

Extended Message Passing Algorithm for Inference in Loopy Gaussian Graphical Models ^{*}

K.H. Plarre and P.R. Kumar ^{a,*}

^a*Department of Electrical and Computer Engineering, and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign*

Abstract

We consider message passing for probabilistic inference in undirected Gaussian graphical models. We show that for singly connected graphs, message passing yields an algorithm that is equivalent to the application of Gaussian elimination to the solution of a particular system of equations. This relation provides a natural way of extending message passing to arbitrary graphs with loops by first studying the operations required by Gaussian elimination. We thus obtain a finite time convergent algorithm that solves the inference problem exactly and whose complexity grows gradually with the “distance” of the graph to a tree. This algorithm can be implemented in a distributed fashion at nodes through message passing, as for example in sensor networks.

Key words: Graphical models, message passing, loopy graphs, probabilistic inference, Gaussian inference

^{*} This material is based upon work partially supported by USARO under Contract Nos. DAAD19-00-1-0466 and DAAD19-01010-465, DARPA under Contract Nos. N00014-01-1-0576 and F33615-01-C-1905, AFOSR under Contract No. F49620-02-1-0217, DARPA/AFOSR under Contract No. F49620-02-1-0325, and NSF under Contract No. NSF ANI 02-21357. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the above agencies.

^{*} University of Illinois at Urbana-Champaign, CSL, 1308 West Main St., Urbana, Illinois, 61801.

Email addresses: plarre@control.cs1.uiuc.edu (K.H. Plarre),
prkumar@control.cs1.uiuc.edu (P.R. Kumar).

URL: <http://black1.cs1.uiuc.edu/~prkumar> (P.R. Kumar).

1 Introduction

Given a set of random variables, the problem of probabilistic inference can be cast as one of computing the posterior probability of a subset of variables, given the values of another subset; see for example [9]. When the number of variables is large, inference requires integration over high dimensional spaces and can easily become intractable.

In some cases, there are several conditional independence relationships between sets of random variables. The collection of all such conditional independence relations gives rise to a factorization of the joint probability distribution into a product of functions, each of which depends on a subset of the variables. This factorization can significantly reduce the complexity of inference. The focus of this paper is on efficient probabilistic inference for such collections of random variables.

In *graphical models*, all such conditional independence relations of a set of random variables are encoded in a graph. Each node in the graph represents a random variable and the independence relations are encoded in the edges. The graph can be directed (e.g., Bayesian networks) or undirected (e.g., Markov random fields). In this paper we focus on undirected graphs. The Hammersley-Clifford theorem (e.g., [3,12,9]) provides the connection between independence and factorization: A strictly positive probability distribution satisfies all conditional independence relations implied by the graph G , if and only if it factors according to the maximal cliques of G . When the underlying graph is singly connected (there is at most one path between any pair of nodes, i.e., it is a tree or a forest), efficient algorithms exist that solve the inference problem; [12,5]. In particular, *message-passing* algorithms (e.g., [12]) associate the nodes with individual processors that can perform “local” computations and communicate with each other through “messages.” The messages converge after a finite number of steps, after which each node has correctly computed its own posterior distribution.

When the graph has loops, the same message passing scheme can be applied, but the algorithm is not guaranteed to converge, and if it does, it will, in general, not converge to the correct posterior distributions; see [12,9]. Despite this, message-passing algorithms have been applied with enormous success in loopy graphical models, as a way of approximating the posterior distributions; see, for example [6].

In Gaussian graphical models (models in which the nodal variables are jointly Gaussian) the problem of probabilistic inference reduces to that of computing the posterior mean and covariance. The problem is linear [10], and the algorithm takes a simplified form [16]. In fact, the posterior mean can be obtained as the solution to a system of linear equations, while the posterior covariance can be obtained as the inverse of a matrix. Recently, message passing algorithms have been studied on Gaussian graphical models with loops – on the turbo-decoding graph with Gaussian nodes in [13], and for arbitrary graphs with Gaussian nodes in [16]. In both cases it is shown that when message passing converges, the computed posterior means are correct, but the covariances are wrong. Also, both papers provide sufficient conditions for convergence. In [15,14] an iterative “Embedded Trees” (ET) algorithm for computing the posterior means and covariances is presented. At each step, a modified inference problem is solved on a spanning subtree of the graph G , and the computed mean and covariance is used in the subsequent iteration. It is shown that if the algorithm converges, it does so geometrically, to the exact means and covariances. In [14], the ET algorithm is studied. The relation of the ET algorithm to different iterative methods for the solution of systems of linear equations is found. In particular, it is shown that the ET algorithm can be used to derive a finite time algorithm that computes the posterior means in time $O(dNE)$ and the covariances in time $O(dNE^2)$. Here N is the number of nodes in the graph G , E is the minimum number of edges that need to be removed from G to reveal an embedded tree, and d is the dimension of the nodal variables. It is important to mention that the ET algorithm can be applied to arbitrary graphical models, not only Gaussian, providing a recursive algorithm for inference.

We consider message passing for general Gaussian graphical models, those with loops, or without loops (i.e., a tree). We focus on scalar random variables, but the results presented can easily be generalized to random vectors. We show that when the graph is a tree, the resulting algorithm is equivalent to the application of Gaussian elimination to the solution of a particular system of equations. It is known that in a rooted tree (a tree in which a particular node has been chosen to be the root), inference can be performed in two steps: A first “fine-to-coarse Kalman filtering step,” equivalent to Gaussian elimination, (in the message passing context, this step corresponds to messages sent starting from the leaves towards the root), and a backsubstitution or “Rauch-Tung-Striebel smoothing step;” see [4,17,11]. We show how message passing solves the inference problem by sending messages directly in the nonrooted tree, and how this is equivalent to Gaussian elimination in many rooted trees. In fact, the inference problem can be solved by considering N rooted trees, each one rooted at a different node, and applying only the Kalman filter step on each of them. Message passing performs these computations simultaneously in the nonrooted tree. This observation not only facilitates the understanding of the

dynamics of the algorithm, but what is more, it allows us to generalize message passing in a natural way to handle graphs with loops. Thus we obtain a finite time algorithm converging in $O(NL)$ steps to the correct posterior means, and an algorithm that computes the posterior means and variances in $O(NL^2)$ steps, where L is the number of nodes that are not isolated in $G - T$ (the subgraph of G remaining after the edges of T have been deleted), where T is a spanning subtree of G . Although obtained through a different approach, the complexity of our algorithms is equivalent to that of the algorithms presented in [14], when $d = 1$.

In Sections 2 and 3 we present the details of the graphical models we consider, the assumptions we make, and introduce some notation. Section 4 presents expressions for the messages in Gaussian graphical models. In Section 5 we show that when G is a tree, the system of equations to be solved has a very simple structure, a fact that is mentioned in [17], and in Section 7 we exploit this structure to solve the system through Gaussian elimination, and show that the algorithm we obtain is equivalent to message passing. We include a brief review of Gaussian elimination in Section 6. In Sections 8 and 9 we use this equivalence to extend the message passing algorithm to compute the posterior means and variances, respectively, in graphs with loops. Section 10 presents the proposed algorithms as pseudo-code. We include a centralized version that can be implemented on a single processor, and a distributed one, that can be implemented in, for example a sensor network, where each node is, in fact, a separate physical unit. In Section 11 we study the complexity of the proposed algorithms, and we conclude with some remarks in Section 12.

