# Estimation in Gaussian Graphical Models using Tractable Subgraphs: A Walk-Sum Analysis

*Venkat Chandrasekaran*, Jason K. Johnson, Alan S. Willsky*

***Abstract*— Graphical models provide a powerful formalism for statistical signal processing. Due to their sophisticated modeling capabilities, they have found applications in a variety of fields such as computer vision, image processing, and distributed sensor networks. In this paper, we present a general class of algorithms for estimation in Gaussian graphical models with arbitrary structure. These algorithms involve a sequence of inference problems on tractable subgraphs over subsets of variables. This framework includes parallel iterations such as Embedded Trees, serial iterations such as block Gauss-Seidel, and hybrid versions of these iterations. We also discuss a method that uses local memory at each node to overcome temporary communication failures that may arise in distributed sensor network applications. We analyze these algorithms based on the recently developed walk-sum interpretation of Gaussian inference. We describe the walks "computed" by the algorithms using *walk-sum diagrams*, and show that for iterations based on a very large and flexible set of sequences of subgraphs, convergence is guaranteed in walk-summable models. Consequently, we are free to choose spanning trees and subsets of variables *adaptively* at each iteration. This leads to efficient methods for optimizing the next iteration step to achieve maximum reduction in error. Simulation results demonstrate that these non-stationary algorithms provide a significant speedup in convergence over traditional one-tree and two-tree iterations.**

***Index Terms*— Graphical models, Gauss-Markov Random Fields, walk-sums, distributed estimation, walk-sum diagrams, subgraph preconditioners, maximum walk-sum tree, maximum walk-sum block.**

## I. Introduction

Graphical models offer a convenient representation for joint probability distributions and convey the Markov structure in a large number of random variables compactly. A graphical model [1, 2] is a collection of variables defined with respect to a graph; each vertex of the graph is associated with a random variable and the edge structure specifies the conditional independence properties among the variables. Due to their sophisticated modeling capabilities, graphical models (also known as Markov random fields or MRFs) have found applications in a variety of signal processing tasks involving distributed sensor networks [3], images [4, 5], and computer vision [6]. Our focus in this paper is on the important class of Gaussian graphical models, also known as Gauss-Markov random fields (GMRFs), which have been widely used to model natural phenomena in many large-scale estimation problems [7, 8].

In estimation problems in which the prior and observation models have normally distributed random components, computing the Bayes least-squares estimate is equivalent to solving a linear system of equations specified in terms of the information-form parameters of the conditional distribution. Due to its cubic computational complexity in the number of variables, direct matrix inversion to solve the Gaussian estimation problem is intractable in many applications in which the number of variables is very large (e.g., in oceanography problems [8] the number of variables may be on the order of $10^6$). For tree-structured MRFs (i.e., graphs with no cycles), Belief Propagation (BP) [9] provides an efficient linear complexity algorithm to compute exact estimates. However, tree-structured Gaussian processes possess limited modeling capabilities [10]. In order to model a richer class of statistical dependencies among variables, one often requires loopy graphical models. As estimation on graphs with cycles is substantially more complex, considerable effort has been and still is being put into developing methods that overcome this computational barrier, including a variety of methods that employ the idea of performing inference computations on tractable subgraphs [11, 12]. The recently proposed Embedded Trees (ET) iteration [10, 13] is one such approach that solves a sequence of inference problems on trees or, more generally, tractable subgraphs. If ET converges, it yields the correct conditional estimates, thus providing an effective inference algorithm for graphs with essentially arbitrary structure.

For the case of *stationary* ET iterations — in which the same tree or tractable subgraph is used at each iteration — necessary and sufficient conditions for convergence are provided in [10, 13]. However, experimental results in [13] provide compelling evidence that much faster convergence can often be obtained by changing the embedded subgraph that is used from one iteration to the next. The work in [13] provided very limited analysis for such *non-stationary* iterations, thus leaving open the problem of providing easily computable broadly applicable conditions that guarantee convergence.

In related work that builds on [10], Delouille et al. [14] describe a stationary block Gauss-Jacobi (GJ) iteration for solving the Gaussian estimation problem with the added constraint that messages between variables connected by an edge in the graph may occasionally be "dropped". The local blocks (subgraphs) are assumed to be small in size. Such a framework provides a simple model for estimation in distributed sensor networks where communication links between nodes may occasionally fail. The proposed solution involves the use of memory at each node to remember past messages from neighboring nodes. The values in this local memory are used if

*Corresponding author: Venkat Chandrasekaran. The authors are with the Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. Email: {venkatc, jasonj, willsky}@mit.edu. Fax: 617-258-8364. This work was supported by the Army Research Office grant W911NF–05–1–0207 and the Air Force Office of Scientific Research grant FA9550–04–1–0351.

there is a breakdown in communication to prevent the iteration from diverging. However, the analysis in [14] is also restricted to the case of stationary iterations, in that the same partitioning of the graph into local subgraphs is used at every iteration.

Finally, we note that ET iterations fall under the class of *parallel* update algorithms, in that every variable must be updated in an iteration before one can proceed to the next iteration. However, *serial* schemes involving updates over subsets of variables also offer tractable methods for solving large linear systems [15, 16]. An important example in this class of algorithms is block Gauss-Seidel (GS) in which each iteration involves updating a small subset of variables.

In this paper, we analyze non-stationary iterations based on an arbitrary sequence of embedded trees or tractable subgraphs. We refer to these trees and subgraphs on which inference is performed at each iteration as *preconditioners*, following the terminology used in the linear algebra literature. We present a general class of algorithms that includes the non-stationary ET and block GS iterations, and provide a general and very easily tested condition that guarantees convergence for any of these algorithms. Our framework allows for hybrid non-stationary algorithms that combine aspects of both block GS and ET. We also consider the problem of failing links and describe a method that uses local memory at each node to address this problem in general non-stationary parallel and serial iterations.

Our analysis is based on a recently introduced framework for interpreting and analyzing inference in GMRFs based on sums over walks in graphs [17]. We describe *walk-sum diagrams* that provide an intuitive interpretation of the estimates computed by each of the algorithms after every iteration. A walk-sum diagram is a graph that corresponds to the walks "accumulated" after each iteration. As developed in [17] walk-summability is an easily tested condition which, as we will show, yields a simple necessary and sufficient condition for the convergence of the algorithms. As there are broad classes of models (including attractive, diagonally-dominant, and so-called pairwise-normalizable models) that are walk-summable, our analysis shows that our algorithms provide a convergent, computationally attractive method for inference.

The walk-sum analysis and convergence results show that arbitrary non-stationary iterations of our algorithms based on a very large and flexible set of sequences of subgraphs or subsets of variables converge in walk-summable models. Consequently, we are free to use any sequence of trees in the ET algorithm or any valid sequence of subsets of variables (one that updates each variable infinitely often) in the block GS iteration, and still achieve convergence in walk-summable models. We exploit this flexibility by choosing trees or subsets of variables adaptively to minimize the error at iteration $n$ based on the residual error at iteration $n-1$. To make these choices optimally, we formulate combinatorial optimization problems that maximize certain re-weighted walk-sums. We describe efficient methods to solve relaxed versions of these problems. For the case of choosing the "next best" tree, our method reduces to solving a maximum-spanning tree problem. Simulation results indicate that our algorithms for choosing trees and subsets of variables adaptively provide a significant speedup in convergence over traditional approaches involving a single preconditioner or alternating between two preconditioners.

Our walk-sum analysis also shows that local memory at each node can be used to achieve convergence for any of the above algorithms when communication failures occur in distributed sensor networks. Our protocol differs from the description in [14], and as opposed to that work, allows for non-stationary updates. Also, our walk-sum diagrams provide a simple, intuitive representation for the propagation of information with each iteration.

One of the conditions for walk-summability in Section II-C shows that walk-summable models are equivalent to models for which the information matrix is an H-matrix [16, 18]. Several methods for finding good preconditioners for such matrices have been explored in the linear algebra literature, but these have been restricted to either cycling through a fixed set of preconditioners [19] or to so-called "multi-splitting" algorithms [20, 21]. These results do not address the problem of convergence of non-stationary iterations using arbitrary (non-cyclic) sequences of subgraphs. The analysis of such algorithms along with the development of methods to pick a good sequence of preconditioners are the main novel contributions of this paper, and the recently developed concept of walk-sums is critical to our analysis.

In Section II, we provide the necessary background about GMRFs and the walk-sum view of inference. Section III describes all the algorithms that we analyze in this paper, while Section IV contains the analysis and walk-sum diagrams that provide interpretations of the algorithms in terms of walk-sum computations. In Section V, we use the walk-sum interpretation of Section IV to show that these algorithms converge in walk-summable models. Section VI presents techniques for choosing tree-based preconditioners and subsets of variables adaptively for the ET and block GS iterations respectively, and demonstrates the effectiveness of these methods through simulation. We conclude with a brief discussion in Section VII. The appendix provides additional details and proofs.

## II. Gaussian graphical models and Walk-sums

### A. Gaussian graphical models and estimation

A graph $\mathcal{G} = (V, \mathcal{E})$ consists of a set of vertices $V$ and associated edges $\mathcal{E} \subset \binom{V}{2}$, where $\binom{V}{2}$ is the set of all unordered pairs of vertices. A subset $S \subset V$ is said to *separate* subsets $A, B \subset V$ if every path in $\mathcal{G}$ between any vertex in $A$ and any vertex in $B$ passes through a vertex in $S$. A graphical model [1, 2] is a collection of random variables indexed by the vertices of a graph; each vertex $s \in V$ corresponds to a random variable $x_s$, and where for any $A \subset V$, $x_A = \{x_s | s \in A\}$. A distribution $p(x_V)$ is *Markov* with respect to $\mathcal{G}$ if for any subsets $A, B \subset V$ that are separated by some $S \subset V$, the subset of variables $x_A$ is conditionally independent of $x_B$ given $x_S$, i.e. $p(x_A, x_B | x_S) = p(x_A | x_S)\ p(x_B | x_S)$.

We consider GMRFs $\{x_s | s \in V\}$ parameterized by a mean vector $\mu$ and a positive-definite covariance matrix $P$ (denoted by $P \succ 0$): $x_V \sim \mathcal{N}(\mu, P)$ [1, 22]. For simplicity, each $x_s$ is assumed to be a scalar variable. An alternate natural parameterization for GMRFs is specified in terms of the *information matrix* $J = P^{-1}$ (also called *precision* or *concentration* matrix) and *potential vector* $h = P^{-1}\mu$, and is denoted by $x_V \sim \mathcal{N}^{-1}(h, J)$. In particular, if $p(x_V)$ is Markov with respect to graph $\mathcal{G}$, then the specialization of the Hammersley-Clifford theorem for Gaussian models [1, 22] directly relates the sparsity of $J$ to the sparsity of $\mathcal{G}$: $J_{s,t} \neq 0$ if and only if the edge $\{s,t\} \in \mathcal{E}$ for every pair of vertices $s,t \in V$. The *partial correlation coefficient* $\rho_{s,t}$ is the correlation coefficient of variables $x_s$ and $x_t$ conditioned on knowledge of all the other variables [1]:

$$\rho_{s,t} \triangleq \frac{\text{cov}(x_s; x_t | x_{\setminus\{s,t\}})}{\sqrt{\text{var}(x_s|x_{\setminus\{s,t\}})\text{var}(x_t|x_{\setminus\{s,t\}})}} = -\frac{J_{s,t}}{\sqrt{J_{s,s}J_{t,t}}}. \quad (1)$$

Hence, $J_{s,t} = 0$ implies that $x_s$ and $x_t$ are conditionally independent given all the other variables $x_{\setminus\{s,t\}}$.

Let $x \sim \mathcal{N}^{-1}(h_{\text{prior}}, J_{\text{prior}})$, and suppose that we are given noisy observations $y = Cx + v$ of $x$, with $v \sim \mathcal{N}(0, S)$. The goal of the Gaussian estimation problem is to compute an estimate $\widehat{x}$ that minimizes the expected squared-error between $\widehat{x}$ and $x$. The solution to this problem is the mean of the posterior distribution $x|y \sim \mathcal{N}^{-1}(h, J)$, with $J = J_{\text{prior}} + C^T S^{-1} C$ and $h = h_{\text{prior}} + C^T S^{-1} y$ [23]. Thus, the posterior mean $\mu = J^{-1}h$ can be computed as the solution to the following linear system:

$$(J_{\text{prior}} + C^T S^{-1} C)\widehat{x} = h_{\text{prior}} + C^T S^{-1} y \quad \Leftrightarrow \quad J\widehat{x} = h. \quad (2)$$

We note that $J$ is a symmetric positive-definite matrix. If $C$ and $S$ are diagonal (corresponding to local measurements) $J$ has the same sparsity structure as that of $J_{\text{prior}}$. The conditions for all our convergence results and analysis in this paper are specified in terms of the posterior graphical model parameterized by $J$. As described in the introduction, solving the linear system (2) is computationally expensive by direct matrix inversion even for moderate-size problems. In this paper, we discuss tractable methods to solve this linear system.

