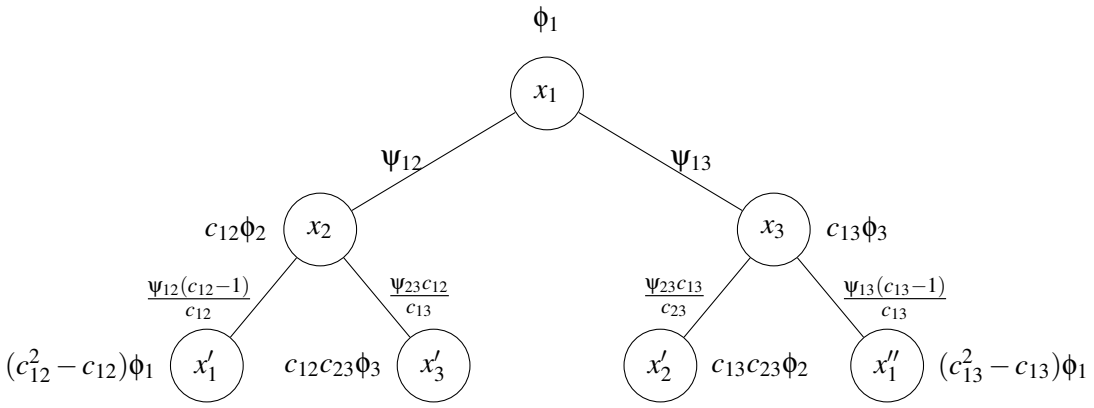


Figure 3: Construction of the computation tree rooted at node x_1 at time $t = 2$ produced by Algorithm 1 for the factor graph in Figure 1. Self-potentials are adjacent to the variable node to which they correspond. One can check that setting $c_{ij} = 1$ for all $(i, j) \in E$ reduces the above computation tree to that of Figure 2.

every potential along this branch of the computation tree is multiplied by c_{ij}. To make this concrete, we can associate a weight to every edge of the computation tree that corresponds to the constant that multiplies the message passed across that edge. To compute the new potential at a variable node i in the computation tree, we now need to multiply the corresponding potential ϕ_i by each of the weights corresponding to the edges that appear along the path from i to the root of the computation tree. An analogous process can be used to compute the potentials on each of the edges. The computation tree produced by Algorithm 1 at time $t = 2$ for the factor graph in Figure 1 is pictured in Figure 3. Compare this with computation tree produced by the standard min-sum algorithm in Figure 2.

If we make these adjustments, then the belief, $\tau_i^t(x_i)$, at node i at time t is given by the min-marginal at the root of $T_i(t)$. In this way, the beliefs correspond to marginals at the root of these computation trees.

2.3 Quadratic Minimization

We now address the quadratic minimization problem in the context of the reweighted min-sum algorithm. Recall that given a matrix Γ the quadratic minimization problem is to find the vector x that minimizes $f(x) = \frac{1}{2}x^T\Gamma x - h^T x$. Without loss of generality, we can assume that the matrix Γ is symmetric as the quadratic function $\frac{1}{2}x^T\Gamma x - h^T x$ is equivalent to $\frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x - h^T x$ for any

$\Gamma \in \mathbb{R}^{n\times n}$.

$$\begin{aligned} f(x) &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)+\frac{1}{2}(\Gamma-\Gamma^T)\right]x - h^T x \\ &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x + \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma-\Gamma^T)\right]x - h^T x \\ &= \frac{1}{2}x^T\left[\frac{1}{2}(\Gamma+\Gamma^T)\right]x - h^T x. \end{aligned}$$

Every quadratic function admits a pairwise factorization

$$\begin{aligned} f(x_1,...,x_n) &= \frac{1}{2}x^T\Gamma x - h^T x \\ &= \sum_i [\frac{1}{2}\Gamma_{ii}x_i^2 - h_i x_i] + \sum_{i>j}\Gamma_{ij}x_i x_j, \end{aligned}$$

where $\Gamma \in \mathbb{R}^{n\times n}$ is a symmetric matrix. We note that we will abusively write min in the reweighted update equations even though the appropriate notion of minimization for the real numbers is inf.

We can explicitly compute the minimization required by the reweighted min-sum algorithm at each time step: the synchronous message update $m_{i\to j}^t(x_j)$ can be parameterized as a quadratic function of the form $\frac{1}{2}a_{i\to j}^t x_j^2 + b_{i\to j}^t x_j$. If we define

$$A_{i\backslash j}^t \triangleq \left[\Gamma_{ii} + \sum_{k\in\partial i} c_{ki}\cdot a_{k\to i}^{t-1}\right] - a_{j\to i}^{t-1}$$

and

$$B_{i\backslash j}^t \triangleq \left[h_i - \sum_{k\in\partial i} c_{ki}\cdot b_{k\to i}^{t-1}\right] - b_{j\to i}^{t-1},$$

then the updates at time t are given by

$$\begin{aligned} a_{i\to j}^t &:= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{A_{i\backslash j}^t}, \\ b_{i\to j}^t &:= \frac{B_{i\backslash j}^t\frac{\Gamma_{ij}}{c_{ij}}}{A_{i\backslash j}^t}. \end{aligned}$$

These updates are only valid when $A_{i\backslash j} > 0$. If this is not the case, then the minimization given in Algorithm 1 is not bounded from below, and we set $a_{i\to j}^t = -\infty$. For the initial messages, we set $a_{i\to j}^0 = b_{i\to j}^0 = 0$.

Suppose that the beliefs generated from a fixed point of Algorithm 1 are locally decodable to x^*. One can show that the gradient of f at x^* is always equal to zero. If the gradient of f at x^* is zero and Γ is positive definite, then x^* must be a global minimum of f. In other words, the min-sum algorithm always computes the correct minimizing assignment if it converges to locally decodable beliefs. This result was previously proven for the GaBP algorithm (Weiss and Freeman, 2001b) and the tree-reweighted algorithm (Wainwright et al., 2003a).

Theorem 5 *If Algorithm 1 converges to a collection of beliefs, τ, that are locally decodable to x^* for a quadratic function f, then x^* is a local minimum of f.*

Proof For completeness, we sketch the proof. By Lemmas 3 and 4, we have that,

$$\min_{x_j} \tau_{ij}(x_i, x_j) = \kappa + \tau_i(x_i)$$

for all $(i, j) \in G$ and

$$f(x_1, \ldots, x_{|V|}) = \kappa + \sum_{i \in V} \tau_i(x_i) + \sum_{(i,j) \in E} c_{ij} \Big[\tau_{ij}(x_i, x_j) - \tau_i(x_i) - \tau_j(x_j) \Big].$$

If τ is locally decodable to x^*, then for each $i \in V$, $\tau_i(x_i)$ must be a positive definite quadratic function that is minimized at x_i^*. Applying Lemma 4, we have that for each $(i, j) \in E$, τ_{ij} is also a positive definite quadratic function and τ_{ij} is minimized at (x_i^*, x_j^*). For each $i \in V$,

$$\frac{d}{dx_i} f(x_1, \ldots, x_{|V|}) \quad = \quad \frac{d}{dx_i} \tau_i(x_i) + \sum_{j \in \partial i} c_{ij} \Big[\frac{d}{dx_i} \tau_{ij}(x_i, x_j) - \frac{d}{dx_i} \tau_i(x_i) \Big].$$

By the above arguments, for each $i \in V$, $\frac{d}{dx_i} \tau_i(x_i) \Big|_{x^*} = 0$. Similarly, for all $(i, j) \in E$, $\frac{d}{dx_i} \tau_{ij}(x_i, x_j) \Big|_{x^*} = 0$. As a result, we must have $\nabla f(x_1^*, \ldots, x_{|V|}^*) = 0$. If Γ is positive semidefinite, then f is convex and x^* must be a global minimum of f. ■

As a consequence of Theorem 5, even if Γ not positive definite, if some fixed point of the reweighted algorithm is locally decodable to a vector x^* then, x^* solves the system $\Gamma x = h$.