2 The setting

We follow closely the setting and notation in [15] and [16]. We consider a Gaussian stochastic process given by an (unobserved) \mathbb{R}^N valued state vector $x \sim \mathcal{N}(0, \Sigma^{-1})$, with probability density function $p(x) \propto \exp\left\{-\frac{1}{2}x^T \Sigma x\right\}$ where $\Sigma = \Sigma^T > 0$. We associate with x an undirected graph $G = (V_G, E_G)$ which contains N nodes, each indexed by an x_i , and with an edge connecting x_i and x_j for $i \neq j$ if and only if $[\Sigma]_{i,j}$, the i, j -th entry of Σ , is nonzero. The manner in which Σ and G have been defined makes x Markov with respect to G , i.e., for any three subsets $\mathcal{S}_1 := \{x_{i_1}, \dots, x_{i_{n_1}}\}$, $\mathcal{S}_2 := \{x_{j_1}, \dots, x_{j_{n_2}}\}$, and $\mathcal{S}_3 := \{x_{k_1}, \dots, x_{k_{n_3}}\}$, if removing the vertices in \mathcal{S}_2 from G completely isolates \mathcal{S}_1 from \mathcal{S}_3 , then the variables in \mathcal{S}_1 are mutually conditionally independent of the variables in \mathcal{S}_3 , given \mathcal{S}_2 .

For simplicity (only) we suppose that each node in G corresponds to a component of x , and not to a subvector as in the more general case. To each x_i

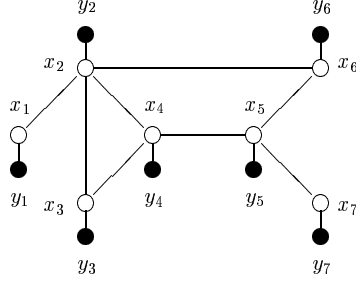


Fig. 1. Example of a Gaussian stochastic process defined on a loopy graph. Variables are represented by circles. Observed variables are colored black.

corresponds a noisy observation y_i such that the observation vector y satisfies $y = Cx + v$, $v \sim \mathcal{N}(0, R)$. We append to G , N more nodes indexed by y_1, \dots, y_N , and edges (x_i, y_i) for $1 \leq i \leq N$. We call this graph as $\bar{G} = (V_{\bar{G}}, E_{\bar{G}})$. In \bar{G} , since each y_i is connected only to x_i , it implies that the random variables $\{y_i\}_{i=1}^N$ are conditionally independent given x , which in turn means that C and R are diagonal matrices. Figure 1 shows an example of such a graphical model, where the observed nodes, the nodes corresponding to y_i for $1 \leq i \leq N$, are colored black. We call this a Gaussian graphical model.

The goal of the inference problem is to determine the conditional marginals $p(x_i | y)$, i.e., the posterior probability of each x_i given the observations, when Σ , C and R are given. Let $P := \Sigma^{-1}$. As is standard [16], we consider $\Sigma = P^{-1}$ rather than P as given, since the joint probability distribution of x is given by the coefficients of Σ . Since the joint distribution is Gaussian, the posterior distribution is also Gaussian, and it suffices to determine the posterior mean \hat{x} and covariance \hat{P} . It is known that \hat{x} and \hat{P} satisfy¹

$$\begin{aligned} \hat{x} &= \hat{P}C^T R^{-1}y, \\ \hat{P} &= [P^{-1} + C^T R^{-1}C]^{-1}. \end{aligned} \tag{1}$$

Note that, since the conditional error variances are the diagonal elements of \hat{P} , solving (1) is, in fact, more general than solving the inference problem as stated above, because the complete posterior covariance is computed, and not just its diagonal elements. Here we are interested only in computing the diagonal elements of \hat{P} .

¹ From the expression for y , we see that, conditioned on x , y is a Gaussian random vector with mean Cx and covariance matrix R . The conditional density of x given y is given by $p(x|y) = p(y|x)p(x)/p(y) \propto \exp\{-\frac{1}{2}[(y - Cx)^T R^{-1}(y - Cx) + x^T P^{-1}x]\}$. Completing the square in this expression we find that conditioned on y , x is a Gaussian random vector with covariance $\hat{P} = [P^{-1} + C^T R^{-1}C]^{-1}$ and mean $\hat{x} = \hat{P}C^T R^{-1}y$.

We now list the notation and conventions that will be used in what follows:

- (1) The nodes of \bar{G} correspond to the state variables x_i and observations y_i . To simplify the notation, we sometimes use indices to indicate the state nodes, e.g., “node i ” means “node x_i ”, and “edge (i, j) ” refers to the edge joining x_i and x_j . We name the observations as y_i .
- (2) N_i denotes the set of indices of all state variables that are neighbors of i in G , i.e., $N_i = \{1 \leq j \leq N \mid (i, j) \in E_G\}$. $N_{i,j}$ is the set of indices of all state variables that are neighbors of x_i , except x_j , i.e., $N_{i,j} = N_i \setminus \{j\}$.
- (3) If M is a matrix, $[M]_{i,j}$ and $[M]_{i,*}$ denote the (i, j) -th component and the i -th row of M , respectively. Likewise $[w]_i$ denotes the i -th component of vector w .
- (4) To simplify the presentation, we define the following matrices: $V_{x,x} := \hat{P}^{-1}$, $V_{x,y} := -CR^{-1}$ and $V_{y,y} := R^{-1}$. Also, we let $b := -V_{x,y}y$. The equation that defines the posterior mean in this notation is

$$V_{x,x}\hat{x} = b.$$

- (5) We only consider the application of message passing on trees. We sometimes distinguish a particular node x_r as the root node, and relabel the variables in breadth first order, starting from the root (see Figure 2). Note that in general, this labeling is not unique. The particular breadth-first-order labeling chosen does not affect our conclusions. Let Q_r be a permutation matrix that maps the original indices to the new ones, and ϕ_r a function such that $\phi_r(i)$ is the index of x_i in the breadth first order labeling when x_r is the root node. In Figure 2, for example, $\phi_r(4) = 1$, $\phi_r(2) = 2$, $\phi_r(3) = 3$, $\phi_r(5) = 4$, $\phi_r(1) = 5$, $\phi_r(6) = 6$ and $\phi_r(7) = 7$.
- (6) When applying message passing in the rooted tree, we use π_i to denote the parent of node i .
- (7) As usual in graphical models, we use the notation $u = \alpha v$, to denote that u is proportional to v . Here, u and v could be any pair of compatible quantities (vectors, functions, etc.).
- (8) In most figures the observed nodes are not shown explicitly. This is only for clarity; we always consider graphical models on the variables $\{x_i\}_{i=1}^N$ and $\{y_i\}_{i=1}^N$.

We begin by studying the message passing algorithm when G is a tree and find its relation to Gaussian elimination. Then we use this relation to extend the algorithm to handle graphs G with loops.

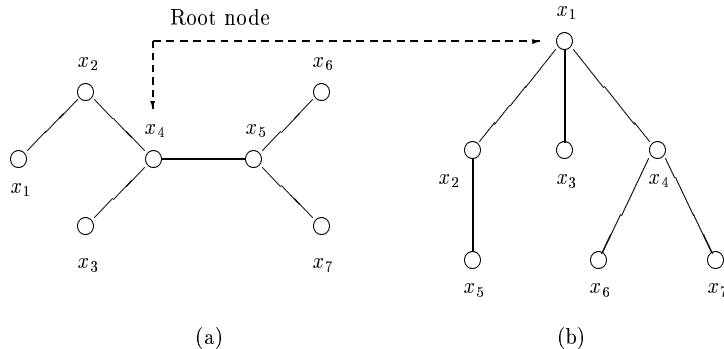


Fig. 2. Example of relabeling process. The tree in (a) shows the original labeling. After node x_4 has been chosen as the root and the variables have been relabeled in breadth first order, the tree in (b) is obtained.