## *B. Walk-summable Gaussian graphical models*

We assume that the information matrix $J$ of a Gaussian model defined on $\mathcal{G} = (V, \mathcal{E})$ has been normalized to have unit diagonal entries. For example, if $D$ is a diagonal matrix containing the diagonal entries of $J$, then the matrix $D^{-\frac{1}{2}} J D^{-\frac{1}{2}}$ contains re-scaled entries of $J$ at off-diagonal locations and 1's along the diagonal. Such a re-scaling does not affect the convergence results of the algorithms in this paper.[1] However, re-scaled matrices are useful in order to provide simple characterizations of walk-sums. Let $R = I - J$. The off-diagonal elements of $R$ are precisely the partial correlation coefficients from (1), and have the same sparsity structure as that of $J$ (and consequently the same structure as $\mathcal{G}$). Let these off-diagonal entries be the edge weights in $\mathcal{G}$, i.e. $R_{s,t} = \rho_{s,t}$ is the weight of the edge $\{s,t\}$. A *walk* in $\mathcal{G}$ is defined to be a sequence of vertices $w = \{w_i\}_{i=0}^{\ell}$ such that $\{w_i, w_{i+1}\} \in \mathcal{E}$ for each $i = 0, \ldots, \ell - 1$. Thus, there is no restriction on a walk crossing the same node or traversing the same edge multiple times. The *weight* of the walk $\phi(w)$ is defined:

$$\phi(w) \triangleq \prod_{i=0}^{\ell-1} R_{w_i, w_{i+1}}.$$

Note that the partial-correlation matrix $R$ is essentially a matrix of edge weights. Interpreted differently, one can also view each element of $R$ as the weight of the length-1 walk between two vertices. In general, $\left(R^\ell\right)_{s,t}$ is then the walk-sum $\phi(s \xrightarrow{\ell} t)$ over the (finite) set of all length-$\ell$ walks from $s$ to $t$ [17], where the *walk-sum* over a finite set is the sum of the weights of the walks in the set. Based on this point of view, we can interpret estimation in Gaussian models from equation (2) in terms of walk-sums:

$$P_{s,t} = \left((I - R)^{-1}\right)_{s,t} = \sum_{\ell=0}^{\infty} \left(R^\ell\right)_{s,t} = \sum_{\ell=0}^{\infty} \phi(s \xrightarrow{\ell} t). \quad (3)$$

Thus, the covariance between variables $x_s$ and $x_t$ is the length-ordered sum over all walks from $s$ to $t$. This, however, is a very specific instance of an inference algorithm that converges if the spectral radius condition $\varrho(R) < 1$ is satisfied (so that the matrix geometric series converges). Other inference algorithms, however, may compute walks in *different orders*. In order to analyze the convergence of general inference algorithms that submit to a walk-sum interpretation, a stronger condition was developed in [17] as follows. Given a countable set of walks $\mathcal{W}$, the *walk-sum* over $\mathcal{W}$ is the unordered sum of the individual weights of the walks contained in $\mathcal{W}$:

$$\phi(\mathcal{W}) \triangleq \sum_{w \in \mathcal{W}} \phi(w).$$

In order for this sum to be well-defined, we consider the following class of Gaussian graphical models.

**Definition 1:** A Gaussian graphical model defined on $\mathcal{G} = (V, \mathcal{E})$ is said to be *walk-summable* if the absolute walk-sums over the set of all walks between every pair of vertices in $\mathcal{G}$ are well-defined. That is, for every pair $s, t \in V$,

$$\bar{\phi}(s \to t) \triangleq \sum_{w \in \mathcal{W}(s \to t)} |\phi(w)| < \infty.$$

Here, $\bar{\phi}$ denotes absolute walk-sums over a set of walks. $\mathcal{W}(s \to t)$ corresponds to the set of all walks[2] beginning at vertex $s$ and ending at the vertex $t$ in $\mathcal{G}$. Section II-C lists some easily tested equivalent and sufficient conditions for walk-summability. Based on the absolute convergence condition, walk-summability implies that walk-sums over a countable set of walks can be computed in *any order* and that the unordered walk-sum $\phi(s \to t)$ is well-defined [24, 25]. Therefore, in walk-summable models, the covariances and means can be interpreted as follows:

$$P_{s,t} = \phi(s \to t), \quad (4)$$

[1]Although our analysis of the algorithms in Section III is specified for normalized models, these algorithms and our analysis can be easily extended to the un-normalized case. See Appendix A.

[2]We denote walk-sets by $\mathcal{W}$ but generally drop this notation when referring to the walk-sum over $\mathcal{W}$, i.e. the walk-sum of the set $\mathcal{W}(\sim)$ is denoted by $\phi(\sim)$.

$$\mu_t = \sum_{s \in V} P_{t,s} h_s = \sum_{s \in V} h_s P_{s,t} = \sum_{s \in V} h_s \phi(s \to t), \quad (5)$$

where (3) is used in the first equation, and (4) in the second. In words, the covariance between variables $x_s$ and $x_t$ is the walk-sum over the set of all walks from $s$ to $t$, and the mean of variable $x_t$ is the walk-sum over all walks ending at $t$ with each walk being re-weighted by the potential value at the starting node.

The goal in walk-sum analysis is to interpret an inference algorithm as the computation of walk-sums in $\mathcal{G}$. If the analysis shows that the walks being computed by an inference algorithm are the same as those required for the computation of the means and covariances above, then the correctness of the algorithm can be concluded directly for walk-summable models. This conclusion can be reached regardless of the order in which the algorithm computes the walks due to the fact that walk-sums can be computed in *any* order in walk-summable models. Thus, the walk-sum formalism allows for very strong yet intuitive statements about the convergence of inference algorithms that submit to a walk-sum interpretation. Indeed, the overall template for analyzing our inference algorithms is simple. First, we show that the algorithms submit to a walk-sum interpretation. Next, we show that the walk-sets computed by these algorithms are nested, i.e. $\mathcal{W}_n \subseteq \mathcal{W}_{n+1}$, where $\mathcal{W}_n$ is the set of walks computed at iteration $n$. Finally, we show that every walk required for the computation of the mean (5) is contained in $\mathcal{W}_n$ for some $n$. A key ingredient in our analysis is that in computing all the walks in (5), the algorithms must *not overcount* any walks. Although each step in this procedure is non-trivial, combined together they allow us to conclude that the algorithms converge in walk-summable models.

### C. Properties of walk-summable models

Very importantly, there are easily testable necessary and sufficient conditions for walk-summability. Let $\bar{R}$ denote the matrix of the absolute values of the elements of $R$. Then, walk-summability is equivalent to either [17]

- $\varrho(\bar{R}) < 1$, or
- $I - \bar{R} \succ 0$.

From the second condition, one can draw a connection to H-matrices in the linear algebra literature [16, 18]. Specifically, walk-summable information matrices are symmetric, positive-definite H-matrices.

Walk-summability of a model is sufficient but not necessary for the validity of the model (positive-definite information/covariance). Many classes of models are walk-summable [17]:

1) Diagonally-dominant models, i.e. for each $s \in V$, $\sum_{t \neq s} |J_{s,t}| < J_{s,s}$.
2) Valid non-frustrated models, i.e. every cycle has an even number of negative edge weights and $I - R \succ 0$. Special cases include valid attractive models ($R_{s,t} \geq 0$ for all $s, t \in V$) and tree-structured models.
3) *Pairwise normalizable* models, i.e. there exists a diagonal matrix $D \succ 0$ and a collection of matrices $\{J_e \succeq 0 | (J_e)_{s,t} = 0 \text{ if } (s,t) \neq e,\ e \in \mathcal{E}\}$ such that $J = D + \sum_{e \in \mathcal{E}} J_e$.

An example of a commonly encountered walk-summable model in statistical image processing is the thin-membrane prior [26]. Further, linear systems involving sparse diagonally dominant matrices are also a common feature in finite element approximations of elliptical partial differential equations [27].

We now describe some operations that can be performed on walk-sets, and the corresponding walk-sum formulas. These relations are valid in walk-summable models [17]:

- Let $\{\mathcal{U}_n\}_{n=1}^{\infty}$ be a countable collection of mutually disjoint walk-sets. From the sum-partition theorem for absolutely summable series [25], we have that $\phi(\cup_{n=1}^{\infty} \mathcal{U}_n) = \sum_{n=1}^{\infty} \phi(\mathcal{U}_n)$. This implies that for a countable collection of walk-sets $\{\mathcal{V}_n\}_{n=1}^{\infty}$ where $\mathcal{V}_{n-1} \subseteq \mathcal{V}_n$, we have that $\phi(\cup_{n=1}^{\infty} \mathcal{V}_n) = \lim_{n \to \infty} \phi(\mathcal{V}_n)$. This is easily seen by defining mutually disjoint walk-sets $\{\mathcal{U}_n\}_{n=1}^{\infty}$ with $\mathcal{U}_n = \mathcal{V}_n \backslash \mathcal{V}_{n-1}$.
- Let $u = u_0 u_1 \cdots u_{\text{end}}$ and $v = v_{\text{start}} v_1 \cdots v_{\ell(v)}$ be walks such that $u_{\text{end}} = v_{\text{start}}$. The concatenation of the walks is defined to be $u \cdot v \triangleq u_0 u_1 \cdots u_{\text{end}} v_1 \cdots v_{\ell(v)}$. Now consider a walk-set $\mathcal{U}$ with all walks ending at vertex $u_{\text{end}}$ and a walk-set $\mathcal{V}$ with all walks starting at $v_{\text{start}} = u_{\text{end}}$. The concatenation of the walk-sets $\mathcal{U}, \mathcal{V}$ is defined:
$$\mathcal{U} \otimes \mathcal{V} \triangleq \{u \cdot v \mid u \in \mathcal{U}, v \in \mathcal{V}\}.$$
If every walk $w \in \mathcal{U} \otimes \mathcal{V}$ can be decomposed uniquely into $u \in \mathcal{U}$ and $v \in \mathcal{V}$ so that $w = u \cdot v$, then $\mathcal{U} \otimes \mathcal{V}$ is said to be *uniquely decomposable* into the sets $\mathcal{U}, \mathcal{V}$. For such uniquely decomposable walk-sets, $\phi(\mathcal{U} \otimes \mathcal{V}) = \phi(\mathcal{U})\phi(\mathcal{V})$.

Finally, the following notational convention is employed in the rest of this paper. We use wild-card symbols ($*$ and $\bullet$) to denote a union over all vertices in $\mathcal{G}$. For example, given a collection of walk-sets $\mathcal{W}(s)$, we interpret $\mathcal{W}(*)$ as $\bigcup_{s \in V} \mathcal{W}(s)$. Further, the walk-sum over the set $\mathcal{W}(*)$ is defined $\phi(\mathcal{W}(*)) \triangleq \sum_{s \in V} \phi(\mathcal{W}(s))$. In addition to edges being assigned weights, vertices can also be assigned weights (for example, the potential vector $h$). A re-weighted walk-sum of a walk $w = w_0 \cdots w_\ell$ with vertex weight vector $h$ is then defined to be $\phi(h; w) \triangleq h_{w_0} \phi(w)$. Based on this notation, the mean of variable $x_t$ from (5) can be re-written as

$$\mu_t = \phi(h; * \to t). \quad (6)$$

## III. Non-Stationary Embedded Subgraph Algorithms

In this section, we describe a framework for the computation of the conditional mean estimates in order to solve the Gaussian estimation problem of Section II-A. We present three algorithms that become successively more complex in nature. We begin with the parallel ET algorithm originally presented in [10, 13]. Next, we describe a serial update scheme that involves processing only a subset of the variables at each iteration. Finally, we discuss a generalization to these non-stationary algorithms that is tolerant to temporary communication failure by using local memory at each node to remember past messages from neighboring nodes. A similar memory-based approach was used in [14] for the special case of stationary iterations. The key theme underlying all these algorithms is that they are based on solving a sequence of inference problems on tractable subgraphs involving all or a subset of the variables. Convergent iterations that compute means can also be used to compute exact error variances [10]. Hence, we restrict ourselves to analyzing iterations that compute the conditional mean.

## *A. Non-Stationary Parallel Updates: Embedded Trees*

Let $\mathcal{S}$ be some subgraph of the graph $\mathcal{G}$. The stationary ET algorithm is derived by splitting the matrix $J = J_\mathcal{S} - K_\mathcal{S}$, where $J_\mathcal{S}$ is known as the *preconditioner* and $K_\mathcal{S}$ is known as the *cutting matrix*. Each edge in $\mathcal{G}$ is either an element of $\mathcal{S}$ or $\mathcal{E}\backslash\mathcal{S}$. Accordingly, every non-zero off-diagonal entry of $J$ is either an element of $J_\mathcal{S}$ or of $-K_\mathcal{S}$. The diagonal entries of $J$ are part of $J_\mathcal{S}$. Hence, the matrix $K_\mathcal{S}$ is symmetric, zero along the diagonal, and contains non-zero entries only in those locations that correspond to edges not included in the subgraph generated by the splitting. Cutting matrices may have non-zero diagonal entries in general, but we only consider zero-diagonal cutting matrices in this paper. The splitting of $J$ according to $\mathcal{S}$ transforms (2) to $J_\mathcal{S}\widehat{x} = K_\mathcal{S}\widehat{x} + h$, which suggests an iterative method to solve (2) by recursively solving $J_\mathcal{S}\widehat{x}^{(n)} = K_\mathcal{S}\widehat{x}^{(n-1)} + h$, yielding a sequence of estimates:

$$\widehat{x}^{(n)} = J_\mathcal{S}^{-1}(K_\mathcal{S}\widehat{x}^{(n-1)} + h). \tag{7}$$

If $J_\mathcal{S}^{-1}$ exists then a necessary and sufficient condition for the iterates $\{\widehat{x}^{(n)}\}_{n=0}^{\infty}$ to converge to $J^{-1}h$ for any initial guess $\widehat{x}^{(0)}$ is that $\varrho(J_\mathcal{S}^{-1}K_\mathcal{S}) < 1$ [10]. ET iterations can be very effective if applying $J_\mathcal{S}^{-1}$ to a vector is efficient, e.g. if $\mathcal{S}$ corresponds to a tree or, in general, any tractable subgraph.