As discussed in the introduction, the GaBP algorithm is known to converge under certain conditions on the matrix Γ. Consider the following definitions.

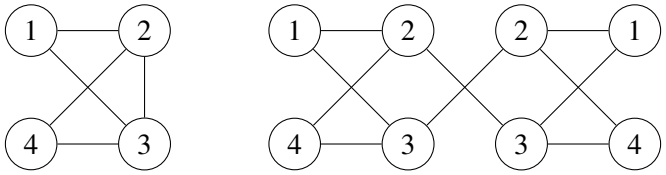
Definition 6 $\Gamma \in \mathbb{R}^{n \times n}$ *is* ***scaled diagonally dominant*** *if* $\exists w > 0 \in \mathbb{R}^n$ *such that* $|\Gamma_{ii}| w_i > \sum_{j \neq i} |\Gamma_{ij}| w_j$.

Definition 7 $\Gamma \in \mathbb{R}^{n \times n}$ *is* ***walk-summable*** *if the spectral radius* $\rho(|I - D^{-1/2} \Gamma D^{-1/2}|) < 1$. *Here,* $D^{-1/2}$ *is the diagonal matrix such that* $D_{ii}^{-1/2} = \frac{1}{\sqrt{\Gamma_{ii}}}$, *and* $|A|$ *denotes the matrix obtained from the matrix* A *by taking the absolute value of each entry of* A.

For any matrix Γ with strictly positive diagonal, Weiss and Freeman (2001b) demonstrated that GaBP converges when Γ is diagonally dominant (scaled diagonally dominant with all scale factors equal to one), Malioutov et al. (2006) proved that the GaBP algorithm converges when Γ is walk-summable, and Moallemi and Van Roy (2009, 2010) showed that GaBP converges when Γ is scaled diagonally dominant. We would like to understand how to choose the parameters of the reweighted algorithm in order to extend the convergence results for GaBP to all positive definite matrices.

3. Graph Covers

In this section, we will explore graph covers and their relationship to iterative message-passing algorithms for the quadratic minimization problem. Before addressing the quadratic minimization problem specifically, we will first make a few observations about general pairwise graphical models. The greatest strength of the above message-passing algorithms, their reliance on only local information, can also be a weakness: local message-passing algorithms are incapable of distinguishing two graphs that have the same local structure. To make this precise, we will need the notion of graph covers.



(a) A graph G (b) A 2-cover of G

Figure 4: An example of a graph cover. Nodes in the cover are labeled by the node that they are a copy of in G.

Definition 8 *A graph H **covers** a graph G if there exists a graph homomorphism $\pi : H \to G$ such that π is an isomorphism on neighborhoods (i.e., for all vertices $i \in H$, ∂i is mapped bijectively onto $\partial\pi(i)$). If $\pi(i) = j$, then we say that $i \in H$ is a copy of $j \in G$. Further, H is an M-cover of G if every vertex of G has exactly M copies in H.*

Graph covers, in the context of graphical models, were originally studied in relation to local message-passing algorithms for coding problems (Vontobel and Koetter, 2005). Graph covers may be connected (i.e., there is a path between every pair of vertices) or disconnected. However, when a graph cover is disconnected, all of the connected components of the cover must themselves be covers of the original graph. For a simple example of a connected graph cover, see Figure 4.

Every finite cover of a connected graph is an M-cover for some integer M. For every base graph G, there exists a graph, possibly infinite, which covers all finite, connected covers of the base graph. This graph is known as the universal cover.

To any finite cover, H, of a factor graph G we can associate a collection of potentials derived from the base graph; the potential at node $i \in H$ is equal to the potential at node $\pi(i) \in G$. Together, these potential functions define a new objective function for the factor graph H. In the sequel, we will use superscripts to specify that a particular object is over the factor graph H. For example, we will denote the objective function corresponding to a factor graph H as f^H, and we will write f^G for the objective function f.

Local message-passing algorithms such as the reweighted min-sum algorithm are incapable of distinguishing the two factor graphs H and G given that the initial messages to and from each node in H are identical to the nodes that they cover in G: for every node $i \in G$ the messages received and sent by this node at time t are exactly the same as the messages sent and received at time t by any copy of i in H. As a result, if we use a local message-passing algorithm to deduce an assignment for i, then the algorithm run on the graph H must deduce the same assignment for each copy of i.

Now, consider an objective function f that factors over the graph G. For any finite cover H of G with covering homomorphism $\pi : H \to G$, we can "lift" any vector of beliefs, τ^G, from G to H by defining a new vector of beliefs, τ^H, such that:

- For all variable nodes $i \in H$, $\tau^H_i = \tau^G_{\pi(i)}$.
- For all edges $(i, j) \in H$, $\tau^H_{ij} = \tau^G_{\pi(i)\pi(j)}$.

Analogously, we can lift any assignment x^G to an assignment x^H by setting $x^H_i = x^G_{\pi(i)}$.

3.1 Graph Covers and Quadratic Minimization

Let G be the pairwise factor graph for the objective function $f^G(x_1,...,x_n) = \frac{1}{2}x^T\Gamma x - h^T x$ whose edges correspond to the nonzero entries of Γ. Let H be an M-cover of G with corresponding objective function $f^H(x_{11},...,x_{1M},...x_{nM}) = \frac{1}{2}x^T\widetilde{\Gamma}x - \widetilde{h}^T x$. Without loss of generality we can assume that $\widetilde{\Gamma}$ and $\widetilde{h}$ take the following form:

$$\begin{aligned} \widetilde{\Gamma} &= \begin{pmatrix} \Gamma_{11}P_{11} & \cdots & \Gamma_{1n}P_{1n} \\ \vdots & \ddots & \vdots \\ \Gamma_{n1}P_{n1} & \cdots & \Gamma_{nn}P_{nn} \end{pmatrix}, \\ \widetilde{h}_i &= h_{\lceil i/M \rceil}, \end{aligned} \tag{1}$$

where $P_{ij} = P_{ji}^T$ is an $M \times M$ permutation matrix for all $i \neq j$ and P_{ii} is the $M \times M$ identity matrix for all i. If $\widetilde{\Gamma}$ is derived from Γ in this way, then we will say that $\widetilde{\Gamma}$ covers Γ.

For the quadratic minimization problem, factor graphs and their covers share many of the same properties. Most notably, we can transform critical points of covers to critical points of the original problem. Let H and G be as above, and let π be the graph homomorphism from H to G. For $x \in \mathbb{R}^{|V_G|}$, define $\text{lift}_H : \mathbb{R}^{|V_G|} \to \mathbb{R}^{|V_H|}$ such that

$$\text{lift}_H(x)_i = x_{\pi(i)}$$

for all $i \in H$. Similarly, for each $y \in \mathbb{R}^{|V_H|}$, define $\text{proj}_G \mathbb{R}^{|V_G|} \to \mathbb{R}^{|V_G|}$ such that

$$\text{proj}(y)_i = \sum_{k \in H : \pi(k) = i} \frac{y_k}{|\{j \in H : \pi(j) = i\}|}$$

for all $i \in G$. With these definitions, we have the following lemma.