3 Parametrization

In order to use message passing to solve the inference problem, it is necessary to have a factorization of the joint distribution as a product of local functions or “kernels” (functions of the maximal cliques in the graph G). In general, the factorization is not unique. Each choice of local functions provides a parametrization of the joint distribution. To be valid, a parametrization needs to satisfy certain conditions, which we study in this section. We begin with an expression for the joint density function:

$$p(x, y) = \alpha \exp \left\{ -\frac{1}{2} \begin{pmatrix} x \\ y \end{pmatrix}^T \begin{pmatrix} V_{x,x} & V_{x,y} \\ V_{x,y}^T & V_{y,y} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \right\}. \quad (2)$$

Our goal is to construct a message passing algorithm for loopy graphs as a modification of the message passing algorithm for trees. We begin by assuming that $V_{x,x}$ corresponds to a tree². Noting that a tree is a pairwise graph, i.e., a graph in which the largest clique has size 2, we know that the joint distribution factors into a product of local kernels, each of which depends only on 2 variables

$$p(x, y) = \prod_{(i,j) \in E_G} \Psi_{i,j}(x_i, x_j) \prod_{k=1}^N \Psi_{k,k}(x_k, y_k).$$

² We say that a matrix $V_{x,x}$ “corresponds” to a graph G if the structure of $V_{x,x}$ respects the connectivity of G , i.e., if $[V_{x,x}]_{i,j} \neq 0$ only if $i = j$ or, for $i \neq j$, x_i is connected to x_j in G .

For Gaussian graphical models the kernels have the following general form:

$$\begin{aligned} \Psi_{i,j}(x_i, x_j) &= \alpha \exp \left\{ -\frac{1}{2} \begin{pmatrix} x_i \\ x_j \end{pmatrix}^T \begin{pmatrix} Z_{x_i, x_i} & Z_{x_i, x_j} \\ Z_{x_i, x_j} & Z_{x_j, x_j} \end{pmatrix} \begin{pmatrix} x_i \\ x_j \end{pmatrix} \right\}, \\ \Psi_{i,i}(x_i, y_i) &= \alpha \exp \left\{ -\frac{1}{2} \begin{pmatrix} x_i \\ y_i \end{pmatrix}^T \begin{pmatrix} Z_{x_i, x_i} & Z_{x_i, y_i} \\ Z_{x_i, y_i} & Z_{y_i, y_i} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \right\}. \end{aligned} \quad (3)$$

For (3) to constitute a correct parametrization of the joint distribution (2), the following conditions have to be met

$$\begin{aligned} [V_{x,x}]_{i,j} &= \begin{cases} Z_{x_i, x_i} + \sum_{k \in N_i} Z_{x_k, x_i} & \text{if } i = j, \\ Z_{x_i, x_j} & \text{if } j \in N_i, \\ 0 & \text{else,} \end{cases} \\ V_{x,y} &= \text{diag}(Z_{x_1, y_1}, \dots, Z_{x_N, y_N}), \\ V_{y,y} &= \text{diag}(Z_{y_1, y_1}, \dots, Z_{y_N, y_N}). \end{aligned} \quad (4)$$

Note that for each pair of indices, i and j , $Z_{x_i, x_j} = Z_{x_j, x_i}$. Likewise $Z_{x_i, x_i} = Z_{x_i, x_i}$, and $Z_{x_i, x_j} = Z_{x_j, x_i}$.

Given Σ , C and R it is an easy task to find a valid parametrization, Z_{x_i, x_j} , Z_{x_i, x_i} , Z_{x_j, x_j} , Z_{x_i, y_i} , Z_{x_i, y_j} , Z_{y_i, y_i} for $1 \leq i, j \leq N$, that satisfies (4).

4 The messages

In message passing, each node in G sends one or more messages to each of its neighbors. The message that x_i sends to x_j contains the parameters of a function of x_j . In a Gaussian graphical model the messages are parametrizations of Gaussian distribution functions and can be therefore be represented by their mean and inverse covariance, i.e. “precision.” We denote the function corresponding to the message from x_i to x_j as $m_{i,j}(x_j)$, and its mean and inverse covariance as $\mu_{i,j}$ and $P_{i,j}$, respectively.

When G is a tree, the order in which the messages are sent is irrelevant to the final result. For our purposes, it is convenient to schedule the algorithm

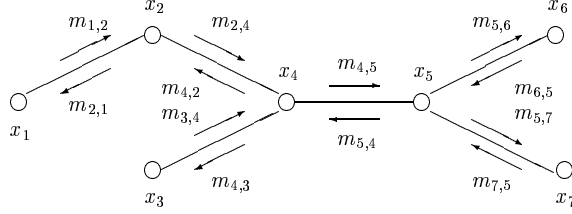


Fig. 3. Complete set of messages transmitted during application of message passing to solve the inference problem for each node in the graph.

according to the following rule:

“Node i is allowed to send a message to node j only after it has received messages from all its neighbors, except j .”

Note that this rule cannot be implemented if G has loops. However, for trees such a procedure can be initiated at the leaves of the tree and progress inwards into the tree, as can be deduced from Figure 3.

The inference problem we want to solve is the computation of the posterior distributions of all the state nodes in G , given the observations. This problem is solved by sending messages between each pair of neighboring nodes. For example, Figure 3 shows a tree and the complete set of messages necessary to solve the inference problem.

Solving the inference problem for only one state variable x_r requires the transmission of only a subset of the messages (but each message is computed exactly as in the general case). We consider this simpler problem first. It is here that the relation to Gaussian elimination becomes clear. For clarity, we consider the tree rooted at x_r and relabel the variables in breadth first order, starting from the root, as depicted in Figure 2. The messages that are necessary to find the posterior distribution of the root node are the ones flowing upwards towards the root. An example of message passing on the rooted tree is shown in Figure 4.

We now present the general message passing algorithm. The mean and precision of the message that a node i sends to its neighbor j are given by

$$P_{i,j} = Z_{x_i,x_j}^{x_j,x_j} - \frac{(Z_{x_i,x_j}^{x_i,x_j})^2}{Z_{x_i,x_j}^{x_i,x_i} + P_0^{i,j}},$$

$$\mu_{i,j} = -\frac{Z_{x_i,x_j}^{x_i,x_j} P_0^{i,j} \mu_0^{i,j}}{P_{i,j} (Z_{x_i,x_j}^{x_i,x_i} + P_0^{i,j})},$$

where

$$P_0^{i,j} = Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_{i,j}} P_{k,i},$$

$$P_0^{i,j} \mu_0^{i,j} = -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_{i,j}} P_{k,i} \mu_{k,i}.$$

When message passing has finished, node i computes its posterior mean, μ_i , and precision, P_i , according to the following equations:

$$P_i = Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_i} P_{k,i},$$

$$P_i \mu_i = -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_i} P_{k,i} \mu_{k,i}.$$

Note that in the rooted tree only the root node (x_1 in the new labeling) computes μ_1 and P_1 . In the rest of this section we suppose that message passing is performed in the rooted tree. To avoid confusing notation, we assume that $V_{x,x}$ and the parametrization of the joint distribution are given with respect to the new labeling.