A non-stationary ET iteration is obtained by letting $J = J_{\mathcal{S}_n} - K_{\mathcal{S}_n}$, where the matrices $J_{\mathcal{S}_n}$ correspond to some embedded tree or subgraph $\mathcal{S}_n$ in $\mathcal{G}$ and can vary in an arbitrary manner with $n$. This leads to the following ET iteration:

$$\widehat{x}^{(n)} = J_{\mathcal{S}_n}^{-1}(K_{\mathcal{S}_n}\widehat{x}^{(n-1)} + h). \tag{8}$$

Our walk-sum analysis proves the convergence of non-stationary ET iterations based on any sequence of subgraphs $\{\mathcal{S}_n\}_{n=1}^{\infty}$ in walk-summable models. Every step of the above algorithm is tractable if applying $J_{\mathcal{S}_n}^{-1}$ to a vector can be performed efficiently. Indeed, an important degree of freedom in the above algorithm is the choice of $\mathcal{S}_n$ at each stage so as to speed up convergence, while keeping the computation at every iteration tractable. We discuss some approaches to addressing this issue in Section VI.

## *B. Non-Stationary Serial Updates of Subsets of Variables*

We begin by describing the block GS iteration [15, 16]. For each $n = 1, 2, \ldots$, let $V_n \subseteq V$ be some subset of $V$. The variables $x_{V_n} = \{x_s : s \in V_n\}$ are updated at iteration $n$. The remaining variables do not change from iteration $n-1$ to $n$. Let $J^{(n)} = [J]_{V_n}$ be the $|V_n| \times |V_n|$-dimensional principal sub-matrix corresponding to the variables $V_n$. The block GS update at iteration $n$ is as follows:

$$\widehat{x}_{V_n}^{(n)} = {J^{(n)}}^{-1}\left(R_{V_n,V_n^c}\ \widehat{x}_{V_n^c}^{(n-1)} + h_{V_n}\right), \tag{9}$$
$$\widehat{x}_{V_n^c}^{(n)} = \widehat{x}_{V_n^c}^{(n-1)}. \tag{10}$$

Here, $V_n^c$ refers to the complement of the vertex set $V_n$. In equation (9), $R_{V_n,V_n^c}$ refers to the sub-matrix of edge weights of edges from the vertices $V_n^c$ to $V_n$. Every step of the above algorithm is tractable as long as applying ${J^{(n)}}^{-1}$ to a vector can be performed efficiently.

We now present a general serial iteration that incorporates an element of the ET algorithm of Section III-A. This update scheme involves a single ET iteration within the induced subgraph of the update variables $V_n$. We split the edges $\mathcal{E}(V_n)$ in the induced subgraph of $V_n$ into a tractable set $\mathcal{S}_n$ and a set of cut edges $\mathcal{E}(V_n)\backslash\mathcal{S}_n$. Such a splitting leads to a tractable subgraph $\mathcal{S}_n$ of the induced subgraph of $V_n$. That is, the matrix $J^{(n)}$ is split as $J^{(n)} = J_{\mathcal{S}_n} - K_{\mathcal{S}_n}$. This matrix splitting is defined analogous to the splitting in Section III-A. The modified conditional mean update at iteration $n$ is as follows:

$$\widehat{x}_{V_n}^{(n)} = J_{\mathcal{S}_n}^{-1}\left(K_{\mathcal{S}_n}\widehat{x}_{V_n}^{(n-1)} + R_{V_n,V_n^c}\widehat{x}_{V_n^c}^{(n-1)} + h_{V_n}\right), \tag{11}$$
$$\widehat{x}_{V_n^c}^{(n)} = \widehat{x}_{V_n^c}^{(n-1)}. \tag{12}$$

Every step of this algorithm is tractable as long as applying $J_{\mathcal{S}_n}^{-1}$ to a vector can be performed efficiently.

The preceding algorithm is a generalization of both the block GS update (9)−(10) and the non-stationary ET algorithm (8), thus allowing for a unified analysis framework. Specifically, by letting $\mathcal{S}_n = \mathcal{E}(V_n)$ for all $n$ above, we obtain the block GS algorithm. On the other hand, by letting $V_n = V$ for all $n$, we recover the ET algorithm. This hybrid approach also offers a tractable and flexible method for inference in large-scale estimation problems, because it possesses all the benefits of the ET and block GS iterations.

We note that in general applications there is one potential complication with both the serial and the parallel iterations presented so far. Specifically, for an arbitrary graphical model with positive-definite information matrix $J$, the corresponding information sub-matrix $J_{\mathcal{S}_n}$ for some choices of subgraphs $\mathcal{S}_n$ may be invalid or even singular, i.e. may have negative or zero eigenvalues.[3] Importantly, this problem *never* arises for walk-summable models, and thus we are free to use any sequence of embedded subgraphs for our iterations and be guaranteed that the computations make sense probabilistically.

**Lemma 1:** Let $J$ be a walk-summable model, let $\widetilde{V} \subseteq V$, and let $J_\mathcal{S}$ be the $|\widetilde{V}| \times |\widetilde{V}|$-dimensional information matrix corresponding to the distribution over some subgraph $\mathcal{S}$ of the induced subgraph $\mathcal{E}(\widetilde{V})$. Then, $J_\mathcal{S}$ is walk-summable, and $J_\mathcal{S} \succ 0$.

**Proof**: For every pair of vertices $s, t \in \widetilde{V}$, it is clear that the walks between $s$ and $t$ in $\mathcal{S}$ are a subset of the walks between these vertices in $\mathcal{G}$, i.e. $\mathcal{W}(s \xrightarrow{\mathcal{S}} t) \subseteq \mathcal{W}(s \rightarrow t)$. Hence, $\bar{\phi}(s \xrightarrow{\mathcal{S}} t) \leq \bar{\phi}(s \rightarrow t) < \infty$, because $J$ is walk-summable. Thus, the model specified by $J_\mathcal{S}$ is walk-summable. This allows us to conclude that $J_\mathcal{S} \succ 0$ because walk-summability implies validity of a model. □

[3]For example, consider a 5-cycle with each edge having a partial correlation of $-0.6$. This model is valid (but not walk-summable) with the corresponding $J$ having a minimum eigenvalue of 0.0292. A spanning tree model $J_\mathcal{S}$ obtained by removing one of the edges in the cycle, however, is invalid with a minimum eigenvalue of $-0.0392$.

### C. Distributed Interpretation of (11)–(12) and Communication Failure

We first re-interpret the equations (11)–(12) as local message-passing steps between nodes followed by inference within the subgraph $\mathcal{S}_n$. At iteration $n$, let $\kappa_n$ denote the set of *directed* edges in $\mathcal{E}(V_n)\backslash\mathcal{S}_n$ and from $V_n^c$ to $V_n$:

$$\kappa_n \triangleq \{(s,t) \mid \{s,t\} \in \mathcal{E}(V_n)\backslash\mathcal{S}_n \text{ or } s \in V_n^c, t \in V_n\}. \quad (13)$$

The edge set $\kappa_n$ corresponds to the non-zero elements of the matrices $K_{\mathcal{S}_n}$ and $R_{V_n,V_n^c}$ in equation (11). Edges in $\kappa_n$ are used to communicate information about the values at iteration $n-1$ to neighboring nodes for processing at iteration $n$.

For each $t \in V_n$, the message $M(s \to t) = R_{t,s}\ \widehat{x}_s^{(n-1)}$ is sent at iteration $n$ from $s$ to $t$ using the links in $\kappa_n$. Let $M_n(t)$ denote the summary of all the messages received at node $t$ at iteration $n$:

$$M_n(t) = \sum_{\{s|(s,t)\in\kappa_n\}} M(s\to t) = \sum_{\{s|(s,t)\in\kappa_n\}} R_{t,s}\ \widehat{x}_s^{(n-1)}. \quad (14)$$

Thus, each $t \in V_n$ *fuses* all the information received about the previous iteration and combines this with its local potential value $h_t$ to form a modified potential vector that is then used for inference within the subgraph $\mathcal{S}_n$:

$$\widehat{x}_{V_n}^{(n)} = J_{\mathcal{S}_n}^{-1}\ (M_n(V_n) + h_{V_n}), \quad (15)$$

where $M_n(V_n)$ denotes the entire vector of fused messages $M_n(t)$ for $t \in V_n$. An interesting aspect of these message-passing operations is that they are *local* and only nodes that are neighbors in $\mathcal{G}$ may participate in any communication. If the subgraph $\mathcal{S}_n$ is tree-structured, the inference step (15) can also be performed efficiently in a distributed manner using only local BP messages [9].

We now present an algorithm that is tolerant to temporary link failure by using local memory at each node $t$ to store the most recent message $M(s \to t)$ received at $t$ from $s$. If the link $(s,t)$ fails at some future iteration the stored message can be used in place of the new expected message. In order for the overall memory-based protocol to be consistent, we also introduce an additional post-inference message-passing step at each iteration. To make the above points precise, we specify a memory protocol that the network must follow; we assume that each node in the network has sufficient memory to store the most-recent messages received from its neighbors. First, $\mathcal{S}_n$ must not contain any failed links; every link $\{s,t\} \in \mathcal{E}(V_n)$ that fails at iteration $n$ must be a part of the cut-set[4]: $(s,t),(t,s) \in \kappa_n$. Therefore, the links $\mathcal{S}_n$ that are used for the inference step (15) must be active at iteration $n$. Second, in order for nodes to synchronize after each iteration, they must perform a post-inference message-passing step. *After* the inference step (15) at iteration $n$, the variables in $V_n$ must update their neighbors *in the subgraph* $\mathcal{S}_n$. That is, for each $t \in V_n$, a message must be received post-inference from every $s$ such that $\{s,t\} \in \mathcal{S}_n$:

$$M(s\to t) = R_{t,s}\ \widehat{x}_s^{(n)}. \quad (16)$$

[4]One way to ensure this is to select $\mathcal{S}_n$ to explicitly avoid the failed links. See Section VI-B for more details.

This operation is possible since the edge $\{s,t\}$ is assumed to active. Apart from these two rules, all other aspects of the algorithm presented previously remain the same. Note that every new message received overwrites the existing stored message, and only the most recent message received is stored in memory.

Thus, link failure affects only equation (14) in our iterative procedure. Suppose that a message to be received at $t \in V_n$ from node $s$ is unavailable due to communication failure. The message $M(s \to t)$ from memory can be used instead in the fusion formula (14). Let $r_n(s \to t)$ denote the iteration count of the most recent information at node $t$ about $s$ at the information fusion step (14) at iteration $n$. In general, $r_n(s \to t) \leq n-1$, with equality if $t \in V_n$ and $(s,t) \in \kappa_n$ is active. With this notation, we can re-write the fusion equation (14):

$$M_n(t) = \sum_{\{s|(s,t)\in\kappa_n\}} M(s\to t) = \sum_{\{s|(s,t)\in\kappa_n\}} R_{t,s}\ \widehat{x}_s^{r_n(s\to t)}. \quad (17)$$

## IV. Walk-sum interpretation and Walk-sum diagrams

In this section, we analyze each iteration of the algorithms of Section III as the computation of walk-sums in $\mathcal{G}$. Our analysis is presented for the most general algorithm involving failing links, since the parallel and serial non-stationary updates without failing links are special cases. For each of these algorithms, we then present walk-sum diagrams that provide intuitive, graphical interpretations of the walks being computed. Examples that we discuss include classic methods such as GJ and GS, and iterations involving general subgraphs. Throughout this section, we assume that the initial guess $\widehat{x}^{(0)} = 0$, and we initialize $M(s \to t) = 0$ and $r_1(s \to t) = 0$ for each directed edge $(s,t) \in \mathcal{E}$. In Section V, we prove the convergence of our algorithms for any initial guess $\widehat{x}^{(0)}$.