Lemma 9 *If $\widetilde{\Gamma}y = \widetilde{h}$ for $y \in \mathbb{R}^{|V_H|}$, then $\Gamma \cdot proj_G(y) = h$. Conversely, if $\Gamma x = h$ for $y \in \mathbb{R}^{|V_G|}$, then $\widetilde{\Gamma} \cdot lift_H(x) = \widetilde{h}$.*

Notice that these solutions correspond to critical points of the cover and the original problem. Similarly, we can transform eigenvectors of covers to either eigenvectors of the original problem or the zero vector.

Lemma 10 *Fix $\lambda \in \mathbb{R}$. If $\widetilde{\Gamma}y = \lambda y$, then either $\Gamma \cdot proj_G(y) = \lambda proj_G(y)$ or $\Gamma \cdot proj_G(y) = 0$. Conversely, if $\Gamma x = \lambda x$, then $\widetilde{\Gamma} \cdot lift_H(x) = \lambda lift_H(x)$.*

These lemmas demonstrate that we can average critical points and eigenvectors of covers to obtain critical points and eigenvectors (or the zero vector) of the original problem, and we can lift critical points and eigenvectors of the original problem in order to obtain critical points and eigenvectors of covers.

Unfortunately, even though the critical points of G and its covers must correspond via Lemma 9, the corresponding minimization problems may not have the same solution. The example in Figure 5 illustrates that there exist positive definite matrices that are covered by matrices that are not positive definite. This observation seems to be problematic for the convergence of iterative message-passing schemes. Specifically, the fixed points of the reweighted algorithm on the base graph are also fixed

$$\Gamma = \begin{pmatrix} 1 & .6 & .6 \\ .6 & 1 & .6 \\ .6 & .6 & 1 \end{pmatrix} \quad \widetilde{\Gamma} = \begin{pmatrix} 1 & 0 & .6 & 0 & 0 & .6 \\ 0 & 1 & 0 & .6 & .6 & 0 \\ .6 & 0 & 1 & 0 & .6 & 0 \\ 0 & .6 & 0 & 1 & 0 & .6 \\ 0 & .6 & .6 & 0 & 1 & 0 \\ .6 & 0 & 0 & .6 & 0 & 1 \end{pmatrix}$$

Figure 5: An example of a positive definite matrix, Γ, which possesses a 2-cover, $\widetilde{\Gamma}$, that has negative eigenvalues.

points of the reweighted algorithm on any graph cover. As such, the reweighted algorithm may not converge to the correct minimizing assignment when the matrix corresponding to some cover of G is not positive definite. Consequently, we will first consider the special case in which Γ and all of its covers are positive definite. We can exactly characterize the matrices for which this property holds.

Theorem 11 *Let Γ be a symmetric matrix with positive diagonal. The following are equivalent.*

1. *Γ is walk-summable.*
2. *Γ is scaled diagonally dominant.*
3. *All covers of Γ are positive definite.*
4. *All 2-covers of Γ are positive definite.*

Proof The two non-trivial implications in the proof ($4 \Rightarrow 1$ and $1 \Rightarrow 2$) make use of the Perron-Frobenius theorem. For the complete details, see Appendix A. ■

This theorem has several important consequences. First, it provides us with a combinatorial characterization of scaled diagonal dominance and walk-summability. Second, it provides an intuitive explanation for why these conditions should be sufficient for the convergence of local message-passing algorithms.

More importantly, we can use Theorem 11 to conclude that MPLP, tree-reweighted max-product, and other message-passing algorithms that guarantee the correctness of locally decodable beliefs cannot converge to the correct solution when Γ is positive definite but not walk-summable. From the discussion in Section 3, every collection of locally decodable beliefs on the base graph can be lifted to locally decodable beliefs on any graph cover. Each of these "convergent and correct" message-passing algorithms guarantees that the lift of x^* to each graph cover must be a global minimum on that cover. By Theorem 11, there exists at least one graph cover with no global minimum. As a result, these algorithms cannot converge to locally decodable beliefs.

As we saw in Theorem 5, the reweighted message-passing algorithm only guarantees that x^* is a local optimum. However, there exist simple choices for the reweighting parameters that guarantee correctness over all covers. As an example, if $c_{ij} \leq \frac{1}{\max_{i \in V} |\partial i|}$ for all $(i, j) \in E$, then one can show that the reweighted algorithm cannot converge to locally decodable beliefs unless all of the graph covers are convex. The traditional choice of parameters for the TRMP algorithm where each c_{ij}

corresponds to an edge appearance probability provides another example. Given this observation, in order to produce convergent message-passing schemes for the quadratic minimization problem, we will need to study choices of the parameters that do not guarantee correctness over all graph covers.

4. Convergence Properties of Reweighted Message-Passing Algorithms

Recall that the GaBP algorithm can converge to the correct minimizer of the objective function even if the original matrix is not scaled diagonally dominant. The most significant problem when the original matrix is positive definite but not scaled diagonally dominant is that the computation trees may eventually possess negative eigenvalues due to the existence of some 2-cover with at least one non-positive eigenvalue. If this happens, then some of the beliefs will not be bounded from below, and the corresponding estimate will be negative infinity. This is, of course, the correct answer on some 2-cover of the problem, but it is not the correct solution to the minimization problem of interest. Our goal in this section is to understand how the choice of the parameters and alternative message-passing orders affect the convergence of the reweighted algorithm.

4.1 Convergence of the Variances

First, we will provide conditions on the choice of the parameter vector such that all of the computation trees produced by the reweighted algorithm remain positive definite throughout the course of the algorithm. Positive definiteness of the computation trees corresponds to the convexity of the beliefs, and the convexity of the belief, τ_i^t, is determined only by the vector a^t. As such, we begin by studying the sequence $a^0, a^1, \ldots$ where a^0 is the zero vector (based on our initialization). We will consider two different choices for the parameter vector: one in which $c_{ij} \geq 1$ for all i and j and one in which $c_{ij} < 0$ for all i and j. The latter of these two requires all of the parameters to be negative. Such a choice is unusual among reweighted algorithms, but it does guarantee that the computation trees will remain positive definite throughout the algorithm. This observation and the experimental results in Section 4.4 suggest that such a choice might be worth studying in other contexts as well.

4.1.1 POSITIVE PARAMETERS

Lemma 12 *If $c_{ij} \geq 1$ for all i and j, then for all $t > 0$, $a^t_{i\to j} \leq a^{t-1}_{i\to j} \leq 0$ for each i and j.*

Proof This result follows by induction on t. First, suppose that $c_{ij} \geq 1$. If the update is not valid, then $a^t_{i\to j} = -\infty$ which trivially satisfies the inequality. Otherwise, we have

$$\begin{aligned} a^t_{i\to j} &= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i}} \\ &\leq \frac{\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-2}_{k\to i} + (c_{ji}-1)a^{t-2}_{j\to i}} \\ &= a^{t-1}_{i\to j}, \end{aligned}$$

where the inequality follows from the observation that $\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i} > 0$ and the induction hypothesis. ■

$$\begin{pmatrix} 1 & 0.39866 & -0.39866 & -0.39866 \\ 0.39866 & 1 & -0.39866 & 0 \\ -0.39866 & -0.39866 & 1 & -0.39866 \\ -0.39866 & 0 & -0.39866 & 1 \end{pmatrix}$$

Figure 6: A positive definite matrix for which the variances in the min-sum algorithm converge but the means do not (Malioutov, 2008).

If we consider only the vector a^t, then the algorithm may exhibit a weaker form of convergence.