In Section 7 we compare message passing to Gaussian elimination. For this we need detailed expressions for the messages. Expanding the expression for $P_0^{i,j}$

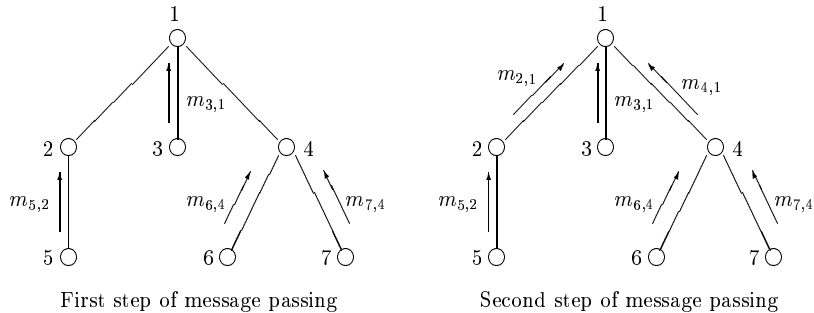


Fig. 4. Example of message passing process in the rooted tree, under the chosen schedule. Note that the tree shown here is equivalent to the tree in Figure 3, rooted at x_4 and relabeled in breadth first order. Note also that only a subset of the messages shown in Figure 3 are transmitted here.

we find that

$$\begin{aligned}
P_0^{i,j} &= Z_{x_i,y_i}^{x_i,x_i} + \sum_{k \in N_{i,j}} P_{k,i} \\
&= Z_{x_i,y_i}^{x_i,x_i} + \sum_{k \in N_{i,j}} \left[Z_{x_k,x_i}^{x_i,x_i} - \frac{(Z_{x_k,x_i}^{x_k,x_i})^2}{Z_{x_k,x_i}^{x_k,x_k} + P_0^{k,i}} \right] \\
&= \left(Z_{x_i,y_i}^{x_i,x_i} + \sum_{k \in N_{i,j}} Z_{x_k,x_i}^{x_i,x_i} \right) - \sum_{k \in N_{i,j}} \frac{(Z_{x_k,x_i}^{x_k,x_i})^2}{Z_{x_k,x_i}^{x_k,x_k} + P_0^{k,i}} \\
&= [V_{x,x}]_{i,i} - Z_{x_i,x_j}^{x_i,x_i} - \sum_{k \in N_{i,j}} \frac{(Z_{x_k,x_i}^{x_k,x_i})^2}{Z_{x_k,x_i}^{x_k,x_k} + P_0^{k,i}},
\end{aligned} \tag{5}$$

where j is the parent of i . Defining $R_{i,j} := Z_{x_i,x_j}^{x_i,x_i} + P_0^{i,j}$ we find

$$R_{i,j} = [V_{x,x}]_{i,i} - \sum_{k \in N_{i,j}} \frac{(Z_{x_k,x_i}^{x_k,x_i})^2}{R_{k,i}}. \tag{6}$$

The initial condition for this recursion is $R_{l,\pi_l} = [V_{x,x}]_{l,l}$ for l a leaf node. In the same fashion we can see that

$$\begin{aligned}
P_0^{i,j} \mu_0^{i,j} &= -Z_{x_i,y_i}^{x_i,y_i} y_i + \sum_{k \in N_{i,j}} P_{k,i} \mu_{k,i} \\
&= -Z_{x_i,y_i}^{x_i,y_i} y_i - \sum_{k \in N_{i,j}} \frac{Z_{x_k,x_i}^{x_k,x_i}}{Z_{x_k,x_i}^{x_k,x_k} + P_0^{k,i}} P_0^{k,i} \mu_0^{k,i}.
\end{aligned} \tag{7}$$

Letting $S_{i,j} := P_0^{i,j} \mu_0^{i,j}$ we can write

$$S_{i,j} = [b]_i - \sum_{k \in N_{i,j}} \frac{Z_{x_k,x_i}^{x_k,x_i}}{R_{k,i}} S_{k,i}, \tag{8}$$

with initial condition $S_{l,\pi_l} = -Z_{x_l,y_l}^{x_l,y_l} y_l = [b]_l$ for each root node l . The detailed expression for the precision at the root node is

$$\begin{aligned}
P_1 &= Z_{x_1,y_1}^{x_1,x_1} + \sum_{k \in N_1} P_{k,1} \\
&= Z_{x_1,y_1}^{x_1,x_1} + \sum_{k \in N_1} \left[Z_{x_k,x_1}^{x_1,x_1} - \frac{(Z_{x_k,x_1}^{x_k,x_1})^2}{Z_{x_k,x_1}^{x_k,x_k} + P_0^{k,1}} \right] \\
&= \left(Z_{x_1,y_1}^{x_1,x_1} + \sum_{k \in N_1} Z_{x_k,x_1}^{x_1,x_1} \right) - \sum_{k \in N_1} \frac{(Z_{x_k,x_1}^{x_k,x_1})^2}{Z_{x_k,x_1}^{x_k,x_k} + P_0^{k,1}} \\
&= [V_{x,x}]_{1,1} - \sum_{k \in N_1} \frac{(Z_{x_k,x_1}^{x_k,x_1})^2}{R_{k,1}}.
\end{aligned} \tag{9}$$

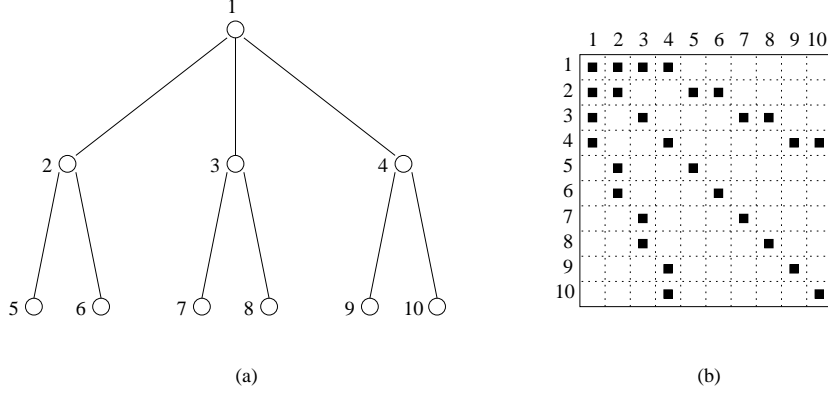


Fig. 5. Figure (b) shows the structure of $V_{x,x}$ corresponding to the tree in Figure (a). Nonzero elements are shown as black squares.

Likewise, the value of $P_1\mu_1$ is given by

$$\begin{aligned}
 P_1\mu_1 &= -Z_{x_1,y_1}^{x_1,y_1}y_1 + \sum_{k \in N_1} P_{k,1}\mu_{k,1} \\
 &= [b]_1 - \sum_{k \in N_1} \frac{Z_{x_k,x_1}^{x_k,x_1}}{R_{k,1}} S_{k,1}.
 \end{aligned} \tag{10}$$

These expressions will be used later.

5 Structure of $V_{x,x}$

In Section 7 we apply Gaussian elimination to solve the inference problem for the root node in the relabeled rooted tree. For this we first need to find the structure³ of $V_{x,x}$. As in Section 4, we simplify the problem by assuming that $V_{x,x}$ is given with respect to the relabeled variables.