### A. Walk-sum interpretation

For every pair of vertices $s,t \in V$, we define a recursive sequence of walk-sets. We then show that these walk-sets are exactly the walks being computed by the iterative procedure in Section III-C:

$$\mathcal{W}_n(s\to t) = \mathcal{W}_{r_n(*\to\bullet)}(s\to *)\otimes\mathcal{W}(*\xrightarrow{\kappa_n(1)}\bullet)\otimes\mathcal{W}(\bullet\xrightarrow{\mathcal{S}_n}t)$$
$$\bigcup\ \mathcal{W}(s\xrightarrow{\mathcal{S}_n}t),\ s\in V, t\in V_n, \quad (18)$$
$$\mathcal{W}_n(s\to t) = \mathcal{W}_{n-1}(s\to t),\ s\in V, t\in V_n^c, \quad (19)$$

with

$$\mathcal{W}_0(s\to t) = \emptyset,\ s,t\in V. \quad (20)$$

The notation in these equations is defined in Section II-C. $\mathcal{W}_{r_n(*\to\bullet)}(s\to *)$ denotes the walks starting at node $s$ computed up to iteration $r_n(*\to\bullet)$. $\mathcal{W}(*\xrightarrow{\kappa_n(1)}\bullet)$ corresponds to a length-1 walk (called a *hop*) across a directed edge in $\kappa_n$. Finally, $\mathcal{W}(\bullet\xrightarrow{\mathcal{S}_n}t)$ denotes walks within $\mathcal{S}_n$ that end at $t$. Thus, the first RHS term in (18) is the set of previously computed walks starting at $s$ that hop across an edge in $\kappa_n$, and then propagate within $\mathcal{S}_n$ (ending at $t$). $\mathcal{W}(s\xrightarrow{\mathcal{S}_n}t)$ is

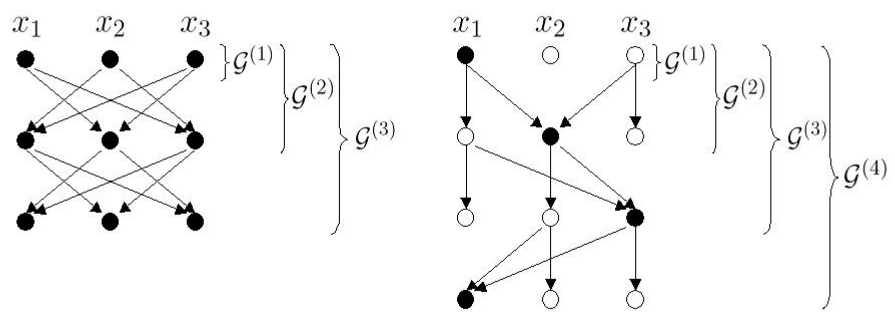


Fig. 1. (Left) Gauss-Jacobi walk-sum diagrams $\mathcal{G}^{(n)}$ for $n = 1, 2, 3$. (Right) Gauss-Seidel walk-sum diagrams $\mathcal{G}^{(n)}$ for $n = 1, 2, 3, 4$.

the set of walks from $s$ to $t$ that live entirely within $\mathcal{S}_n$. To simplify notation, we define $\phi_n(s \to t) \triangleq \phi(\mathcal{W}_n(s \to t))$. We now relate the walk-sets $\mathcal{W}_n(s \to t)$ to the estimate $\widehat{x}_t^{(n)}$ at iteration $n$.

**Proposition 1:** At iteration $n = 0, 1, \ldots,$ with $\widehat{x}^{(0)} = 0$, the estimate for node $t \in V$ in walk-summable models is given by:

$$\widehat{x}_t^{(n)} = \sum_{s \in V} h_s \phi_n(s \to t) = \phi_n(h; * \to t), \qquad (21)$$

where the walk-sum is over the walk-sets defined by (18−20), and $\widehat{x}_t^{(n)}$ is computed using (15,17).

This proposition, proven in Appendix B, states that each of our algorithms has a precise walk-sum interpretation. A consequence of this statement is that no walk is over-counted, i.e., each walk in $\mathcal{W}_n$ submits to a unique decomposition with respect to the construction process (18−20) (see proof for details), and appears exactly once in the sum at each iteration. As discussed in Section V (Propositions 3 and 4), the iterative process does even more; the walk-sets at successive iterations are nested and, under an appropriate condition, are "complete" so that convergence is guaranteed for walk-summable models. Showing and understanding all these properties are greatly facilitated by the introduction of a visual representation of how each of our algorithms computes walks, and that is the subject of the next subsection.

## B. Walk-sum diagrams

In the rest of this section, we present a graphical interpretation of our algorithms, and of the walk-sets $\mathcal{W}_n$ (18−20) that are central to Proposition 1 (which in turn is the key to our convergence analysis in Section V). This interpretation provides a clearer picture of memory usage and information flow at each iteration. Specifically, for each algorithm we construct a sequence of graphs $\mathcal{G}^{(n)}$ such that a particular set of walks in these graphs corresponds *exactly* to the sets $\mathcal{W}_n$ (18−20) computed by the sequence of iterates $\widehat{x}^{(n)}$. The graphs $\mathcal{G}^{(n)}$ are called *walk-sum diagrams*. Recall that $\mathcal{S}_n$ corresponds to the subgraph used at iteration $n$, generally using some of the values computed from a preceding iteration. The graph $\mathcal{G}^{(n)}$ captures all of these preceding computations leading up to and including the computations at iteration $n$.

As a result, $\mathcal{G}^{(n)}$ has very specific structure for each algorithm. It consists of a number of *levels* — within each level we capture the subgraph used at the corresponding iteration, and the final level $n$ corresponds to the results at the end of iteration $n$. Although some variables may not be updated at each iteration, the values of those variables are preserved for use in subsequent iterations; thus, each level of $\mathcal{G}^{(n)}$ includes all the nodes in $V$. The update variables at any iteration (i.e., the nodes in $\mathcal{S}_n$) are represented as solid circles, and the non-update ones as open circles. All edges in each $\mathcal{S}_n$ — edges of $\mathcal{G}$ included in this subgraph — are included in that level of the diagram. As in $\mathcal{G}$, these are undirected edges, as our algorithms perform inference on this subgraph. However, this inference update uses some values from preceding iterations (15,17); hence, we use directed edges (corresponding to $\kappa_n$) from nodes at preceding levels. The directed nature of these edges is critical as they capture the one-directional flow of computations from iteration to iteration, while the undirected edges within each level capture the inference computation (15) at each iteration. At the end of iteration $n$, only the values at level $n$ are of interest. Therefore, the set of walks (re-weighted by $h$) in $\mathcal{G}^{(n)}$ that begin at any solid node at any level, and end at any node at the last level are of importance, where walks can only move in the direction of directed edges between levels, but in any direction along the undirected edges within each level.

Later in this section we provide a general procedure for constructing walk-sum diagrams for our most general algorithms, but we begin by illustrating these diagrams and the points made in the preceding paragraph using a simple 3-node, fully connected graph (with variables denoted $x_1, x_2, x_3$). We look at two of the simplest iterative algorithms in the classes we have described, namely the classic GJ and GS iterations [15, 16]. Figure 1 shows the walk-sum diagrams for these algorithms.

In the GJ algorithm each variable is updated at each iteration using the values from the preceding iteration of every other variable (this corresponds to a stationary ET algorithm (7) with the subgraph $\mathcal{S}_n$ being the fully disconnected graph of all the nodes $V$). Thus each level on the left in Figure 1 is fully disconnected, with solid nodes for all variables and directed edges from each node at the preceding level to every other node at the next level. This provides a simple way of seeing both how walks are extended from one level to the next and, more subtly, how walks captured at one iteration are also captured at subsequent iterations. For example, the walk 12 in $\mathcal{G}^{(2)}$ is captured by the directed edge that begins at node 1 at level 1 and proceeds to node 2 at level 2 (the final level of

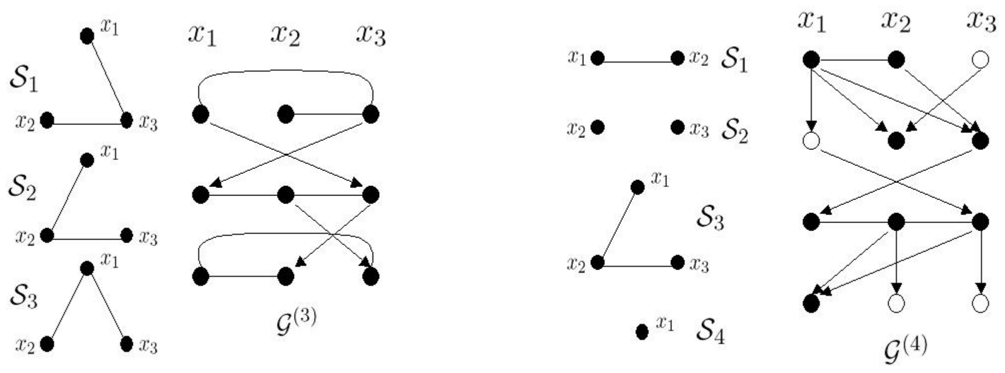


Fig. 2. (Left) Non-stationary ET: subgraphs and walk-sum diagram. (Right) Hybrid serial updates: subgraphs and walk-sum diagram.

$\mathcal{G}^{(2)}$). However, this walk in $\mathcal{G}^{(3)}$ is captured by the walk that begins at node 1 at level 2 and proceeds to node 2 at level 3 in $\mathcal{G}^{(3)}$.

The GS algorithm is a serial iteration that updates one variable at a time, cyclically, so that after $|V|$ iterations each variable is updated exactly once. On the right-hand side of Figure 1, only one node at each level is solid, using values of the other nodes from the preceding level. For non-update variables at any iteration, a weight-1 directed edge is included from the same node at the preceding level. For example, since $x_2$ is updated at level 2, we have open circles for nodes 1 and 3 at that level and weight-1 directed edges from their copies at level 1. Weight-1 edges do not affect the weight of any walk. Hence, at level 4 we still capture the walk 12 from level 2 (from node 1 at level 1 to node 2 at level 2); the walk is extended to node 2 at levels 3 and 4 with weight-1 directed edges.

For general graphs, the walk-sum diagram $\mathcal{G}^{(n)}$ of one of our algorithms is constructed as follows:

1) For $n = 1$, create a new copy of each $t \in V$ using solid circles for update variables and open circles for non-update variables; label these $t^{(1)}$. Draw the subgraph $\mathcal{S}_1$ using the solid nodes and undirected edges weighted by the partial correlation coefficient of each edge. $\mathcal{G}^{(1)}$ is the same as $\mathcal{S}_1$ with the exception that $\mathcal{G}^{(1)}$ also contains non-update variables denoted by open circles.
2) Given $\mathcal{G}^{(n-1)}$, create a new copy of each $t \in V$ using solid circles for update variables and open circles otherwise; label these $t^{(n)}$. Draw $\mathcal{S}_n$ using the update variables with undirected edges. Draw a *directed* edge from the variable $u^{r_n(u\to v)}$ in $\mathcal{G}^{(n-1)}$ (since $r_n(u \to v) \leq n-1$) to $v^{(n)}$ for each $(u, v) \in \kappa_n$. If there are no failed links, $r_n(u \to v) = n - 1$. Both these undirected and directed edges are weighted by their respective partial correlation coefficients. Draw a directed edge to each non-update variable $t^{(n)}$ from the corresponding $t^{(n-1)}$ with unit edge weight.

A *level* $k$ in a walk-sum diagram refers to the $k$'th replica of the variables.

**Rules for walks in** $\mathcal{G}^{(n)}$: Walks must respect the orientation of each edge, i.e., walks can cross an undirected edge in either direction, but can only cross directed edges in one direction. In addition, walks can only start at the update variables $V_k$ for each level $k \leq n$. Interpreted in this manner, walks in $\mathcal{G}^{(n)}$ re-weighted by $h$ and ending at one of the variables $t^{(k)}$ are exactly the walks computed in $\widehat{x}_t^{(k)}$.

**Proposition 2:** Let $\mathcal{G}^{(n)}$ be a walk-sum diagram constructed and interpreted according to the preceding rules. In walk-summable models, for any $t \in V$ and $k \leq n$,

$$\widehat{x}_t^{(k)} = \phi(h; * \xrightarrow{\mathcal{G}^{(n)}} t^{(k)}). \tag{22}$$

**Proof**: Based on the preceding discussion, one can check the equivalence of the walks computed by the walk-sum diagrams with the walk-sets (18$-$20). Proposition 1 then yields (22). □

The following sections describe walk-sum diagrams for the various algorithms presented in Section III.

## C. Non-Stationary Parallel Updates

We describe walk-sum diagrams for the parallel ET algorithm of Section III-A. Here, $V_n = V$ for all $n$. Since there is no link failure $r_n(* \to \bullet) = n - 1$. Hence, the walk-sum formulas (18$-$19) reduce to

$$\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t) \\ \bigcup \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t), \;\; s, t \in V. \tag{23}$$

The stationary GJ iteration discussed previously falls in this class. The left-hand side of Figure 2 shows the trees $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$, and the corresponding first three levels of the walk-sum diagrams for a more general non-stationary ET iteration. This example illustrates how walks are "collected" in walk-sum diagrams at each iteration. First, walks can proceed along undirected edges within each level, and from one level to the next along directed edges (capturing cut edges). Second, the walks relevant at each iteration must end at that level. For example, the walk 13231 is captured at iteration 1 as it is present in the undirected edges at level 1. At iteration 2, however, we are interested in walks ending at level 2. The walk 13231 is still captured, but in a different manner — through the walk 1323 at level 1, followed by the hop 31 along the directed edge from node 3 at level 1 to node 1 at level 2. At iteration 3, this walk is captured first by the hop from node 1 at level 1 to node 3 at level 2, then by the hop 32 at level 2, followed by the hop from node 2 at level 2 to node 3 at level 3, and finally by the hop 31 at level 3.