Lemma 13 *If $c_{ij} \geq 1$ for all i and j and all of the computation trees are positive definite, then the sequence $a^0_{i\to j}, a^1_{i\to j}, \ldots$ converges.*

Proof Suppose $c_{ij} \geq 1$. By Lemma 12, the $a^t_{i\to j}$ are monotonically decreasing. Because all of the computation trees are positive definite, we must have that for each i, $\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + c_{ji} a^{t-1}_{j\to i} > 0$. Therefore, for all $(i,j) \in E$, $a^t_{i\to j} \geq -\frac{\Gamma_{ii}}{c_{ij}}$. Consequently, the sequence $a^0_{i\to j}, a^1_{i\to j}, \ldots$ is monotonically decreasing and bounded from below. This implies that the sequence converges. ■

Because the estimates of the variances only depend on the vector a^t, if the $a^t_{i\to j}$ converge, then the estimates of the variances also converge. Therefore, requiring all of the computation trees to be positive definite is a sufficient condition for convergence of the variances. Note, however, that the estimates of the means which correspond to the sequence $b^t_{i\to j}$ need not converge even if all of the computation trees are positive definite (see Figure 6).

Our strategy will be to ensure that all of the computation trees are positive definite by leveraging the choice of parameters, c_{ij}. Specifically, we want to use these parameters to weight the diagonal elements of the computation tree much more than the off-diagonal elements in order to force the computation trees to be positive definite. If we can show that there is a choice of each $c_{ij} = c_{ji}$ that will cause all of the computation trees to be positive definite, then Algorithm 1 should behave almost as if the original matrix were scaled diagonally dominant. Indeed, there always exists a choice of the vector c that achieves this.

Theorem 14 *For any symmetric matrix Γ with strictly positive diagonal, $\exists r \geq 1$ and an $\varepsilon > 0$ such that the eigenvalues of the computation trees are bounded from below by ε when generated by Algorithm 1 with $c_{ij} = r$ for all i and j.*

The proof of this theorem exploits the Geršgorin disc theorem in order to show that there exists a choice of r such that each computation tree is scaled diagonally dominant. The complete proof can be found in Appendix B.

4.1.2 NEGATIVE PARAMETERS

For the case in which $c_{ij} < 0$ for all i and j, we also have that the computation trees are always positive definite when the initial messages are uniformly equal to zero as characterized by the following lemmas.

Lemma 15 *If $c_{ij} < 0$ for all i and j, then for all $t > 0$, $a^t_{i\to j} \leq 0$.*

Proof This result follows by induction on t. First, suppose that $c_{ij} < 0$ for all $(i,j) \in E$. If the update is not valid, then $a^t_{i\to j} = -\infty$ which trivially satisfies the inequality. Otherwise, we have

$$\begin{aligned} a^t_{i\to j} &= \frac{-\left(\frac{\Gamma_{ij}}{c_{ij}}\right)^2}{\Gamma_{ii} + \sum_{k\in\partial i\setminus j} c_{ki} a^{t-1}_{k\to i} + (c_{ji}-1)a^{t-1}_{j\to i}} \\ &\leq 0, \end{aligned}$$

where the inequality follows from the induction hypothesis. ■

Lemma 16 *For any symmetric matrix Γ with strictly positive diagonal, if $c_{ij} < 0$ for all i and j, then all of the computation trees are positive definite.*

Proof The computation trees are all positive definite if and only if $\Gamma_{ii} + \sum_{k\in\partial i} c_{ki} a^t_{k\to i} > 0$ for all t. By Lemma 15, $a^t_{i\to j} \leq 0$ for all t, and as result, $\Gamma_{ii} + \sum_{k\in\partial i} c_{ki} a^t_{k\to i} \geq \Gamma_{ii} > 0$ for all t. ■

As when $c_{ij} \geq 1$ for all $(i,j) \in E$, the eigenvalues on each computation tree are again bounded way from zero, but the $a^t_{i\to j}$ no longer form a monotonic decreasing sequence when $c_{ij} < 0$ for all $(i,j) \in E$. If all of the computation trees remain positive definite in the limit, then the beliefs will all be positive definite upon convergence. If the estimates for the means converge as well, then the converged beliefs must be locally decodable to the correct minimizing assignment. Notice that none of the above arguments for the variances require Γ to be positive definite. Indeed, we have already seen an example of a matrix with a strictly positive diagonal and negative eigenvalues (see the matrix in Figure 5) such that the variance estimates converge.

4.2 Alternative Message Passing Schedules

The synchronous message-passing updates described in Algorithm 1 enforce a particular ordering on the updates performed at each time step. In practice, alternative message-passing schedules may improve the rate of convergence. One such alternative message-passing schedule is given by Algorithm 2. Because each computation tree produced by this algorithm is a principal submatrix of a synchronous computation tree and principal submatrices of positive definite matrices are positive definite, we can easily check that all of the results of the previous section extend to this modified schedule as well.

Algorithm 2 allows for quite a bit more flexibility in the scheduling of message updates, and as we will see experimentally in Section 4.4, it can have better convergence properties than the corresponding synchronous algorithm. To see why this might be the case, we will again exploit the properties of graph covers. Specifically, we will show that these two algorithms are related via a special 2-cover of the base factor graph.

Every pairwise factor graph, $G = (V_G, E_G)$, admits a bipartite 2-cover, $H = (V_G \times \{1,2\}, E_H)$, called the Kronecker double cover of G. We will denote copies of the variable x_i in this 2-cover as x_{i_1} and x_{i_2}. For every edge $(i,j) \in E_G$, (i_1, j_2) and (i_2, j_1) belong to E_H. In this way, nodes labeled with a one are only connected to nodes labeled with a two (see Figure 7). Note that if G is already a bipartite graph, then the Kronecker double cover of G is simply two disjoint copies of G.

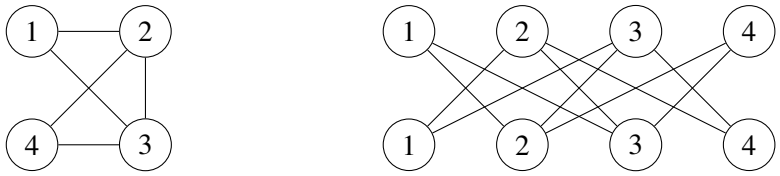
We can view the synchronous algorithm described in Algorithm 1 as a specific message-update schedule of messages on the Kronecker double cover where we perform the update in Algorithm 2 for every variable in the same partition on alternating iterations (see Algorithm 3).

Algorithm 2 Alternative Reweighted Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: Choose some ordering of the variables such that each variable is updated infinitely often, and perform the following update for each variable j in order
3: **for** each $i \in \partial j$ **do**
4: Update the message from i to j:

$$m_{i\to j}(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + (c_{ij}-1)m_{j\to i}(x_i) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}(x_i)\right].$$

5: **end for**



(a) A pairwise factor graph G (b) Bipartite 2-cover of G

Figure 7: The Kronecker double cover (b) of a pairwise factor graph (a). The node labeled $i \in G$ corresponds to the variable node x_i.

Algorithm 3 Bipartite Message-Passing Algorithm

1: Initialize the messages to some finite vector.
2: Iterate the following until convergence: update all of the outgoing messages from nodes labeled one to nodes labeled two and then update all of the outgoing messages from nodes labeled two to nodes labeled one using the asynchronous update rule:

$$m_{i\to j}(x_j) := \kappa + \min_{x_i}\left[\frac{\psi_{ij}(x_i,x_j)}{c_{ij}} + (c_{ij}-1)m_{j\to i}(x_i) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} c_{ki}m_{k\to i}(x_i)\right].$$

By construction, the message vector produced by Algorithm 3 is simply a concatenation of two consecutive time steps of the synchronous algorithm. Specifically, for all $t \geq 1$

$$m_H^t = \begin{bmatrix} m_G^{2t-1} \\ m_G^{2t-2} \end{bmatrix}.$$

Therefore, the messages passed by Algorithm 1 are identical to those passed by a specific ordering of the updates in Algorithm 2 on the Kronecker double cover. From our earlier analysis, we know that even if Γ is positive definite, not every cover necessarily corresponds to a convex objective function. If the Kronecker double cover is such a "bad" cover, then we might expect that the synchronous reweighted algorithm may not converge to the correct solution. This reasoning is not unique to the iterative message-passing algorithms described above. In the next section, we will see that it can also be applied to other iterative techniques for quadratic minimization.