Recall that in a Gaussian graphical model $[V_{x,x}]_{i,j}$ can be nonzero only if x_i is connected by an edge to x_j in G ; see (4). In other words, $V_{x,x}$ has the same structure as the adjacency matrix of G (except for the diagonal terms).

The breadth first order labeling gives $V_{x,x}$ a very simple structure. To see this, consider for example the relabeled rooted tree shown in Figure 5(a). Consider a fixed row i of $V_{x,x}$. Entry $[V_{x,x}]_{i,j}$ can be nonzero only if x_j is neighbor of x_i in G . We can see that the only neighbor x_j of x_i with $j < i$ is the parent of x_i in the rooted tree. Likewise the neighbors of x_i with index higher than i are the children of x_i . This means that row i of $V_{x,x}$ contains nonzero entries in

³ By “structure” we mean the positions of the nonzero entries.

the positions corresponding to the parent of x_i , x_i itself (the diagonal term), and the children of x_i . It is also important to note that in each row i there is a unique nonzero element to the left of the diagonal, corresponding to the unique parent of i . By the symmetry of $V_{x,x}$, it follows that in each column i there is a unique nonzero element above the diagonal, again corresponding to the parent of x_i . This structure of $V_{x,x}$ for the tree in Figure 5(a) is shown in Figure 5(b).

6 Review of Gaussian elimination

In this section we review briefly Gaussian elimination and introduce some terminology which will be helpful in describing how Gaussian elimination is applied in the context of graphical models.

Consider the system of equations

$$Ax = b. \tag{11}$$

Suppose that we want to transform this system into an equivalent one (one that has the same solution)

$$A'x = b',$$

such that $[A']_{i,j} = 0$ for some fixed $i \neq j$. We will say that element $[A]_{i,j}$ has been “eliminated.” We do so by adding a scaled version of row j to row i . In this case we say that “row j has been used to eliminate element $[A]_{i,j}$ ” or, sometimes that “element $[A]_{j,j}$ has been used to eliminate element $[A]_{i,j}$.” To formalize, we premultiply (11) by a matrix W defined in the following way:

$$[W]_{k,l} = \begin{cases} 1 & \text{if } k = l, \\ -\frac{[A]_{i,j}}{[A]_{j,j}} & \text{if } i \neq j \text{ and } (k, l) = (i, j), \\ 0 & \text{else.} \end{cases}$$

We call $[W]_{i,j}$ the “elimination factor.” Then, $A' = WA$ and $b' = Wb$. To make the notation simpler, we do not mention the matrix W explicitly. In fact we consider that the i – th row of matrix A and the i – th element of b are

updated directly, i.e.,

$$\begin{aligned}
[A]_{i,*} &\leftarrow [A]_{i,*} - \frac{[A]_{i,j}}{[A]_{j,j}} [A]_{j,*} , \\
[b]_i &\leftarrow [b]_i - \frac{[A]_{i,j}}{[A]_{j,j}} [b]_j .
\end{aligned}$$

This update operation is known as “row operation.”

When the values of some of the variables in (11) are known, they can be substituted into the equations, thus reducing the order of the system to be solved. This procedure is called “backsubstitution.” See [8] for a fuller more detailed explanation of Gaussian elimination.

7 Gaussian elimination for trees

In this section we still assume that inference has to be made only for a particular node x_r . After relabeling the variables, the problem can be stated as one of finding $[\hat{x}_r]_1$ and $[V_r^{-1}]_{1,1}$, where

$$\begin{aligned}
V_r \hat{x}_r &= b_r , \\
V_r &:= Q_r V_{x,x} Q_r^T , \\
b_r &:= Q_r b .
\end{aligned}$$

As in Section 4, we assume that $V_{x,x}$ and the parametrization of the joint distribution are given in terms of the breadth-first-order labeling. This allows us to use $V_{x,x}$ directly and drop the subindex r . So, we have to find $[\hat{x}]_1$ and $[V_{x,x}^{-1}]_{1,1}$, with

$$V_{x,x} \hat{x} = b .$$

We use Gaussian elimination to solve this problem. In order to do this, we eliminate the elements in the upper triangle of $V_{x,x}$ using row operations. After this process, the system is transformed into

$$V'_{x,x} \hat{x} = b' ,$$

where $V'_{x,x}$ is a lower triangular matrix. Let d_i denote the i -th diagonal entry of $V'_{x,x}$, and $[b']_i$ the i -th component of b' . Then

$$\begin{aligned} [\hat{x}]_1 &= \frac{[b']_1}{d_1}, \\ [V_{x,x}^{-1}]_{1,1} &= \frac{1}{d_1}, \end{aligned}$$

are, in fact, the correct posterior mean and variance of the root node.

We now determine the order in which the elements in the upper triangle have to be eliminated. Considering the tree shown in Figure 5(a) and the structure of $V_{x,x}$ in Figure 5(b), it is easy to infer the following rules:

- (1) If i is a leaf node and $j > i$ then $[V_{x,x}]_{i,j} = 0$. This means that the rows corresponding to leaf nodes need not be changed.
- (2) An element in row j is eliminated using the diagonal term in the same column. Note that the elimination of the term $[V_{x,x}]_{j,i} = Z_{x_i,x_j}^{x_i,x_j}$ is done using the diagonal term in the row i , where i is a child of j in the rooted tree.
- (3) Since each column of $V_{x,x}$ contains only one nonzero element in the upper triangle, each row is used only once in the elimination process (which, in fact, corresponds to one “message” sent by a node to its parent).
- (4) The elements in row j contained in the upper triangle are exactly $[V_{x,x}]_{j,i}$, for i a child of j . These terms are eliminated using the rows corresponding to the children of j (which corresponds to the children sending a message to their parent).
- (5) Once all terms in row j in the upper triangle have been eliminated, row j can be used to eliminate the unique nonzero element contained in the upper triangle, in column j . This element is $[V_{x,x}]_{k,j}$, where k is the parent of j .

These observations imply that the elimination process can be done by following exactly the same order as message passing, i.e., start at the leaf nodes, then the parents of the leaves, and so on, until the root is reached. The complete elimination process for the tree in Figure 5 is shown in Figure 6. The order in which elimination is done gives a recursive expression for the diagonal terms $\{d_i\}_{i=1}^N$. For $i \neq 1$,

$$d_i = [V_{x,x}]_{i,i} - \sum_{k \in N_{i,j}} \frac{(Z_{x_k,x_i}^{x_k,x_i})^2}{d_k}, \quad (12)$$