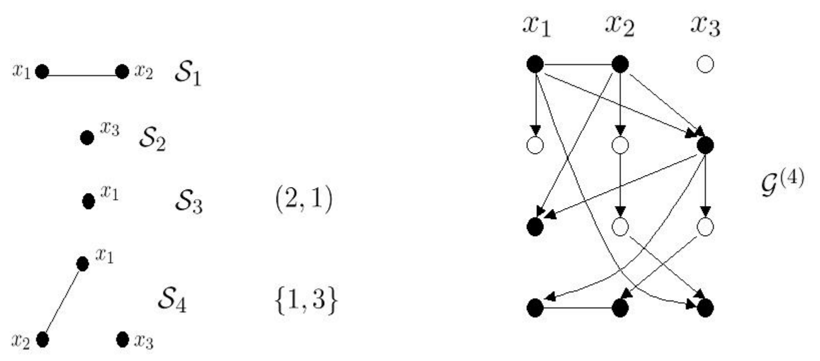


Fig. 3. Non-stationary updates with failing links: Subgraphs used along with failed edges at each iteration (left) and walk-sum diagram $\mathcal{G}^{(4)}$ (right).

### D. Non-Stationary Serial Updates

We describe similar walk-sum diagrams for the serial update scheme of Section III-B. Since there is no link failure, $r_n(* \to \bullet) = n-1$. The recursive walk-set update (18) can be specialized as follows:

$$\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t) \bigcup \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t),\ s \in V, t \in V_n. \quad (24)$$

While (23) is a specialization to iterations with parallel updates, (24) is relevant for serial updates. The GS iteration discussed in Section IV-B falls in this class, as do more general serial updates described in Section III-B in which we update a subset of variables $V_n$ based on a subgraph of the induced graph of $V_n$. The right-hand side of Figure 2 illustrates an example for our 3-node model. We show the subgraphs $\mathcal{S}_n$ used in the first four stages of the algorithm and the corresponding 4-level walk-sum diagram. Note that at iteration 2 we update variables $x_2$ and $x_3$ without taking into account the edge connecting them. Indeed, the updates at the first four iterations of this example include block GS, a hybrid of ET and block GS, parallel ET, and GS, respectively.

### E. Failing links

We now discuss the general non-stationary update scheme of Section III-C involving failing links. The recursive walk-set computation equations for this iteration are given by (18−20). Figure 3 shows the subgraph and the edges in $\kappa_n$ that fail at each iteration, and the corresponding 4-level walk-sum diagram. We elaborate on the computation and propagation of information at each iteration. At iteration 1, inference is performed using subgraph $\mathcal{S}_1$, followed by nodes 1 and 2 passing a message to each other according to the post-inference message-passing rule (16). At iteration 2 only $x_3$ is updated. As no links fail, node 3 gets information from nodes 1 and 2 at level 1. At iteration 3, the link $(2,1)$ fails. But node 1 has information about $x_2$ at level 1 (due to the post-inference message passing step from iteration 1). This information is used from the local memory at node 1 in (17), and is represented by the arrow from node 2 at level 1 to node 1 at level 3. At iteration 4, the links $(1,3)$ and $(3,1)$ fail. Similar reasoning as in iteration 3 applies to the arrows drawn across multiple levels from node 1 to node 3, and from node 3 to node 1. Further, post-inference message-passing at this iteration only takes place between nodes 1 and 2 because the only edge in $\mathcal{S}_4$ is $\{1,2\}$.

## V. Convergence Analysis

We now show that all the algorithms of Section III converge in walk-summable models. As in Section IV-A, we focus on the most general non-stationary algorithm with failing links of Section III-C. We begin by showing that $\widehat{x}^{(n)}$ converges to the correct means when $\widehat{x}^{(0)} = 0$. Next, we use this result to show that we can achieve convergence to the correct means for any initial guess $\widehat{x}^{(0)}$.

The proof that $\phi_n(h; * \to t) \to \left(J^{-1}h\right)_t$ as $n \to \infty$ relies on the fact that $\mathcal{W}_n(s \to t)$ eventually contains every element of the set $\mathcal{W}(s \to t)$ of all the walks in $\mathcal{G}$ from $s$ to $t$, a condition we refer to as *completeness*. Showing this begins with the following proposition proved in Appendix C.

**Proposition 3:** (*Nesting*) The walk-sets defined in equations (18−20) are nested, i.e. for every pair of vertices $s,t \in V$, $\mathcal{W}_{n-1}(s \to t) \subseteq \mathcal{W}_n(s \to t)$ for each $n$.

This statement is easily seen for a stationary ET algorithm because the walk-sum diagram $\mathcal{G}^{(n)}$ from levels 2 to $n$ is a replica of $\mathcal{G}^{(n-1)}$ (for example, the GJ diagram in Figure 1). However, the proposition is less clear for non-stationary iterations. The discussion in Section IV-C illustrates this point; the paths that a walk traverses change drastically depending on the level in the walk-sum diagram at which the walk ends. Nonetheless, as shown in Appendix C, the structure of the estimation algorithms that we consider ensures that whenever a walk is not explicitly captured in the same form it appeared in the preceding iteration, it is recovered through a different path in the subsequent walk-sum diagram (no walks are lost).

Completeness relies on both nesting and the following additional condition.

**Definition 2:** Let $(u,v)$ be any directed edge in $\mathcal{G}$. For each $n$, let $\kappa_n^{active} \subseteq \kappa_n$ denote the set of directed active edges (links that do not fail) in $\kappa_n$ at iteration $n$. The edge $(u,v)$ is said to be *updated infinitely often*[5] if for every $N \geq 0$, there exists an $m > N$ such that $(u,v) \in \mathcal{S}_m \cup \kappa_m^{active}$.

If there is no link failure, this definition reduces to including each vertex in $V$ in the update set $V_n$ infinitely often.

[5]If $\mathcal{G}$ contains a singleton node, then this node must be updated at least once.

For parallel non-stationary ET iterations (Section III-A), this property is satisfied for *any sequence of subgraphs*. Note that there are cases in which inference algorithms may not have to traverse each edge infinitely often. For instance, suppose that $\mathcal{G}$ can be decomposed into subgraphs $\mathcal{G}_1$ and $\mathcal{G}_2$ that are connected by a single edge, with $\mathcal{G}_2$ having small size so that we can perform exact computations. For example, $\mathcal{G}_2$ could be a leaf node (i.e., have degree one). We can eliminate the variables in $\mathcal{G}_2$, propagate information "into" $\mathcal{G}_1$ along the single connecting edge, perform inference within $\mathcal{G}_1$, and then back-substitute. Hence, the single connecting edge is traversed only finitely often. In this case the hard part of the overall inference procedure is on the reduced graph with leaves and small, dangling subgraphs eliminated, and we focus on inference problems on such graphs. Thus, we assume that each vertex in $\mathcal{G}$ has degree at least two and study algorithms that traverse each edge infinitely often.

**Proposition 4:** (*Completeness*) Let $w = s \cdots t$ be an arbitrary walk from $s$ to $t$ in $\mathcal{G}$. If every edge in $\mathcal{G}$ is updated infinitely often (in both directions), then there exists an $N$ such that $w \in \mathcal{W}_n(s \to t)$ for all $n \geq N$, where the walk-set $\mathcal{W}_n(s \to t)$ is defined in (18−20).

The proof of this proposition appears in Appendix D. We can now state and prove the following.

**Theorem 1:** If every edge in $\mathcal{G}$ is updated infinitely often (in both directions), then $\phi_n(h; * \to t) \to \left(J^{-1}h\right)_t$ as $n \to \infty$ in walk-summable models, with $\phi_n(s \to t)$ as defined in Section IV-A.

**Proof**: One can check that $\mathcal{W}_n(s \to t) \subseteq \mathcal{W}(s \to t), \forall n$. This is because equations (18−20) only use edges from the original graph $\mathcal{G}$. We have from Proposition 4 that every walk from $s$ to $t$ in $\mathcal{G}$ is eventually contained in $\mathcal{W}_n(s \to t)$. Thus, $\cup_{n=0}^{\infty} \mathcal{W}_n(s \to t) = \mathcal{W}(s \to t)$. Given these arguments and the nesting of the walk-sets $\mathcal{W}_n(s \to t)$ from Proposition 3, we can appeal to the results in Section II-C to conclude that $\phi_n(h; * \to t) \to \left(J^{-1}h\right)_t$ as $n \to \infty$. $\square$

Theorem 1 shows that $\widehat{x}_t^{(n)} \to \left(J^{-1}h\right)_t$ for $\widehat{x}^{(0)} = 0$. The following result, proven in Appendix E, shows that in walk-summable models convergence is achieved for any choice of initial condition.[6]

**Theorem 2:** If every edge is updated infinitely often, then $\widehat{x}^{(n)}$ computed according to (15,17) converges to the correct means in walk-summable models for any initial guess $\widehat{x}^{(0)}$.

Next, we describe a stronger convergence result when there is no communication failure.

**Corollary 1:** Assuming no communication failure, we have the following special cases for convergence in walk-summable models (with any initial guess):

1) The hybrid algorithms of Section III-B converge to the correct mean as long as each variable is updated infinitely often.
2) The parallel ET iterations of Section III-A (with $V_n = V$) converge to the correct mean using *any* sequence of subgraphs.

[6]Note that in this case the messages must be initialized as $M(s \to t) = R_{t,s}\ \widehat{x}_s^{(0)}$ for each directed edge $(s, t) \in \mathcal{E}$.

We note that in the parallel ET iteration (with no link failure), it is *not* necessary to include each edge in some subgraph $\mathcal{S}_n$; indeed, even stationary algorithms that use the same subgraph at each iteration are guaranteed to converge.

Theorem 2 shows that walk-summability is a *sufficient* condition for *all* our algorithms to converge for a very large and flexible set of sequences of tractable subgraphs or subsets of variables (ones that update each edge infinitely often) on which to perform successive updates. Corollary 1 requires even weaker conditions for convergence if there is no communication failure. The following result, proven in Appendix F, shows that walk-summability is also *necessary* for this complete flexibility. Thus, while any of our algorithms *may* converge for some sequence of subsets of variables and tractable subgraphs in a non-walk-summable model, there is at least one sequence of updates that leads to a divergent iteration.

**Theorem 3:** For any non-walk-summable model, there exists at least one sequence of iterative steps that is ill-posed, or for which $\widehat{x}^{(n)}$, computed according to (15,17), diverges.

## VI. Adaptive Iterations and Experimental Results

In this section we address two topics. The first is taking advantage of the great flexibility in choosing successive iterative steps by developing techniques that adaptively optimize the on-line choice of the next tree or subset of variables to use in order to reduce the error as quickly as possible. The second is providing experimental results that demonstrate the convergence behavior of these adaptive algorithms.

### *A. Choosing trees and subsets of variables adaptively*

At iteration $n$, let the *error* be $e^{(n)} = \widehat{x} - \widehat{x}^{(n)}$ and the *residual error* be $h^{(n)} = h - J\ \widehat{x}^{(n)}$. Note that it is tractable to compute the residual error at each iteration.

*A.1 Trees.* We describe an efficient algorithm to choose spanning trees adaptively to use as preconditioners in the ET algorithm of Section III-A. We have the following relationship between the error at iteration $n$ and the residual error at iteration $n-1$:

$$e^{(n)} = (J^{-1} - J_{\mathcal{S}_n}^{-1})\ h^{(n-1)}.$$

Based on this relationship, we have the walk-sum interpretation $e_s^{(n)} = \phi(h^{(n-1)}; * \xrightarrow{\mathcal{G} \backslash \mathcal{S}_n} s)$, and consequently the following bound on the $\ell_1$ norm of $e^{(n)}$:

$$\begin{aligned} \|e^{(n)}\|_{\ell_1} &= \sum_{s \in V} \left| \phi(h^{(n-1)}; * \xrightarrow{\mathcal{G} \backslash \mathcal{S}_n} s) \right| \\ &\leq \bar{\phi}(|h^{(n-1)}|; \mathcal{G} \backslash \mathcal{S}_n) \\ &= \bar{\phi}(|h^{(n-1)}|; \mathcal{G}) - \bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n), \quad (25) \end{aligned}$$

where $\mathcal{G} \backslash \mathcal{S}_n$ denotes walks in $\mathcal{G}$ that must traverse edges not in $\mathcal{S}_n$, $|h^{(n-1)}|$ refers to the entry-wise absolute value vector of $h^{(n-1)}$, $\bar{\phi}(|h^{(n-1)}|; \mathcal{G})$ refers to the re-weighted absolute walk-sum over all walks in $\mathcal{G}$, and $\bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n)$ refers to the re-weighted absolute walk-sum over all walks in $\mathcal{S}_n$. Minimizing the error $e^{(n)}$ reduces to choosing $\mathcal{S}_n$ to *maximize*

$\bar{\phi}(|h^{(n-1)}|;\mathcal{S}_n)$. Hence, if we maximize among all trees, we have the following *maximum walk-sum tree* problem:

$$\arg\max_{\mathcal{S}_n \text{ a tree}} \quad \bar{\phi}(|h^{(n-1)}|;\mathcal{S}_n). \tag{26}$$

Rather than solving this combinatorially complex problem, we instead solve a problem that minimizes a looser upper bound than (25). Specifically, consider any edge $\{u,v\}\in\mathcal{E}$ and all of the walks $\mathcal{S}(u,v)=(uv,vu,uvu,vuv,uvuv,vuvu,\ldots)$ that live solely on this single edge. It is not difficult to show that

$$\begin{aligned} w_{u,v} &\triangleq \bar{\phi}(|h^{(n-1)}|;\mathcal{S}(u,v)) \\ &= (|h_u^{(n-1)}|+|h_v^{(n-1)}|)\sum_{\ell=1}^{\infty}|R_{u,v}|^{\ell} \\ &= (|h_u^{(n-1)}|+|h_v^{(n-1)}|)\frac{|R_{u,v}|}{1-|R_{u,v}|}. \end{aligned} \tag{27}$$

This weight provides a measure of the error-reduction capacity of edge $\{u,v\}$ by itself at iteration $n$. This leads directly to choosing the *maximum spanning tree* [28] by solving

$$\arg\max_{\mathcal{S}_n \text{ a tree}} \sum_{\{u,v\}\in\mathcal{S}_n} w_{u,v}. \tag{28}$$

For any tree $\mathcal{S}_n$ the set of walks captured in the sum in (28) is a subset of all the walks in $\mathcal{S}_n$, so that solving (28) provides a lower bound on (26) and thus a looser upper bound than (25). Each iteration of this method can be solved using $\mathcal{O}(|\mathcal{E}|\log|V|)$ computations based on standard greedy approaches for the maximum spanning tree problem [28]. For sparse graphs, more sophisticated variants can also be used to achieve a per-iteration complexity of $\mathcal{O}(|\mathcal{E}|\log\log|V|)$ [28].