Algorithm 4 Jacobi Iteration

1: Choose an initial vector $x^0 \in \mathbb{R}^n$.
2: For iteration $t = 1, 2, \ldots$ set

$$x_j^t = \frac{h_j - \sum_k \Gamma_{jk} x_k^{t-1}}{\Gamma_{jj}}$$

for each $j \in \{1, \ldots, n\}$.

Algorithm 5 Gauss-Seidel Iteration

1: Choose an initial vector $x \in \mathbb{R}^n$.
2: Choose some ordering of the variables, and perform the following update for each variable j, in order,

$$x_j = \frac{h_j - \sum_k \Gamma_{jk} x_k}{\Gamma_{jj}}.$$

4.2.1 THE GAUSS-SEIDEL AND JACOBI METHODS

Because minimizing symmetric positive definite quadratic functions is equivalent to solving symmetric positive definite linear systems, well-studied algorithms such as Gaussian elimination, Cholesky decomposition, etc. can be used to compute the minimum. In addition, many iterative algorithms have been proposed to solve the linear system $\Gamma x = h$: Gauss-Seidel iteration, Jacobi iteration, the algebraic reconstruction technique, etc.

In this section, we will show that the previous graph cover analysis can also be used to reason about the Jacobi and Gauss-Seidel algorithms (Algorithms 4 and 5). In Section 4.3, we will see that there is an even deeper connection between these algorithms and reweighted message-passing algorithms. When Γ is symmetric positive definite, the objective function, $\frac{1}{2}x^T\Gamma x - h^T x$, is a convex function of x. Consequently, we could use a coordinate descent scheme in an attempt to minimize the objective function. The standard cyclic coordinate descent algorithm for this problem is known as the Gauss-Seidel algorithm.

In the same way that Algorithm 1 is a synchronous version of Algorithm 2, the Jacobi algorithm is a synchronous version of the Gauss-Seidel algorithm. To see this, observe that the iterates produced by the Jacobi algorithm are related to the iterates of the Gauss-Seidel algorithm on a larger problem. Specifically, given a symmetric $\Gamma \in \mathbb{R}^{n\times n}$ and $h \in \mathbb{R}^n$, construct $\Gamma' \in \mathbb{R}^{2n\times 2n}$ and $h' \in \mathbb{R}^{2n}$ as follows

$$\begin{aligned} h_i' &= \begin{bmatrix} h \\ h \end{bmatrix}, \\ \Gamma' &= \begin{bmatrix} D & M \\ M & D \end{bmatrix}, \end{aligned}$$

where D is a diagonal matrix with the same diagonal entries as Γ and $M = \Gamma - D$.

Γ' is the analog of the Kronecker double cover discussed in Section 4.2. Let $x^0 \in \mathbb{R}^n$ be an initial vector for the Jacobi algorithm performed on the matrix Γ and fix $y^0 \in \mathbb{R}^{2n}$ such that $y^0 = \begin{bmatrix} x^0 \\ x^0 \end{bmatrix}$. Further, suppose that we update the variables in the order $1, 2, \ldots, 2n$ in the Gauss-Seidel algorithm.

If y^t is the vector produced after t complete cycles of the Gauss-Seidel algorithm, then $y^t = \begin{bmatrix} x^{2t-1} \\ x^{2t} \end{bmatrix}$. Also, observe that, for any y^t such that $\Gamma' y^t = h'$, we must have that $\Gamma \left[\frac{x^{2t-1}+x^{2t}}{2} \right] = h$.

With these two observations, any convergence result for the Gauss-Seidel algorithm can be extended to the Jacobi algorithm.

Theorem 17 *Let Γ be a symmetric positive semidefinite matrix with a strictly positive diagonal. The Gauss-Seidel algorithm converges to a vector x^* such that $\Gamma x^* = h$ whenever such a vector exists.*

Proof See Section 10.5.1 of Byrne (2008). ■

Using our observations, we can immediately produce the following new result.

Corollary 18 *Let Γ be a symmetric positive semidefinite matrix with positive diagonal and let Γ' be constructed as above. If Γ' is a symmetric positive semidefinite matrix and there exists an x^* such that $\Gamma x^* = h$, then the sequence $\frac{x^t + x^{t-1}}{2}$ converges to x^* where x^t is the t^{th} iterate of the Jacobi algorithm.*

If Γ' is not positive semidefinite, then the Gauss-Seidel algorithm (and by extension the Jacobi algorithm) may or may not converge when run on Γ'.

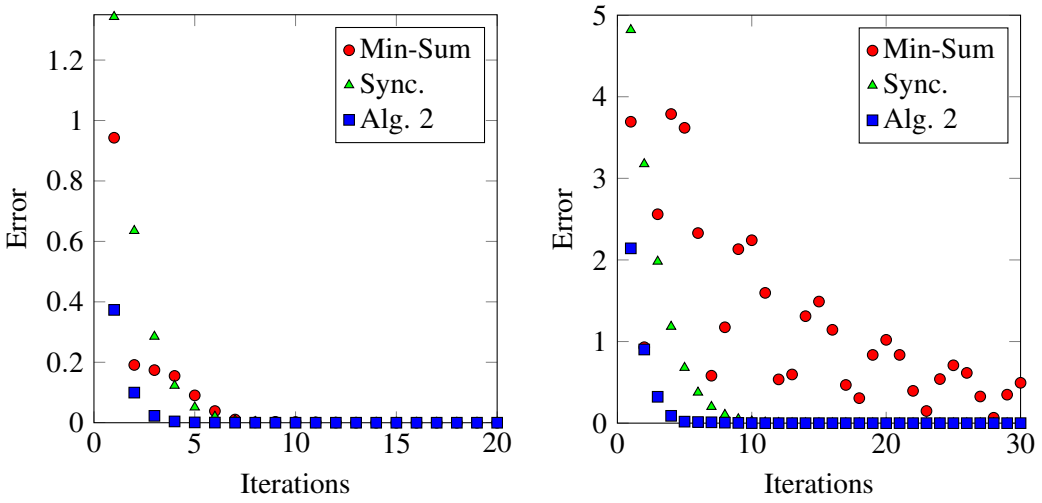
4.3 Convergence of the Means

If the variances converge, then the fixed points of the message updates for the means correspond to the solution of a particular linear system $Mb = d$. In fact, we can show that Algorithm 2 is exactly the Gauss-Seidel algorithm for this linear system. First, we construct the matrix $M \in \mathbb{R}^{2|E| \times 2|E|}$:

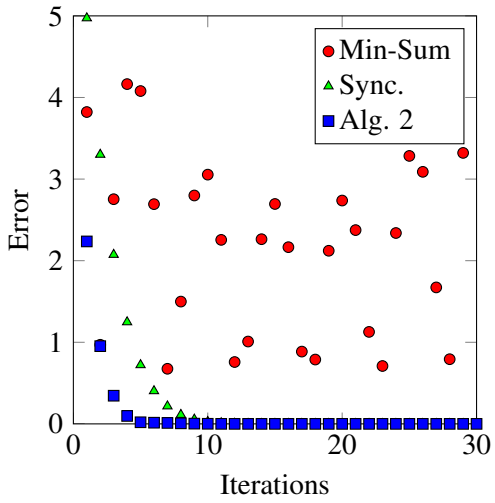
$$\begin{aligned} M_{ij,ij} &= A^*_{i \setminus j} \text{ for all } i \in V \text{ and } j \in \partial i, \\ M_{ij,ki} &= c_{ki} \frac{\Gamma_{ij}}{c_{ij}} \text{ for all } i \in V \text{ and for all } j,k \in \partial i \text{ such that } k \neq j, \\ M_{ij,ji} &= (c_{ij} - 1) \frac{\Gamma_{ij}}{c_{ij}} \text{ for all } i \in V \text{ and } j \in \partial i. \end{aligned}$$

Here, A^* is constructed from the vector of converged variances, a^*. All other entries of the matrix are equal to zero. Next, we define the vector $d \in \mathbb{R}^{2|E|}$ by setting $d_{ij} = h_i \Gamma_{ij} / c_{ij}$ for all $i \in V$ and $j \in \partial i$.