where j is the parent of i in the rooted tree. The initial conditions for the

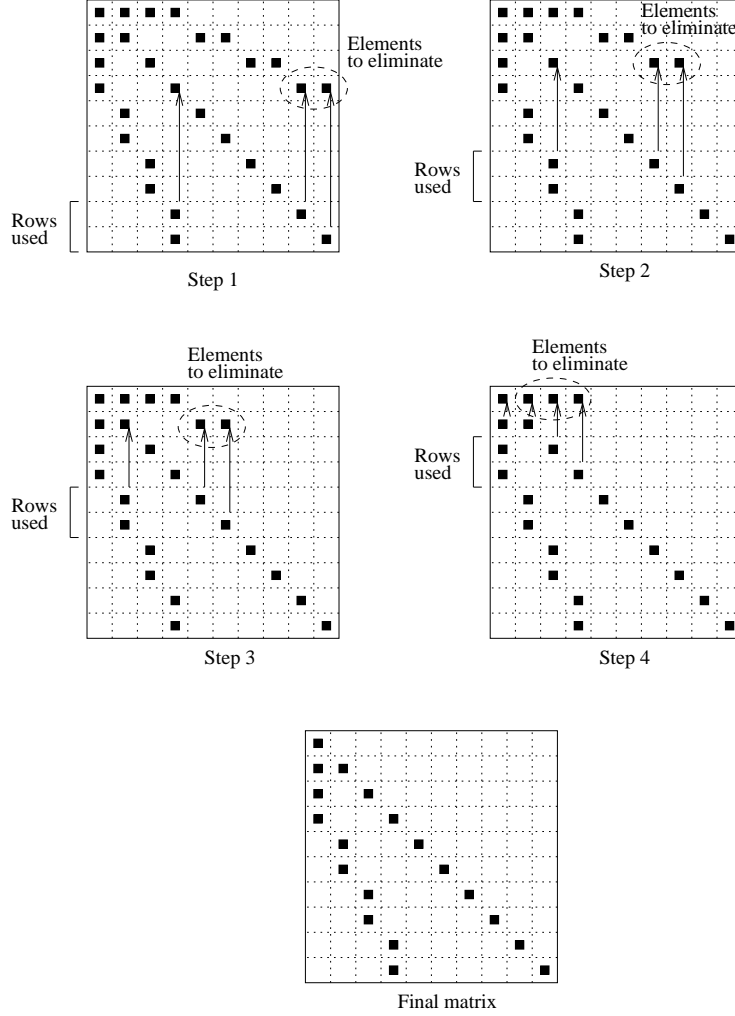


Fig. 6. The complete sequence of row operations to find the posterior mean of the root node of the tree in Figure 5(a). Each step here shows two row operations. The action of one element on another is shown as an arrow. Nonzero elements are shown as black squares.

recursion are given by $d_l = [V_{x,x}]_{l,l}$ for each leaf node l . For $i = 1$ we have

$$d_1 = [V_{x,x}]_{1,1} - \sum_{k \in N_1} \frac{(Z_{x_k, x_1}^{x_k, x_1})^2}{d_k}. \quad (13)$$

Comparing (12) and (13) to (6) and (9) we see that for each $i \geq 2$, $d_i = R_{i, \pi_i}$, and $d_1 = P_1$.

To find an expression for $[b']_1$, note that the factor used in the elimination of the term $[V_{x,x}]_{j,i}$ is $\left(-\frac{Z_{x_i, x_j}^{x_i, x_j}}{d_i}\right)$. Again, the order in which elimination is done

gives a recursion for $[b']_i$, for $2 \leq i \leq N$:

$$[b']_i = [b]_i - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{d_k} [b']_k, \quad (14)$$

with initial conditions $[b']_l = [b]_l$ for each leaf l . For $[b']_1$ we have

$$[b']_1 = [b]_1 - \sum_{k \in N_1} \frac{Z_{x_k, x_i}^{x_k, x_i}}{d_k} [b']_k. \quad (15)$$

Comparing (14) and (15) to (8) and (10), we find that $[b']_i = S_{i, \pi_i}$ for $i \geq 2$, and $[b']_1 = P_1 \mu_1$.

We have thus shown that the computations performed by message passing are equivalent to the application of Gaussian elimination to solve the associated system of equations.

Note that in the previous analysis we have fixed the root node and performed message passing to find the posterior distribution of only the root node. A priori, solving the inference problem for all nodes would mean having to run message passing on N rooted trees or, equivalently, using Gaussian elimination to find $[\hat{x}_r]_1$ in each of the N systems of equations given by

$$V_r \hat{x}_r = b_r. \quad (16)$$

However, a critical observation is that node $\phi_r(i)$ will send the same message to $\phi_r(j)$ in any rooted tree in which $\phi_r(j)$ is parent of $\phi_r(i)$. So, these messages do *not* have to be recomputed for every choice of r . In particular, if b'_r is the right hand side after Gaussian elimination. We have $P_0^{i,j} \mu_0^{i,j} = [b'_r]_{\phi_r(i)}$ in any rooted tree in which $\phi_r(j)$ is the parent of $\phi_r(i)$.

Based on this observation, message passing can be applied directly on the non-rooted tree G . In G there are no parent-child relationships. Node i sends a message to each of its neighbors (see Figure 3). When i sends a message to j , j acts as the parent of i in a rooted tree, and the nodes in $N_{i,j}$ act as its children. For example, $P_0^{i,j} \mu_0^{i,j}$ can be interpreted as: “This would be the right hand side in position $\phi_r(i)$, after Gaussian elimination, in any rooted tree in which $\phi_r(j)$ is parent of $\phi_r(i)$.”

This scheme allows message passing to solve the N inference problems *simultaneously*. The messages that a node x_r receives from its neighbors are exactly the same as those it would receive in the relabeled tree rooted at x_r .

8 Extended message passing algorithm for computing the means in loopy graphs

The fact that Gaussian elimination can be used on any matrix, suggests extending message passing to “loopy graphs,” i.e., graphs with cycles, by first studying the operations performed by Gaussian elimination. In this section we extend the message passing algorithm to compute the posterior means when G has loops. As we will see, the algorithm can be divided into three stages: Message passing, solution of a smaller system of equations, and backsubstitution of a partial solution.

8.1 Stage 1: Modified message passing

Let $V_{x,x}$ correspond to a loopy graph. We begin by finding a decomposition of $V_{x,x}$ in the following way:

- (1) Let $G = (V_G, E_G)$ be the loopy graph and $T = (V_G, E_T)$ be a spanning subtree of G .
- (2) Let $\mathcal{E} := E_G \setminus E_T$ be the set of extra edges. Let $\mathcal{V} = \{\nu_1, \dots, \nu_L\}$ denote the set of endpoints (i.e., vertices) of edges in \mathcal{E} . We call the nodes indexed by \mathcal{V} “special nodes.” Note that the number of special nodes is L (recall that L is the number of non-isolated nodes in $G - T$).
- (3) Let $K \in \mathbb{R}^{N \times N}$ be such that

$$[K]_{i,j} := \begin{cases} \sum_{(k,i) \in \mathcal{E}} Z_{x_k, x_i}^{x_i, x_i} & \text{if } i = j \in \mathcal{V}, \\ Z_{x_i, x_j}^{x_i, x_j} & \text{if } i, j \in \mathcal{V} \text{ and } i \neq j, \\ 0 & \text{else.} \end{cases}$$

Then $V_{x,x}^{\text{tree}} := V_{x,x} - K$ corresponds to a tree.

Note that in this case, N_i is the set of neighbors of i in T , and $N_{i,j} = N_i \setminus \{j\}$.

For a given root node r , let $V_r := Q_r V_{x,x}^{\text{tree}} Q_r^T$, $K_r := Q_r K Q_r^T$, and $b_r := Q_r b$. The task is now to find $[\hat{x}_r]_1$ in each of the systems

$$(V_r + K_r) \hat{x}_r = b_r, \quad 1 \leq r \leq N.$$

Let V'_r , K'_r and b'_r be the matrices after Gaussian elimination. We know that message passing in T will provide each node x_r with information about the

first row of V_r' and the first element of b_r' . We need to create new messages that allow x_r to compute the nonzero elements in the first row of K_r' .

Considering the observations at the end of last section, we let the new message from i to j in the nonrooted tree be: “These would be the values of the nonzero elements in row $\phi_r(i)$ of K_r' if $\phi_r(j)$ were the parent of $\phi_r(i)$ in the rooted tree.”