*A.2 Subsets of variables.* We present an algorithm to choose the next best subset of $k$ variables for the block GS algorithm of Section III-B. The error at iteration $n$ can be written as follows:

$$\begin{aligned} e_{V_n}^{(n)} &= \widehat{x}_{V_n}-\widehat{x}_{V_n}^{(n)} = {J^{(n)}}^{-1}R_{V_n,V_n^c}[J^{-1}\ h^{(n-1)}]_{V_n^c}, \\ e_{V_n^c}^{(n)} &= \widehat{x}_{V_n^c}-\widehat{x}_{V_n^c}^{(n)} = e_{V_n^c}^{(n-1)} = [J^{-1}\ h^{(n-1)}]_{V_n^c}. \end{aligned}$$

As with (25), we have the following upper bound:

$$\begin{aligned} \|e^{(n)}\|_{\ell_1} &= \|e_{V_n}^{(n)}\|_{\ell_1}+\|e_{V_n^c}^{(n)}\|_{\ell_1} \\ &\leq \left[\bar{\phi}(|h^{(n-1)}|;*\xrightarrow{\mathcal{G}}V_n)-\bar{\phi}(|h^{(n-1)}|;V_n\xrightarrow{\mathcal{E}(V_n)}V_n)\right] \\ &\quad + \bar{\phi}(|h^{(n-1)}|;*\xrightarrow{\mathcal{G}}V_n^c) \\ &= \bar{\phi}(|h^{(n-1)}|;\mathcal{G}) \ - \ \bar{\phi}(|h^{(n-1)}|;V_n\xrightarrow{\mathcal{E}(V_n)}V_n), \end{aligned} \tag{29}$$

where $\mathcal{E}(V_n)$ refers to the edges in the induced subgraph of $V_n$. Minimizing this upper bound reduces to solving the following *maximum walk-sum block* problem:

$$\arg\max_{|V_n|\leq k} \quad \bar{\phi}(|h^{(n-1)}|;V_n\xrightarrow{\mathcal{E}(V_n)}V_n). \tag{30}$$

As with the maximum walk-sum tree problem, finding the optimal such block directly is combinatorially complex. Therefore, we consider the following relaxed maximum walk-sum block problem based on single-edge walks:

$$\arg\max_{|V_n|\leq k} \quad \bar{\phi}(|h^{(n-1)}|;V_n\xrightarrow{1e}V_n), \tag{31}$$

where $\xrightarrow{1e}$ denotes the restriction that walks can traverse at most one edge. The walks in (31) are a subset of the walks in (30). Thus, solving (31) provides a lower bound on (30), hence minimizing a looser upper bound on the error than (29).

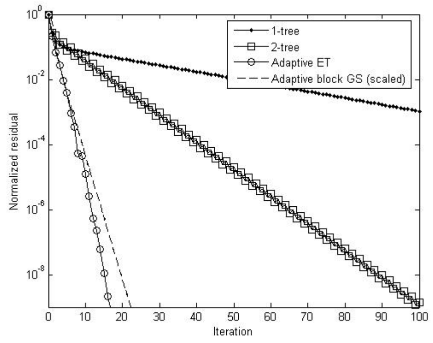
Solving (31) is also combinatorially complex; therefore, we use a greedy method for an approximate solution:

1) Set $V_n=\emptyset$. Assuming that the goal is to solve the problem for $k=1$, compute node weights
$$w_u=|h_u^{(n-1)}|,$$
based on the walks captured by (31) if node $u$ were to be included in $V_n$.
2) Find the maximum weight node $u^*$ from $V\backslash V_n$, and set $V_n\leftarrow V_n\cup u^*$.
3) If $|V_n|=k$, stop. Otherwise, update each neighbor $v\in V\backslash V_n$ of $u^*$ and go to step 2:
$$w_v\leftarrow w_v+\left(|h_{u^*}^{(n-1)}|+|h_v^{(n-1)}|\right)\frac{|R_{u^*,v}|}{1-|R_{u^*,v}|}.$$
This update captures the extra walks in (31) if $v$ were to be added to $V_n$.

Step 3 is the greedy aspect of the algorithm as it updates weights by computing the extra walks that would be captured in (31) if node $v$ were added to $V_n$, with the assumption that the nodes already in $V_n$ remain unchanged. Note that only the weights of the neighbors of $u^*$ are updated in step 3; thus, there is a bias towards choosing a connected block. In choosing successive blocks in this way, we collect walks adaptively without explicit regard for the objective of updating each node infinitely often. However, our method is biased towards choosing variables that have not been updated for a few iterations as the residual error of such variables becomes larger relative to the other variables. Indeed, empirical evidence confirms this behavior with all the simulations leading to convergent iterations. For $k$ bounded, each iteration using this technique requires $\mathcal{O}(k\log|\mathcal{E}|)$ computations using an efficient sort data structure [28]. The space complexity of maintaining such a structure can be significant compared to the adaptive ET procedure.

*A.3 Experimental Illustration.* We test the preceding two adaptive algorithms on randomly generated 15 x 15 nearest-neighbor grid models with[7] $\varrho(\bar{R})=0.99$, and with $\widehat{x}^{(0)}=0$. The blocks used in block GS were of size $k=5$. We compare these adaptive methods to standard non-adaptive one-tree and two-tree ET iterations [13]. Figure 4 shows the performance of these algorithms. The plot shows the relative decrease in the normalized residual error $\frac{\|h^{(n)}\|_{\ell_2}}{\|h^{(0)}\|_{\ell_2}}$ versus the number of iterations. The table shows the average number of iterations required for these algorithms to reduce the normalized residual error below $10^{-10}$. The average was computed based on the performance on $100$ randomly generated models. All these models are poorly conditioned because they are barely walk-summable. The number of iterations for block

[7]The grid edge weights are chosen uniformly at random from $[-1,1]$. The matrix $R$ is then scaled so that $\varrho(\bar{R})=0.99$. The potential vector $h$ is chosen to be the all-ones vector.



| Method | Avg. iterations |
|---:|---:|
| One-tree | 143.07 |
| Two-tree | 102.70 |
| Adaptive ET | 44.04 |
| Adaptive block GS (scaled) | 41.81 |

Fig. 4. (Left) Convergence results for a randomly generated 15 x 15 nearest-neighbor grid-structured model. (Right) Average number of iterations required for the normalized residual to reduce by a factor of $10^{-10}$ over 100 randomly generated models.

GS is scaled appropriately to account for the different per-iteration computational costs ($\mathcal{O}(|\mathcal{E}| \log\log|V|)$ for adaptive ET and $\mathcal{O}(k \log|\mathcal{E}|)$ for adaptive block GS). The one-tree ET method uses a spanning tree obtained by removing all the vertical edges except the middle column. The two-tree method alternates between this tree and its rotation (obtained by removing all the horizontal edges except the middle row).

Both the adaptive ET and block GS algorithms provide far faster convergence compared to the one-tree and two-tree iterations, thus providing a computationally attractive method for estimation in the broad class of walk-summable models.

## *B. Communication Failure: Experimental Illustration*

To illustrate our adaptive methods in the context of communication failure, we consider a simple model for a distributed sensor network in which links (edges) fail independently with failure probability $\alpha$, and each failed link remains inactive for a certain number of iterations given by a geometric random variable with mean $\frac{1}{\beta}$. At each iteration, we find the best spanning tree (or forest) among the active links using the approach described in Section VI-A.1. The maximum spanning tree problem can be solved in a distributed manner using the algorithms presented in [29, 30]. Figure 5 shows the convergence of our memory-based algorithm from Section III-C on the same randomly generated 15 x 15 grid model used to generate the plot in Figure 4 (again, with $\widehat{x}^{(0)} = 0$). The different curves are obtained by varying $\alpha$ and $\beta$. As expected, the first plot shows that our algorithm is slower to converge as the failure probability $\alpha$ increases, while the second plot shows that convergence is faster as $\beta$ is increased (which decreases the average inactive time). These results show that our adaptive algorithms provide a scalable, flexible, and convergent method for the estimation problem in a distributed setting with communication failure.

## VII. Conclusion

In this paper, we have described and analyzed a rich set of algorithms for estimation in Gaussian graphical models with arbitrary structure. These algorithms are iterative in nature and involve a sequence of inference problems on tractable subgraphs over subsets of variables. Our framework includes parallel iterations such as ET, in which inference is performed on a tractable subgraph of the whole graph at each iteration, and serial iterations such as block GS, in which the induced subgraph of a small subset of variables is directly inverted at each iteration. We describe hybrid versions of these algorithms that involve inference on a subgraph of a subset of variables. We also discuss a method that uses local memory at each node to overcome temporary communication failures that may arise in distributed sensor networks. Our memory-based framework applies to the non-stationary ET, block GS, and hybrid algorithms.

We analyze these algorithms based on the recently introduced walk-sum interpretation of Gaussian inference. A salient feature in our analysis is the development of walk-sum diagrams. These graphs correspond exactly to the walks computed after each iteration, and provide an intuitive graphical comparison between the various algorithms. This walk-sum analysis allows us to conclude that for the broad class of walk-summable models, our algorithms converge for a very large and flexible set of sequences of subgraphs and subsets of variables used. We also describe how this flexibility can be exploited by formulating efficient algorithms that choose spanning trees and subsets of variables adaptively at each iteration. These algorithms are then used in the ET and block GS iterations respectively to demonstrate that significantly faster convergence can be obtained using these methods over traditional one-tree and two-tree ET iterations.

Our adaptive algorithms are greedy in that they consider the effect of edges individually (by considering walk-sums concentrated on single edges). A strength of our analysis for the case of finding the “next best” tree is that we do obtain an upper bound on the resulting error, and hence on the possible gap between our greedy procedure and the truly optimal one. Obtaining tighter error bounds, or conditions on graphical models under which our choice of tree yields near-optimal solutions is an open problem. Another interesting question is the development of general versions of the maximum walk-sum tree and maximum walk-sum block algorithms that choose the $K$ next best trees or blocks jointly in order to achieve maximum reduction in error after $K$ iterations. For applications involving communication failure, extending our adaptive algorithms in a principled manner to explicitly avoid

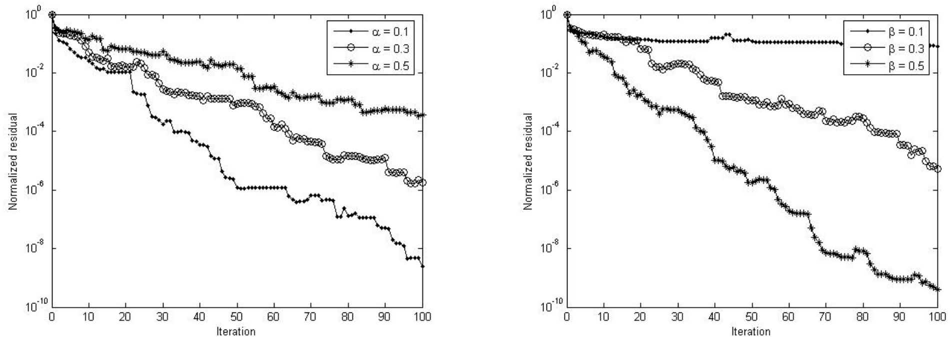


Fig. 5. Convergence of memory-based algorithm on same randomly generated 15 x 15 used in Figure 4: Varying $\alpha$ with $\beta = 0.3$ (left) and varying $\beta$ with $\alpha = 0.3$ (right).

failed links while optimizing the rate of convergence is an important problem. Finally, the fundamental principle of solving a sequence of tractable inference problems on subgraphs has been exploited for non-Gaussian inference problems (e.g. [12]), and extending our methods to this more general setting is of clear interest.

## Acknowledgements

The authors would like to thank Dr. Erik Sudderth for helpful discussions and comments on early drafts of this work.