By definition, any fixed point, b^*, of the message update equations for the means must satisfy $Mb^* = d$. With these definitions, Algorithm 2 is precisely the Gauss-Seidel algorithm for this matrix. Similarly, Algorithm 1 corresponds to the Jacobi algorithm. Unfortunately, M is neither symmetric nor diagonally dominant, so the standard results for the convergence of the Gauss-Seidel algorithm do not necessarily apply to this situation. In practice, we have observed that Algorithm 2 converges for positive definite matrices if each c_{ij} is sufficiently large (or sufficiently negative).



(a) $p = .3$ (b) $p = .398$



(c) $p = .4$

Figure 8: The error, measured by the 2-norm, between the current mean estimate and the true mean at each step of the min-sum algorithm, the alternative message-passing schedule in Algorithm 2 with $c_{ij} = 2$ for all $i \neq j$, and the synchronous algorithm with $c_{ij} = 2$ for all $i \neq j$ for the matrix in (2). Notice that all of the algorithms have a similar performance when p is chosen such that the matrix is scaled diagonally dominant. When the matrix is not scaled diagonally dominant, the min-sum algorithm converges more slowly or does not converge at all.

4.4 Experimental Results

Even simple experiments demonstrate the advantages of the reweighted message-passing algorithm compared to the typical min-sum algorithm. Throughout this section, we will assume that h is

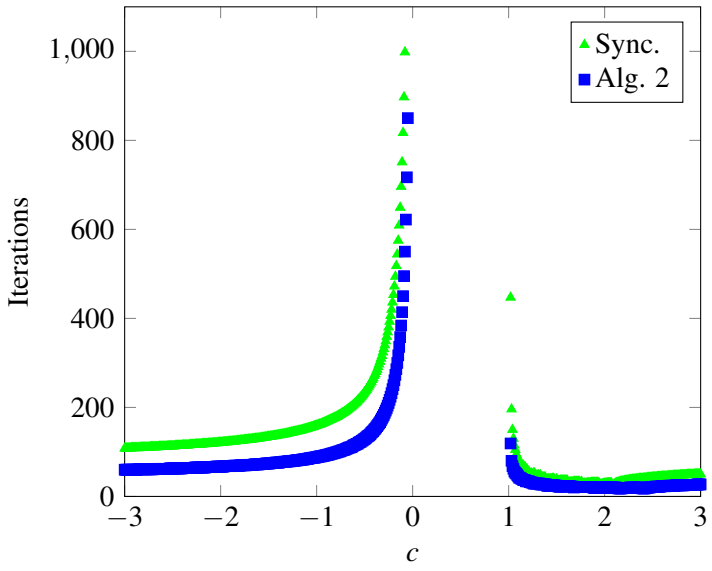


Figure 9: The number of iterations needed to reduce the error of the mean estimates below 10^{-6} using the reweighted algorithms as a function of c for the matrix in (2) with $p = .4$. The gap in the plot is predicted by the arguments at the end of Section 3.1.

chosen to be the vector of all ones. Let Γ be the following matrix.

$$\begin{pmatrix} 1 & p & -p & -p \\ p & 1 & -p & 0 \\ -p & -p & 1 & -p \\ -p & 0 & -p & 1 \end{pmatrix}. \tag{2}$$

The standard min-sum algorithm converges to the correct solution for $0 \leq p < .39865$ (Malioutov et al., 2006). Figure 8 illustrates the behavior of the min-sum algorithm, Algorithm 2 with $c_{ij} = 2$ for all $i \neq j$, and the synchronous algorithm with $c_{ij} = 2$ for all $i \neq j$ for different choices of the constant p. Each iteration of Algorithm 2 algorithm consists of cyclically updating the incoming messages to all nodes. In the examples in Figure 8, the synchronous algorithm and Algorithm 2 always converge rapidly to the correct mean while the min-sum algorithm converges slowly or not at all as p approaches .5.

While this is a simple graph, the behavior of the algorithm for different choices of the vector c is already apparent. If we set $c_{ij} = 3$ for all $i \neq j$, then empirically, both the synchronous algorithm and Algorithm 2 converge for all $p \in (-.5, .5)$, the entire positive definite region for this matrix. However, different choices of the parameter vector can greatly increase or decrease the number of iterations required for convergence. Figure 9 illustrates the iterations to convergence for the reweighted algorithms at $p = .4$ versus c.

Although both the synchronous algorithm and Algorithm 2 converge for the entire positive definite region in the above example, they can have very different convergence properties and damping may be required in order to force the synchronous algorithm to converge over arbitrary graphs, even

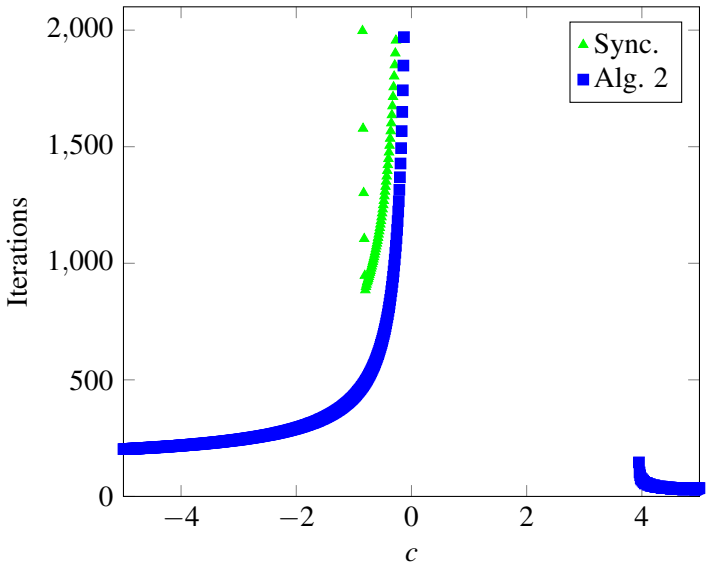


Figure 10: The number of iterations needed to reduce the error of the mean estimates below 10^{-6} using the reweighted algorithms as a function of c for the matrix in (3). Again, the gap in the plot is predicted by the arguments at the end of Section 3.1.

for sufficiently large c. Figure 10 illustrates these convergence issues for the matrix,

$$\begin{pmatrix} 45 & 21 & 23 & -42 \\ 21 & 83 & 8 & -32 \\ 23 & 8 & 14 & -29 \\ -42 & -32 & -29 & 134 \end{pmatrix}. \tag{3}$$

The above matrix was randomly generated. Similar observations can be made for many other positive definite matrices as well.