To formalize, we let each node i record a vector $\theta_{i,i} \in \mathbb{R}^L$, and each edge $(i, j) \in E_T$ a vector $\theta_{i,j} \in \mathbb{R}^L$, such that:

- $[\theta_{i,i}]_l = [K]_{i,\nu_l}$. These vectors are fixed throughout the algorithm.
- $\theta_{i,j}$ contains the nonzero elements in row $\phi_r(i)$ of K_r' in a rooted tree in which $\phi_r(j)$ is parent of $\phi_r(i)$.

From these definitions we see that the vectors $\theta_{i,j}$ satisfy the recursion

$$\theta_{i,j} = \theta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_i} + P_0^{k,i}} \theta_{k,i}.$$

When message passing has finished, each node i can compute $[V_i']_{1,1} = P_i$ and $[b_i']_1 = P_i \mu_i$. Likewise i can compute the the perturbation terms in the first row of K_i' as

$$\theta_i = \theta_{i,i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_i} + P_0^{k,i}} \theta_{k,i}.$$

The interpretation of θ_i is that $[\theta_i]_l = [K_i']_{1,\nu_l}$. We know that $(V_i' + K_i')\hat{x}_i = b_i'$. The first equation of this system, rewritten in the original labeling is

$$P_i [\hat{x}]_i + \sum_{l=1}^L [\theta_i]_l [\hat{x}]_{\nu_l} = P_i \mu_i. \quad (17)$$

Since this is true for each $1 \leq i \leq N$, equation (17), in fact, defines a new system of equations for \hat{x} .

Now, consider only the equations corresponding to the nodes indexed by \mathcal{V} . For each of these nodes we have

$$(P_{\nu_l} + [\theta_{\nu_l}]_l) [\hat{x}]_{\nu_l} + \sum_{\substack{k=1 \\ k \neq l}}^L [\theta_{\nu_l}]_k [\hat{x}]_{\nu_k} = P_{\nu_l} \mu_{\nu_l}, \quad (18)$$

for $1 \leq l \leq L$. Equation (18) corresponds to a subsystem of equations that can be solved independently. So, what message passing does in this case is to

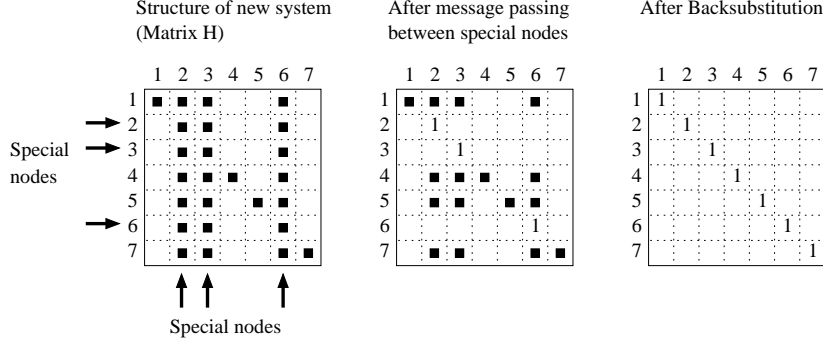


Fig. 7. Structure of new system of equations defined by equation (17). The process for solving this new system is done in two steps: Solution of a subsystem and backsubstitution.

reduce the system of equations of order N to one of order L .

8.2 Stage 2: Solution of subsystem of equations

Let $H \in \mathbb{R}^{N \times N}$ and $\beta \in \mathbb{R}^N$ such that for each $1 \leq i \leq N$:

$$[H]_{i,*} := [V'_i + K'_i]_{1,*},$$

$$[\beta]_i := P_i \mu_i.$$

The structure of H is shown in Figure 7. The system of equations defined by (17) can be rewritten as

$$H \hat{x} = \beta.$$

The system of equations defined by equation (18) can be solved by standard Gaussian elimination, which we write here as message passing, now between only the special nodes. For $1 \leq l \leq L$, let $\gamma_l \in \mathbb{R}$, and $h_l \in \mathbb{R}^L$ be such that $\gamma_l := P_{\nu_l} \mu_{\nu_l}$ and $[h_l]_k := [H]_{\nu_l, \nu_k}$ for $1 \leq k \leq L$. We implement Gaussian elimination to solve (18) in the following way:

- (1) For $l = 1$ to L do
 - (a) Node ν_l sends h_l and γ_l to each node in $\mathcal{V} \setminus \{\nu_l\}$.
 - (b) Each node $\nu_k \in \mathcal{V} \setminus \{\nu_l\}$ updates its parameters in the following way

$$\gamma_k \leftarrow \gamma_k - \gamma_l \frac{[h_k]_l}{[h_l]_l},$$

$$h_k \leftarrow h_k - h_l \frac{[h_k]_l}{[h_l]_l}.$$

(2) When the process is finished, each node $\nu_l \in \mathcal{V}$ computes $[\hat{x}]_{\nu_l}$ as

$$[\hat{x}]_{\nu_l} = \frac{\gamma_l}{[h_l]_l}. \quad (19)$$

Note that this algorithm corresponds to a “message passing version” of standard Gaussian elimination.

8.3 Stage 3: Backsubstitution

After the values of the variables $[\hat{x}]_{\nu_l}$, $1 \leq l \leq L$ are computed, they can be backsubstituted into the system defined by (17). The value of each $[\hat{x}]_i$, $i \notin \mathcal{V}$ can be found from

$$[\hat{x}]_i = \frac{1}{P_i} \left([\beta]_i - \sum_{l=1}^L [\theta_l]_i [\hat{x}]_{\nu_l} \right).$$

9 Extended message passing algorithm for computing the posterior variances in loopy graphs

In this section we construct a finite time convergent message passing algorithm to compute the variance of each estimator $[\hat{x}]_i$, $1 \leq i \leq N$, i.e the diagonal terms of the covariance matrix \hat{P} .

In the rooted tree, when solving for $[\hat{x}_r]_1$, we are really finding a matrix W_r such that $W_r V_r$ is lower triangular, and then reading out $[W_r V_r]_{1,1}$ and $[W_r b_r]_1$ (see Section 6). The new messages introduced in the previous section allow us to compute $[W_r K_r]_{1,\nu_l}$ for $1 \leq l \leq L$. We now introduce new messages to compute the first row of $I'_r := W_r I = W_r$ (the effect of the row operations on the identity matrix). Computing $[I'_r]_{1,*}$ for each node r would allow us to compute the complete posterior covariance \hat{P} , but doing so would increase excessively the complexity of the algorithm. Since we are interested only in the diagonal terms of \hat{P} , we need to compute only the rows $[I'_r]_{1,*}$ for each special node $r \in \mathcal{V}$.

Let $1 \leq l \leq L$ be fixed. To compute $[I'_{\nu_l}]_{1,i}$, we observe that

$$\begin{aligned} [\hat{P}]_{\nu_l,i} &= \frac{1}{P_{\nu_l}} [I'_{\nu_l}]_{1,i} , \\ [\hat{P}]_{i,\nu_l} &= \frac{1}{P_i} [I'_i]_{1,\nu_l} . \end{aligned}$$

By the symmetry of \hat{P} we have that

$$[I'_{\nu_l}]_{1,i} = \frac{P_{\nu_l}}{P_i} [I'_i]_{1,\nu_l} .$$

In view of this, we introduce new messages to provide each node x_r with the information necessary to compute a vector $\eta_r \in \mathbb{R}^L$, such that, $[\eta_r]_l = [I'_r]_{1,\nu_l}$. First we find expressions for the desired elements of $[I'_r]_{1,*}$ and then we translate them to messages.