## Appendix

### A. Dealing with un-normalized models

Consider an information matrix $J = D - H$ (where $D$ is the diagonal part of $J$) that is not normalized, i.e. $D \neq I$. The weight of a walk $w = \{w_i\}_{i=0}^{\ell}$ can be re-defined as follows:

$$\begin{aligned}\psi(w) &= \frac{\prod_{i=0}^{\ell-1} H_{w_i,w_{i+1}}}{\prod_{i=0}^{\ell} D_{w_i,w_i}} \\ &= \frac{\prod_{i=0}^{\ell-1} \sqrt{D_{w_i,w_i}} R_{w_i,w_{i+1}} \sqrt{D_{w_{i+1},w_{i+1}}}}{\prod_{i=0}^{\ell} D_{w_i,w_i}} \\ &= \frac{\phi(w)}{\sqrt{D_{w_0,w_0} D_{w_\ell,w_\ell}}},\end{aligned}$$

where $\psi(w)$ is the weight of $w$ with respect to the un-normalized model, and $\phi(w)$ is the weight of $w$ in the corresponding normalized model. We can then define walk-summability in terms of the absolute convergence of the un-normalized walk-sum $\bar{\psi}(s \to t)$ over all walks from $s$ to $t$ (for each pair of vertices $s, t \in V$). A necessary and sufficient condition for this un-normalized notion of walk-summability is $\varrho\left(D^{-\frac{1}{2}}\ H\ D^{-\frac{1}{2}}\right) < 1$, which is equivalent to the original condition $\varrho(\bar{R}) < 1$ in the corresponding normalized model. Un-normalized versions of the algorithms in Section III can be constructed by replacing every occurrence of the partial correlation matrix $R$ by the un-normalized off-diagonal matrix $H$. The rest of our analysis and convergence results remain unchanged because we deal with abstract walk-sets. (Note that in the proof of Proposition 1, every occurrence of $R$ must be changed to $H$.) Alternatively, given an un-normalized model, one can first normalize the model ($J_{norm} \leftarrow D^{-\frac{1}{2}}\ J_{unnorm}\ D^{-\frac{1}{2}}, h_{norm} \leftarrow D^{-\frac{1}{2}}\ h_{unnorm}$), then apply the algorithms of Section III, and finally "de-normalize" the resulting estimate ($\widehat{x}_{unnorm}^{(n)} \leftarrow D^{-\frac{1}{2}}\ \widehat{x}_{norm}^{(n)}$). Such a procedure would provide the same estimate as the direct application of the un-normalized versions of the algorithms in Section III as outlined above.

### B. Proof of Proposition 1

**Remarks**: Before proceeding with the proof of the proposition, we make some observations about the walk-sets $\mathcal{W}_n(s \to t)$ that will prove useful for the other proofs as well. For $t \in V_n$, notice that since the set of edges contained in $\mathcal{S}_n$ and $\kappa_n$ are disjoint, the walk-sets $\mathcal{W}(s \xrightarrow{\mathcal{S}_n} t)$ and $\mathcal{W}_{r_n(*\to\bullet)}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t)$ are disjoint. Therefore, from Section II-C,

$$\begin{aligned}\phi_n(s \to t) &= \phi(s \xrightarrow{\mathcal{S}_n} t) \\ &\quad + \phi\left(\mathcal{W}_{r_n(*\to\bullet)}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t)\right) \\ &= \phi(s \xrightarrow{\mathcal{S}_n} t) \\ &\quad + \phi\left(\bigcup_{u,v \in V} \mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)\right).\end{aligned} \tag{32}$$

Every walk $w \in \mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$ can be *decomposed uniquely* as $w = w_a \cdot w_b \cdot w_c$, where $w_a \in \mathcal{W}_{r_n(u\to v)}(s \to u)$, $w_b \in \mathcal{W}(u \xrightarrow{\kappa_n(1)} v)$, and $w_c \in \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$. The unique decomposition property is a consequence of $\mathcal{S}_n$ and $\kappa_n$ being disjoint, and the walk in $\kappa_n$ being restricted to a length-1 hop. This property also implies that $\mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$ and $\mathcal{W}_{r_n(u'\to v')}(s \to u') \otimes \mathcal{W}(u' \xrightarrow{\kappa_n(1)} v') \otimes \mathcal{W}(v' \xrightarrow{\mathcal{S}_n} t)$ are disjoint if $(u, v) \neq (u', v')$. Based on these two observations, we have from Section II-C that

$$\phi\left(\bigcup_{u,v\in V}\mathcal{W}_{r_n(u\to v)}(s\to u)\otimes\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)\otimes\mathcal{W}(v\xrightarrow{\mathcal{S}_n}t)\right)$$
$$=\sum_{u,v\in V}\phi\Big(\mathcal{W}_{r_n(u\to v)}(s\to u)\otimes\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)\otimes\mathcal{W}(v\xrightarrow{\mathcal{S}_n}t)\Big)$$
$$=\sum_{u,v\in V}\phi_{r_n(u\to v)}(s\to u)\ \phi(u\xrightarrow{\kappa_n(1)}v)\ \phi(v\xrightarrow{\mathcal{S}_n}t). \qquad (33)$$

**Proof of proposition**: We provide an inductive proof. From (20), $\phi_0(s\to t)=0$. Thus,

$$\phi_0(h;*\to t)=\sum_{s\in V}h_s\ \phi_0(s\to t)=0=\widehat{x}_t^{(0)},$$

which is consistent with the proposition because we assume that our initial guess is 0 at each node.

Assume that $\widehat{x}_t^{(n')}=\phi_{n'}(h;*\to t)$, for $0\le n'\le n-1$, as the inductive hypothesis. For $t\in V_n^c$,

$$\widehat{x}_t^{(n)}=\widehat{x}_t^{(n-1)}=\phi_{n-1}(h;*\to t)=\phi_n(h;*\to t),$$

where the first equality is from (12), the second from the inductive hypothesis, and the third from (19). Hence, we can focus on nodes in $V_n$. For $t\in V_n$, (32−33) can be re-written as:

$$\begin{aligned}\phi_n(s\to t)\ &=\ \phi(s\xrightarrow{\mathcal{S}_n}t)\\ &+\sum_{(u,v)\in\kappa_n}\phi_{r_n(u\to v)}(s\to u)\phi(u\xrightarrow{\kappa_n(1)}v)\phi(v\xrightarrow{\mathcal{S}_n}t),\end{aligned} \qquad (34)$$

because $\phi(u\xrightarrow{\kappa_n(1)}v)=0$ if $(u,v)\notin\kappa_n$. From (32−34) we have that:

$$\begin{aligned}\phi_n(h;*\to t)\ &=\ \sum_{s\in V}h_s\phi(s\xrightarrow{\mathcal{S}_n}t)\\ &+\sum_{s\in V}h_s\sum_{(u,v)\in\kappa_n}\phi_{r_n(u\to v)}(s\to u)\phi(u\xrightarrow{\kappa_n(1)}v)\phi(v\xrightarrow{\mathcal{S}_n}t)\\ &=\ \sum_{s\in V}h_s(J_{\mathcal{S}_n}^{-1})_{t,s}\\ &+\sum_{s\in V}h_s\sum_{(u,v)\in\kappa_n}\phi_{r_n(u\to v)}(s\to u)\ R_{v,u}\ (J_{\mathcal{S}_n}^{-1})_{t,v},\end{aligned}$$

where we have used the walk-sum interpretation of $J_{\mathcal{S}_n}^{-1}$ and $\kappa_n$. Simplifying further, we have that

$$\begin{aligned}\phi_n(h;*\to t)\ &=\ \left(J_{\mathcal{S}_n}^{-1}\ h_{V_n}\right)_t\\ &+\sum_{(u,v)\in\kappa_n}\phi_{r_n(u\to v)}(h;*\to u)\ R_{v,u}\ (J_{\mathcal{S}_n}^{-1})_{t,v}\\ &=\ \left(J_{\mathcal{S}_n}^{-1}\ h_{V_n}\right)_t\\ &+\sum_{(u,v)\in\kappa_n}\widehat{x}_u^{r_n(u\to v)}\ R_{v,u}\ (J_{\mathcal{S}_n}^{-1})_{t,v}.\end{aligned} \qquad (35)$$

The last equality is from the inductive hypothesis because $0\le r_n(u\to v)\le n-1$. Next, we have that

$$\begin{aligned}\phi_n(h;*\to t)\ &=\ \left(J_{\mathcal{S}_n}^{-1}\ h_{V_n}\right)_t\\ &+\sum_{v\in V_n}(J_{\mathcal{S}_n}^{-1})_{t,v}\sum_{\{u|(u,v)\in\kappa_n\}}R_{v,u}\ \widehat{x}_u^{r_n(u\to v)}\\ &=\ \left(J_{\mathcal{S}_n}^{-1}\ h_{V_n}\right)_t+\sum_{v\in V_n}(J_{\mathcal{S}_n}^{-1})_{t,v}\ M_n(v)\\ &=\ \widehat{x}_t^{(n)},\end{aligned}$$

where the second equality is from (17), and the third from (15). □

### C. Proof of Proposition 3

We prove the following lemma that will be useful later for the proof of the proposition.

**Lemma 2:** Let $w=w_{start}\cdots p\cdot q\cdots w_{end}$ be an arbitrary walk in $\mathcal{W}_n(w_{start}\to w_{end})$, and let $\widetilde{w}=w_{start}\cdots p$ be a *leading* sub-walk of $w$. There exists a $k_n\le n$ with $\widetilde{w}\in\mathcal{W}_{k_n}(w_{start}\to p)$ so that at least one of the following conditions is true: $k_n=n$ and the edge $(p,q)\in\mathcal{S}_n$, or $k_n\le r_n(p\to q)$.

**Proof**: The base case is vacuously true because $\mathcal{W}_0(w_{start}\to w_{end})=\emptyset$. For the inductive hypothesis, assume that the statement is true for $0\le n'\le n-1$. This can be used to prove the statement if $w_{end}\in V_n^c$. Assume that $w\in\mathcal{W}_n(w_{start}\to w_{end})$ with $w_{end}\in V_n$. From the remarks in Appendix B, either $w\in\mathcal{W}(w_{start}\xrightarrow{\mathcal{S}_n}w_{end})$, or $w\in\mathcal{W}_{r_n(u\to v)}(w_{start}\to u)\otimes\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)\otimes\mathcal{W}(v\xrightarrow{\mathcal{S}_n}w_{end})$ for some unique pair of vertices $u,v\in V$ with $r_n(u\to v)\le n-1$. If $w\in\mathcal{W}(w_{start}\xrightarrow{\mathcal{S}_n}w_{end})$, then $k_n=n$ and $(p,q)\in\mathcal{S}_n$.

If $w\in\mathcal{W}_{r_n(u\to v)}(w_{start}\to u)\otimes\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)\otimes\mathcal{W}(v\xrightarrow{\mathcal{S}_n}w_{end})$, then from the remarks in Appendix B, $w$ can be uniquely decomposed as $w=w_a\cdot w_b\cdot w_c$ with $w_a\in\mathcal{W}_{r_n(u\to v)}(w_{start}\to u)$, $w_b=uv\in\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)$, and $w_c\in\mathcal{W}(v\xrightarrow{\mathcal{S}_n}w_{end})$. Suppose the trailing part $p\cdots w_{end}$ is a sub-walk of $w_c$, or is equal to $w_c$. We can uniquely decompose $\widetilde{w}$ as $w_a\cdot w_b\cdot(v\cdots p)\in\mathcal{W}_{r_n(u\to v)}(w_{start}\to u)\otimes\mathcal{W}(u\xrightarrow{\kappa_n(1)}v)\otimes\mathcal{W}(v\xrightarrow{\mathcal{S}_n}p)$. This shows that $k_n=n$. Also, $(p,q)\in\mathcal{S}_n$ because $w_c\in\mathcal{W}(v\xrightarrow{\mathcal{S}_n}w_{end})$.

Suppose $p\cdots w_{end}$ is not a sub-walk of $w_c$; then either $\widetilde{w}=w_a$ or $\widetilde{w}$ must be a leading sub-walk of $w_a$. If $\widetilde{w}=w_a$, then $(p,q)=(u,v)$ and $k_n=r_n(p\to q)$. If $\widetilde{w}$ is a leading sub-walk of $w_a$, we can use the inductive hypothesis (because $r_n(u\to v)\le n-1$) to obtain a $k_n=k_{r_n(u\to v)}\le r_n(u\to v)<n$. If $k_n=k_{r_n(u\to v)}=r_n(u\to v)$, then $(p,q)\in\mathcal{S}_{r_n(u\to v)}$ and one can check that $r_n(p\to q)\ge k_n$ (because a post-inference message is passed on edge $(p,q)$ at iteration $r_n(u\to v)=k_n$). Otherwise, $k_n=k_{r_n(u\to v)}\le r_{r_n(u\to v)}(p\to q)\le r_n(p\to q)$. □

**Proof of proposition**: We provide an inductive proof. Let any two vertices $s,t\in V$ be given. The base case $\mathcal{W}_0(s\to t)\subseteq\mathcal{W}_1(s\to t)$ clearly follows from the fact that $\mathcal{W}_0(s\to t)=\emptyset$ from (18). For the inductive hypothesis, assume that $\mathcal{W}_{n'-1}(s\to t)\subseteq\mathcal{W}_{n'}(s\to t)$ for $0<n'\le n-1$. If $t\in V_n^c$,

the proposition follows because $\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to t)$ from (19). So we can restrict ourselves to the case that $t \in V_n$. Let some $w \in \mathcal{W}_{n-1}(s \to t)$ be given.