5. Conclusions and Future Research

In this work, we explored the properties of reweighted message-passing algorithms for the quadratic minimization problem. Our motivation was to address the convergence issues in the GaBP algorithm by leveraging the reweighting parameters. To this end, we employed graph covers to prove that standard approaches to convergence and correctness that exploit duality and coordinate ascent/descent such as MPLP (Globerson and Jaakkola, 2007), tree-reweighted max-product (Wainwright et al., 2003b), and Sontag and Jaakkola (2009) are doomed to fail outside of walk-summable models. While the GaBP variances may not converge outside of walk-summable matrices, we showed that there always exists a choice of reweighting parameters that guarantees monotone convergence of the variances. Empirically, a similar strategy seems to guarantee convergence of the means for positive definite matrices as well. As a result, our approach demonstrably outperforms the GaBP algorithm on this problem. We conclude this work with a discussion of a few open problems and directions for future research.

5.1 Convergence

The main open questions surrounding the performance of the reweighted algorithm relate to questions of convergence. First, for all positive definite Γ, we conjecture that there exists a sufficiently large (or sufficiently negative) choice of the parameters such that the means always converge under Algorithm 2.

Second, in practice, one typically uses a damped version of the message updates in order to attempt to force convergence. For the min-sum algorithm, the damped updates are given by

$$m_{i\to j}^t(x_j) \quad = \quad \kappa + \delta m_{i\to j}^t(x_j) + (1-\delta)\Big[\min_{x_i} \psi_{ij}(x_i,x_j) + \phi_i(x_i) + \sum_{k\in\partial i\setminus j} m_{k\to i}^{t-1}(x_i)\Big]$$

for some $\delta \in [0,1)$. The damped min-sum algorithm with damping factor $\delta = 1/2$ empirically seems to converge if Γ is positive definite and all of the computation trees remain positive definite (Malioutov et al., 2006). We make the same observation for the damped version of Algorithm 1.

In practice, the damped synchronous algorithm with $\delta = 1/2$ and Algorithm 2 appear to converge for all sufficiently large choices of the parameter vector as long as Γ is positive definite. We conjecture that this is indeed the case: for all positive definite Γ there exists a c such that if $c_{ij} = c$ for all $i \neq j$, then the damped algorithm and Algorithm 2 always converge. In this line of exploration, the relationship between the synchronous algorithm and Algorithm 2 described in Section 4.2 may be helpful.

Finally, Moallemi and Van Roy (2010) were able to provide rates of convergence in the case that Γ is walk-summable by using a careful analysis of the computation trees. Perhaps similar ideas could be adapted for the computation trees produced by the reweigthed algorithm.

5.2 General Convex Minimization

The reweighted algorithm can, in theory, be applied to minimize general convex functions, but in practice, computing and storing the message vector for these more general models may be inefficient. Despite this, many of the previous observations can be extended to the general case of a convex function $f : C \to \mathbb{R}$ such that $C \subseteq \mathbb{R}^n$ is a convex set.

As was the case for quadratic minimization, convexity of the objective function f^G does not necessarily guarantee convexity of the objective function f^H for every finite cover H of G. Recall that the existence of graph covers that are not bounded from below can be problematic for the reweighted message-passing algorithm. For quadratic functions, this cannot occur if the matrix is scaled diagonally dominant or, equivalently, if the objective function corresponding to every finite graph cover is positive definite. This equivalence suggests a generalization of scaled diagonal dominance for arbitrary convex functions based on the convexity of their graph covers. Such convex functions would have desirable properties with respect to iterative message-passing schemes.

Lemma 19 *Let f be a convex function that factorizes over a graph G. Suppose that for every finite cover H of G, f^H is convex. If $x^G \in \arg\min_x f(x)$, then for every finite cover H of G, x^H, the lift of x^G to H, minimizes f^H.*

Proof This follows from the observation that all convex functions are subdifferentiable over their domains and that x^H is a minimum of f^H if and only if the zero vector is contained in the subgradient of f^H at x^H. ■

Even if the objective function is not convex for some cover, we may still be able to use the same trick as in Theorem 14 in order to force the computation trees to be convex. Let $C \subseteq \mathbb{R}^n$ be a convex set. If $f : C \to \mathbb{R}$ is twice continuously differentiable, then f is convex if and only if its Hessian, the matrix of second partial derivatives, is positive semidefinite on the interior of C. For each fixed $x \in C$, Theorem 14 demonstrates that there exists a choice of the vector c such that all of the computation trees are convex at x, but it does not guarantee the existence of a c that is independent of x.

For twice continuously differentiable functions, Moallemi and Van Roy (2010) provide sufficient conditions for the convergence of the min-sum algorithm that are based on a generalization of scaled diagonal dominance, and extending the above ideas is the subject of future research.

Appendix A. Proof of Theorem 11

Without loss of generality, we can assume that Γ has a unit diagonal. We break the proof into several pieces:

- $(1 \Rightarrow 2)$ Without loss of generality we can assume that $|I-\Gamma|$ is irreducible (if not we can make this argument on each of its connected components). Let $1 > \lambda > 0$ be an eigenvalue of $|I-\Gamma|$ with eigenvector $x > 0$ whose existence is guaranteed by the Perron-Frobenius theorem. For any row i, we have:

  $$x_i > \lambda x_i = \sum_{j \neq i} |\Gamma_{ij}| x_j.$$

  Since $\Gamma_{ii} = 1$ this is the definition of scaled diagonal dominance with $w = x$.

- $(2 \Rightarrow 3)$ If Γ is scaled diagonally dominant then so is every one of its covers. Scaled diagonal dominance of a symmetric matrix with a positive diagonal implies that the matrix is symmetric positive definite. Therefore, all covers must be symmetric positive definite.

- $(3 \Rightarrow 4)$ Trivial.

- $(4 \Rightarrow 1)$ Let $\widetilde{\Gamma}$ be any 2-cover of Γ. Without loss of generality, we can assume that $\widetilde{\Gamma}$ has the form (1).

  First, observe that by the Perron-Frobenius theorem there exists an eigenvector $x > 0 \in \mathbb{R}^n$ of $|I-\Gamma|$ with eigenvalue $\rho(|I-\Gamma|)$. Let $y \in \mathbb{R}^{2n}$ be constructed by duplicating the values of x so that $y_{2i} = y_{2i+1} = x_i$ for each $i \in \{0...n\}$. By Lemma 10, y is an eigenvector of $|I-\widetilde{\Gamma}|$ with eigenvalue equal to $\rho(|I-\Gamma|)$. We claim that this implies $\rho(|I-\widetilde{\Gamma}|) = \rho(|I-\Gamma|)$. Assume without loss of generality that $|I-\widetilde{\Gamma}|$ is irreducible; if not, then we can apply the following argument to each connected component of $|I-\widetilde{\Gamma}|$. By the Perron-Frobenius theorem again, $|I-\widetilde{\Gamma}|$ has a unique positive eigenvector (up to scalar multiple), with eigenvalue equal to the spectral radius. Thus, $\rho(|I-\Gamma|) = \rho(|I-\widetilde{\Gamma}|)$ because $y > 0$.

  We will now construct a specific cover $\widetilde{\Gamma}$ such that $\widetilde{\Gamma}$ is positive definite if and only if Γ is walk-summable. To do this, we'll choose the P_{ij} as in (1) such that $P_{ij} = I$ if $\Gamma_{ij} < 0$ and $P_{ij} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ otherwise. Now define $z \in \mathbb{R}^{2n}$ by setting $z_i = (-1)^i c y_i$, where the constant c ensures that $\|z\| = 1$.

Consider the following:

$$\begin{aligned} z^T\widetilde{\Gamma}z &= \sum_{i=1}^{n}\sum_{j\neq i}\Gamma_{ij}[z_{2i},z_{2i+1}]P_{ij}\begin{bmatrix} z_{2j} \\ z_{2j+1}\end{bmatrix} + \sum_i \Gamma_{ii}z_i^2 \\ &= 1-2\sum_{i>j}|\Gamma_{ij}|c^2y_iy_j. \end{aligned}$$

Recall that y is the eigenvector of $|I-\widetilde{\Gamma}|$ corresponding to the largest eigenvalue and $\|cy\|=1$. By definition and the above,

$$\begin{aligned} \rho(|I-\Gamma|) &= \rho(|I-\widetilde{\Gamma}|) \\ &= \frac{cy^T|I-\widetilde{\Gamma}|cy}{c^2y^Ty} \\ &= 2\sum_{i>j}|\Gamma_{ij}|c^2y_iy_j. \end{aligned}$$

Combining all of the above we see that $z^T\widetilde{\Gamma}z = 1-\rho(|I-\Gamma|)$. Now, $\widetilde{\Gamma}$ positive definite implies that $z^T\widetilde{\Gamma}z > 0$, so $1-\rho(|I-\Gamma|) > 0$. In other words, Γ is walk-summable.

Appendix B. Proof of Theorem 14

Let $T_v(t)$ be the depth t computation tree rooted at v, and let Γ' be the matrix corresponding to $T_v(t)$ (i.e., the matrix generated by the potentials in the computation tree). We will show that the eigenvalues of Γ' are bounded from below by some $\varepsilon > 0$. For any $i \in T_v(t)$ at depth d define:

$$w_i = \left(\frac{s}{r}\right)^d,$$

where r is as in the statement of the theorem and s is a positive real to be determined below. Let W be a diagonal matrix whose entries are given by the vector w. By the Geršgorin disc theorem (Horn and Johnson, 1990), all of the eigenvalues of $W^{-1}\Gamma' W$ are contained in

$$\cup_{i\in T_v(t)}\left\{z\in\mathbb{R} : |z-\Gamma'_{ii}| \leq \frac{1}{w_i}\sum_{j\neq i} w_j|\Gamma'_{ij}|\right\}.$$

Because all of the eigenvalues are contained in these discs, we need to show that there is a choice of s and r such that for all $i \in T_v(t)$, $|\Gamma'_{ii}| - \frac{1}{w_i}\sum_{j\neq i} w_j|\Gamma'_{ij}| \geq \varepsilon$.

Recall from Section 2.2.1 that $|\Gamma'_{ij}| = \eta\frac{|\Gamma_{ij}|}{r}$ for some constant η that depends on r. Further, all potentials below the potential on the edge (i,j) are multiplied by $\eta\gamma$ for some constant γ. We can divide out by this common constant to obtain equations that depend on r and the elements of Γ. Note that some self-potentials will be multiplied by $r-1$ while others will be multiplied by r. With this rewriting, there are three possibilities:

1. i is a leaf of $T_v(t)$. In this case, we need $|\Gamma_{ii}| > \frac{1}{w_i}\frac{|\Gamma_{ip(i)}|}{r}w_{p(i)}$. Plugging in the definition of w_i, we have

$$|\Gamma_{ii}| > \frac{|\Gamma_{ip(i)}|}{s}. \tag{4}$$

2. i is not a leaf of $T_v(t)$ or the root. In this case, we need

$$|\Gamma_{ii}| \quad > \quad \frac{1}{w_i}\Big[\frac{|\Gamma_{ip(i)}|}{r}w_{p(i)} + \frac{s^2(r-1)}{r^3}|\Gamma_{ip(i)}|w_{p(i)} + \sum_{k\in\partial i - p(i)}|\Gamma_{ki}|w_k\Big].$$

Again, plugging the definition of w_i into the above yields

$$|\Gamma'_{ii}| \quad > \quad \frac{|\Gamma_{ip(i)}|}{s} + \frac{s}{r}\Big[\frac{r-1}{r}|\Gamma_{ip(i)}| + \sum_{k\in\partial i - p(i)}|\Gamma_{ki}|\Big].$$

3. i is the root of $T_v(t)$. Similar to the previous case, we need $|\Gamma_{ii}|w_i > \sum_{k\in\partial i}|\Gamma_{ki}|w_k$. Again, plugging the definition of w_i into the above yields

$$|\Gamma_{ii}| > \frac{s}{r}\sum_{k\in\partial i}|\Gamma_{ki}|.$$

None of these bounds are time dependent. As such, if we choose s and r to satisfy the above constraints, then there must exist some $\varepsilon > 0$ such that smallest eigenvalue of any computation tree is at least ε. Fix s to satisfy (4) for all leaves of $T_v(t)$. This implies that $(|\Gamma_{ii}| - \frac{|\Gamma_{ip(i)}|}{s}) > 0$ for any $i \in T_v(t)$. Finally, we can choose a sufficiently large r that satisfies the remaining two cases for all $i \in T_v(t)$.

References

C. L. Byrne. *Applied Iterative Methods*. A K Peters, Ltd., 2008.

B. J. Frey, R. Koetter, and A. Vardy. Signal-space characterization of iterative decoding. *Information Theory, IEEE Transactions on*, 47(2):766–781, Feb. 2001.

A. Globerson and T. S. Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Proc. Neural Information Processing Systems (NIPS)*, Vancouver, B. C., Canada, Dec. 2007.

T. Hazan and A. Shashua. Norm-product belief propagation: Primal-dual message-passing for approximate inference. *Information Theory, IEEE Transactions on*, 56(12):6294 –6316, Dec. 2010.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

J. K. Johnson, D. Bickson, and D. Dolev. Fixing convergence of Gaussian belief propagation. In *Proc. Information Theory, IEEE International Symposium on (ISIT)*, pages 1674–1678, Seoul, South Korea, July 2009.

D. M. Malioutov. Approximate inference in Gaussian graphical models. Ph.D. thesis, EECS, MIT, 2008.

D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-sums and belief propagation in Gaussian graphical models. *Journal of Machine Learning Research (JMLR)*, 7:2031–2064, 2006.

T. Meltzer, A. Globerson, and Y. Weiss. Convergent message passing algorithms: a unifying view. In *Proc. of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, Montreal, Canada, June 2009.

C. C. Moallemi and B. Van Roy. Convergence of min-sum message passing for quadratic optimization. *Information Theory, IEEE Transactions on*, 55(5):2413 –2423, May 2009.

C. C. Moallemi and B. Van Roy. Convergence of min-sum message-passing for convex optimization. *Information Theory, IEEE Transactions on*, 56(4):2041 –2050, April 2010.

N. Ruozzi, J. Thaler, and S. Tatikonda. Graph covers and quadratic minimization. In *Proc. Communication, Control, and Computing, 47th Annual Allerton Conference on*, Allerton, IL, Sept. 2009.

D. Sontag and T. S. Jaakkola. Tree block coordinate descent for MAP in graphical models. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Clearwater Beach, Florida, April 2009.

S. Tatikonda and M. I. Jordan. Loopy belief propagation and Gibbs measures. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 493–500, Edmonton, Alberta, Canada, 2002.

P. O. Vontobel and R. Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. *CoRR*, abs/cs/0512078, 2005.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *Information Theory, IEEE Transactions on*, 49 (5):1120 – 1146, May 2003a.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation via pseudo-moment matching. In *Proceedings of the 9th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Key West, Florida, Jan. 2003b.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, 14(2): 143–166, 2004.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP estimation via agreement on (hyper)trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11): 3697–3717, Nov. 2005.

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

Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Information Theory, IEEE Transactions on*, 47(2):736 –744, Feb. 2001a.

Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Comput.*, 13(10):2173–2200, Oct. 2001b.

T. Werner. A linear programming approach to max-sum problem: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1165–1179, 2007.