First, we check if $w \in \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t)$. If this is the case, then we are done. If not, $w$ can be uniquely decomposed as $w = w_a \cdot w_b \cdot w_c$, where $w_b \in \mathcal{W}(p \xrightarrow{\kappa_n(1)} q)$, and $w_c \in \mathcal{W}(q \xrightarrow{\mathcal{S}_n} t)$ for some $p, q \in V$. We must show that $w_a \in \mathcal{W}_{r_n(p\to q)}(s \to p)$. But $w_a$ is a leading sub-walk of $w$. We have from Lemma 2 that, with respect to the walk-set $\mathcal{W}_{n-1}(s \to t)$, there exists a $k_{n-1} \le n-1$ such that $w_a \in \mathcal{W}_{k_{n-1}}(s \to p)$. If $k_{n-1} = n-1$, then $(p,q) \in \mathcal{S}_{n-1}$ and $r_n(p \to q) = n-1$ (due to post-inference message (16)). Hence, $w_a \in \mathcal{W}_{k_{n-1}}(s \to p) = \mathcal{W}_{r_n(p\to q)}(s \to p)$. If $k_{n-1} < n-1$, then $k_{n-1} \le r_{n-1}(p \to q)$ from Lemma 2. But $k_{n-1} \le r_{n-1}(p \to q) \le r_n(p \to q) \le n-1$ and we can apply the inductive hypothesis to show the relation $\mathcal{W}_{k_{n-1}}(s \to p) \subseteq \mathcal{W}_{r_n(p\to q)}(s \to p)$. Thus, $w_a \in \mathcal{W}_{k_{n-1}}(s \to p) \subseteq \mathcal{W}_{r_n(p\to q)}(s \to p)$. □

## D. Proof of Proposition 4

Let $w = s \cdots u \cdot t$. We provide an inductive proof with respect to the length of $w$. If every edge is updated infinitely often, it is clear that every node is updated infinitely often. Therefore, the leading length-0 part $(s)$ is computed when $s$ is first updated at some iteration $k$. By the nesting of the walk-sets $\mathcal{W}_n$ from Proposition 3, we have that $(s) \in \mathcal{W}_{k'}(s \to s)$ for all $k' \ge k$. Now assume (as the inductive hypothesis) that the leading sub-walk $s \cdots u$ including all but the last step $u \cdot t$ of $w$ is contained in $\mathcal{W}_N(s \to u)$ for some $N$ $(\ge k)$. Given the infinitely-often update property, there exists an $m > N$ such that the edge $(u,t) \in \mathcal{S}_m \cup \kappa_m^{active}$. If $(u,t) \in \kappa_m^{active}$, then $w \in \mathcal{W}_{m-1}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_m(1)} t) \otimes \mathcal{W}(t \xrightarrow{\mathcal{S}_m} t) \in \mathcal{W}_m(s \to t)$. This can be concluded from (18) and because $s \cdots u \in \mathcal{W}_{m-1}(s \to u)$ by the nesting argument $(m-1 \ge N)$ of Proposition 3. Again applying the nesting argument, we can prove the proposition because we now have that $w \in \mathcal{W}_n(s \to t)$ for all $n \ge m$. We can use a similar argument to conclude that $w \in \mathcal{W}_n(s \to t)$ for all $n \ge m$, if $(u,t) \in \mathcal{S}_m$. □

## E. Proof of Theorem 2

From Theorem 1 and Proposition 1, we can conclude that $\widehat{x}^{(n)}$ converges to $J^{-1}h$ element-wise as $n \to \infty$ for $\widehat{x}^{(0)} = 0$. Assume that $\widehat{x}^{(0)} \ne 0$. Consider a shifted linear system $J\widehat{y} = \tilde{h}$, where $\tilde{h} = h - J\widehat{x}^{(0)}$. If we solve this system using the same sequence of operations (subgraphs and failed links) that were used to obtain the iterates $\widehat{x}^{(n)}$, and with $\widehat{y}^{(0)} = 0$, then $\widehat{y}^{(n)}$ converges to the correct solution $J^{-1}h - \widehat{x}^{(0)}$ of the system $J\widehat{y} = \tilde{h}$. We will show that $\widehat{y}^{(n)} = \widehat{x}^{(n)} - \widehat{x}^{(0)}$, which would allow us to conclude that $\widehat{x}^{(n)} \to J^{-1}h$ element-wise as $n \to \infty$ for any $\widehat{x}^{(0)}$. We prove this final step inductively. The base case is clear because $\widehat{y}^{(0)} = 0$. Assume as the inductive hypothesis that $\widehat{y}^{(n')} = \widehat{x}^{(n')} - \widehat{x}^{(0)}$ for $0 \le n' \le n-1$. From this, one can check that $\widehat{y}^{(n)}_{V_n^c} = \widehat{x}^{(n)}_{V_n^c} - \widehat{x}^{(0)}_{V_n^c}$. For $t \in V_n$, we have from (15,17) that:

$$\begin{aligned}
\widehat{y}_t^{(n)} &= (J_{\mathcal{S}_n}^{-1}\, \tilde{h}_{V_n})_t + \sum_{(u,v)\in\kappa_n} (J_{\mathcal{S}_n}^{-1})_{t,v}\, R_{v,u}\, \widehat{y}_u^{r_n(u\to v)} \\
&= (J_{\mathcal{S}_n}^{-1}\, h_{V_n})_t + \sum_{(u,v)\in\kappa_n} (J_{\mathcal{S}_n}^{-1})_{t,v}\, R_{v,u}\, \widehat{x}_u^{r_n(u\to v)} \\
&\quad - \left(J_{\mathcal{S}_n}^{-1}\, (J\, \widehat{x}^{(0)})_{V_n}\right)_t - \sum_{(u,v)\in\kappa_n} (J_{\mathcal{S}_n}^{-1})_{t,v}\, R_{v,u}\, \widehat{x}_u^{(0)} \\
&= \widehat{x}_t^{(n)} - \Big(J_{\mathcal{S}_n}^{-1}\, (J_{V_n,V_n^c}\, \widehat{x}^{(0)}_{V_n^c} + J_{V_n,V_n}\, \widehat{x}^{(0)}_{V_n} \\
&\qquad\qquad + K_{\mathcal{S}_n}\, \widehat{x}^{(0)}_{V_n} + R_{V_n,V_n^c}\, \widehat{x}^{(0)}_{V_n^c})\Big)_t \\
&= \widehat{x}_t^{(n)} - \widehat{x}_t^{(0)}.
\end{aligned}$$

The second equality follows from the inductive hypothesis, and the last two from simple algebra. □

## F. Proof of Theorem 3

Before proving the converse, we have the following lemma that is proved in [31].

**Lemma 3:** Suppose $J$ is a symmetric positive-definite matrix, and $J = J_{\mathcal{S}} - K_{\mathcal{S}}$ is some splitting with $K_{\mathcal{S}}$ symmetric and $J_{\mathcal{S}}$ non-singular. Then, $\varrho(J_{\mathcal{S}}^{-1}K_{\mathcal{S}}) < 1$ if and only if $J + 2K_{\mathcal{S}} \succ 0$.

**Proof of converse**: Assume that $J = I - R$ is valid but non-walk-summable. Therefore, $R$ must contain some negative partial correlation coefficients (since all valid attractive models, i.e. those containing only non-negative partial correlation coefficients, are walk-summable; see Section II-C). Let $R = R_+ + R_-$ with $R_+$ containing the positive coefficients and $R_-$ containing the negative coefficients (including the negative sign). Consider a stationary ET iteration (7) based on the cutting the negative edges so that $J_{\mathcal{S}} = I - R_+$ and $K_{\mathcal{S}} = R_-$. If $J_{\mathcal{S}}$ is singular, then the iteration is ill-posed. Otherwise, the iteration converges if and only if $\varrho(J_{\mathcal{S}}^{-1}K_{\mathcal{S}}) < 1$ [15, 16]. From Lemma 3, we need to check the validity of $J + 2K_{\mathcal{S}}$:

$$J + 2K_{\mathcal{S}} = I - R + 2R_- = I - \bar{R}.$$

But $I - \bar{R} \succ 0$ if and only if the model is walk-summable (from Section II-C). Thus, this stationary iteration, if well-posed, does not converge in non-walk-summable models. □

## References

[1] S. L. Lauritzen, *Graphical Models*. Oxford, U.K.: Oxford University Press, 1996.

[2] M. I. Jordan, "Graphical Models," *Statistical Science (Special Issue on Bayesian Statistics)*, vol. 19, pp. 140–155, 2004.

[3] C. Crick and A. Pfeffer, "Loopy Belief Propagation as a basis for communication in sensor networks," in *19th Conference on Uncertainty in Artificial Intelligence*, 2003.

[4] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 721–741, June 1984.

[5] J. Woods, "Markov Image Modeling," *IEEE Transactions on Automatic Control*, vol. 23, pp. 846–850, October 1978.

[6] R. Szeliski, "Bayesian modeling of uncertainty in low-level vision," *Journal of Computer Vision*, vol. 5, no. 3, pp. 271–301, December 1990.

[7] P. Rusmevichientong and B. Van Roy, "An Analysis of Turbo Decoding with Gaussian densities," in *Neural Information Processing Systems*. Advances in MIT press, 2000.

[8] P. W. Fieguth, W. C. Karl, A. S. Willsky, and C. Wunsch, "Multiresolution optimal interpolation and statistical analysis of TOPEX/POSEIDON satellite altimetry," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 33, pp. 280–292, March 1995.

[9] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kauffman, 1988.

[10] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky, "Embedded Trees: Estimation of Gaussian processes on graphs with cycles," *IEEE Transactions on Signal Processing*, vol. 52, no. 11, pp. 3136–3150, November 2004.

[11] L. K. Saul and M. I. Jordan, "Exploiting Tractable Substructures in Intractable Networks," in *Neural Information Processing Systems*, 1995.

[12] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, "Tree-based reparameterization framework for analysis of sum-product and related algorithms," *IEEE Transactions on Information Theory*, vol. 49, pp. 1120–1146, May 2003.

[13] E. B. Sudderth, "Embedded Trees: Estimation of Gaussian Processes on Graphs with Cycles," Master's thesis, Massachusetts Institute of Technology, 2002.

[14] V. Delouille, R. Neelamani, and R. Baraniuk, "Robust Distributed Estimation using the Embedded Subgraphs Algorithm," *IEEE Transactions on Signal Processing*, vol. 54, pp. 2998–3010, August 2006.

[15] G. H. Golub and C. H. Van Loan, *Matrix Computations*. Baltimore, Maryland: The Johns Hopkins University Press, 1990.

[16] R. S. Varga, *Matrix Iterative Analysis*. New York: Springer-Verlag, 2000.

[17] D. M. Malioutov, J. K. Johnson, and A. S. Willsky, "Walk-Sums and Belief Propagation in Gaussian Graphical Models," *Journal of Machine Learning Research*, vol. 7, pp. 2031–2064, October 2006.

[18] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*. Cambridge, U.K.: Cambridge University Press, 1994.

[19] R. Bru, F. Pedroche, and D. B. Szyld, "Overlapping Additive and Multiplicative Schwarz iterations for H-matrices," *Linear Algebra and its Applications*, vol. 393, pp. 91–105, 2004.

[20] A. Frommer and D. B. Szyld, "H-Splittings and Two-stage iterative methods," *Numerische Mathematik*, vol. 63, pp. 345–356, 1992.

[21] T. Gu, X. Liu, and X. Chi, "Relaxed Parallel Two-Stage Multisplitting Methods II: Asynchronous Version," *International Journal of Computer Mathematics*, vol. 80, no. 10, pp. 1277–1287, October 2003.

[22] T. Speed and H. Kiiveri, "Gaussian Markov probability distributions over finite graphs," *Annals of Statistics*, vol. 14, no. 1, pp. 138–150, March 1986.

[23] L. Scharf, *Statistical Signal Processing*. Upper Saddle River, NJ: Prentice-Hall, 2002.

[24] W. Rudin, *Principles of Mathematical Analysis*. New York: Mc-Graw Hill, 1976.

[25] R. Godement, *Analysis I*. Springer-Verlag, 2004.

[26] P. Fieguth, W. Karl, and A. Willsky, "Efficient multiresolution counterparts to variational methods for surface reconstruction," *Computer Vision and Image Understanding*, vol. 70, no. 2, pp. 157–176, May 1998.

[27] W. G. Strang and G. J. Fix, *An Analysis of the Finite Element method*. Wellesley Cambridge Press, 1973.

[28] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2001.

[29] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Transactions on Programming Languages and Systems*, vol. 5, no. 1, pp. 66–77, January 1983.

[30] B. Awerbuch, "Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems," in *Annual ACM Symposium on Theory of Computing*, 1987.

[31] L. Adams, "m-Step Preconditioned Conjugate Gradient methods," *SIAM Journal on Scientific and Statistical Computing*, vol. 6, pp. 452–463, April 1985.