In the rooted tree, if the (unique) path joining the root node and x_{ν_l} is $1 = i_1, i_2, \dots, i_m = \nu_l$, then

$$[I'_r]_{1,\nu_l} = \prod_{k=1}^{m-1} \left[-\frac{Z_{x_{i_k}, x_{i_{k+1}}}}{d_{i_{k+1}}} \right] .$$

To use this expression in the non-rooted tree T , we define $d_{i,j} \in \mathbb{R}$ through

$$d_{i,j} = [V'_r]_{i,i} = Z_{x_i, x_i} + P_0^{i,j} ,$$

and interpret it as “the i – th diagonal element of I'_r in any rooted tree in which $\Phi_r(j)$ is parent of $\Phi_r(i)$.” If the path joining x_{ν_l} and x_r in T is $\nu_l = j_1, j_2, \dots, j_n = r$ then

$$[I'_r]_{1,\nu_l} = \prod_{k=1}^{n-1} \left[-\frac{Z_{x_{j_k}, x_{j_{k+1}}}}{d_{j_k, j_{k+1}}} \right]$$

In order to compute all the required products, we define vectors $\eta_{i,i}$, $\eta_{i,j}$, and $\eta_i \in \mathbb{R}^L$, such that

$$[\eta_{i,i}]_k = \begin{cases} 1 & \text{if } i = \nu_k , \\ 0 & \text{else ,} \end{cases}$$

and $\eta_{i,j}$ is a *new message* from x_i to x_j , satisfying the recursion

$$\eta_{i,j} = \eta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}}{d_{k,i}} \eta_{k,i} .$$

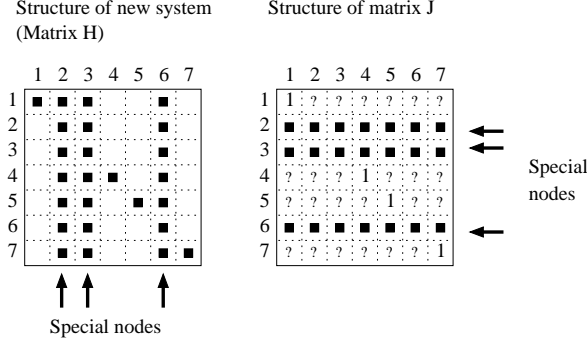


Fig. 8. Structure of matrices H and J for computing the posterior variances. Irrelevant entries in J are marked with question marks.

When message passing has finished, each node i computes vector η_i in the following way

$$\eta_i = \eta_{i,i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}}{d_{k,i}} \eta_{k,i}.$$

The interpretation is that $[\eta_i]_l = [I'_i]_{1, \nu_l}$. With the information collected by each node, we can construct a new system of equations, just as we did for computing the mean, although in this case we perform row operations, not to find the solution of the system, but to find the diagonal elements of \hat{P} .

We define H and β as before, and a matrix $J \in \mathbb{R}^{L \times N}$ such that

$$[J]_{i,j} := \begin{cases} 1 & \text{if } i = j \notin \mathcal{V}, \\ [\eta_j]_l & \text{if } i = \nu_l \in \mathcal{V}, \\ 0 & \text{else.} \end{cases}$$

Note that the elements in J that have been defined as zero, are really unknown, but they are unimportant for our purposes. The structure of J is shown in Figure 8. When solving the subsystem of equations, the effect of the row operations on the rows of J have to be recorded. Since information about the matrix J is distributed among all nodes, we need to transmit the information about the row operations performed by the special nodes to each node in G . For this, each special node records a the elimination factor used in each operation in Step 1-b of the algorithm. To be specific, we let each special node ν_k record a vector $f_k \in \mathbb{R}^L$ such that after message passing, f_k contains the elimination factors used by node ν_k during the solution of the subsystem of equations. For doing this, we initialize $f_k \leftarrow (0, 0, \dots, 0)^T$, and add to the

update operations defined in Step 1-b of the algorithm the following rule

$$[f_k]_l \leftarrow -\frac{[h_k]_l}{[h_l]_l}.$$

After message between special nodes has been finished, each special node x_{ν_l} sends f_l , $[h_l]_l$, and P_{ν_l} to every other node in G . Each node i then defines a vector v_i :

$$[v_i]_l = \frac{P_{\nu_l}}{P_i} [\eta_i]_l,$$

so that $[v_i]_l = [I'_{\nu_l}]_{1,i}$. After this, node i reproduces the row operations performed during the solution of the subsystem of equations and finds their effect on v_i (we give the details of the algorithm in the next section). Finally, node x_i can compute its posterior variance as

$$[\hat{P}]_{i,i} = \begin{cases} [v_i]_l & \text{if } i = \nu_l \in \mathcal{V}, \\ \frac{1}{P_i} (1 - \theta_i^T v_i) & \text{if } i \notin \mathcal{V}. \end{cases}$$

In the next section we provide pseudocodes for the algorithms presented.

10 The complete algorithms for computing the means and variances in loopy graphs

To summarize the complete procedure, we provide the algorithm in toto. We present the algorithm that computes the posterior means and variances. If only the means are desired, the part of the code that computes the variances can be omitted. We present two versions of the algorithm: a centralized version that can be implemented on a single processor, and a distributed one that can be implemented in a distributed fashion on, for example, a sensor network where each node has a measurement y_i .

Before applying the algorithm it is necessary to find a spanning subtree T of G , and the set of special nodes \mathcal{V} . In the centralized version of the algorithm, this can be done by specifying a root node x_r , and applying Dijkstra's algorithm to find the shortest path from every node to x_r ; see, for example, [2]. In the distributed implementation of the algorithm, each node can run Dijkstra's algorithm to find T . Since every node executes the same (deterministic) algorithm, each one finds the same spanning subtree T . Another approach, is to execute a distributed Bellman-Ford algorithm before starting message

passing. This reduces the amount of information that each node requires. For a detailed presentation of Dijkstra's, Bellman-Ford, and other algorithms to solve the shortest path problem, see [2].

To order the computations in the distributed algorithm we require that each node has a unique identifying number. This is specially important when solving the subsystem of equations.

The presentation of the algorithms follows. The algorithms are presented as pseudo-code.

10.1 Centralized algorithm

We now present a centralized algorithm that can be implemented on a single processor.

Begin centralized algorithm

Input: A graph $G = (V_G, E_G)$ on N nodes $\{x_i\}_{i=1}^N$.

A prespecified root node r .

For each $1 \leq i \leq N$, a Gaussian kernel: $\Psi_{i,i}(x_i, y_i)$ (equation 3).

For each $(i, j) \in E_G$, a Gaussian kernel: $\Psi_{i,j}(x_i, x_j)$ (equation 3).

A vector of observations, $y \in \mathbb{R}^N$.

Output: The posterior mean $[\hat{x}]_i$, and variance $[\hat{P}]_{i,i}$ of each x_i .

Find a spanning subtree $T = (V_G, E_T)$ of G , for example by running Dijkstra's algorithm on G with root r .

Define:

$$\mathcal{E} := E_G \setminus E_T,$$

$$\mathcal{V} := \text{non-isolated vertices in } G - T.$$

For each node $i \in \{1, \dots, N\}$ define:

$$N_i := \{1 \leq j \leq N : (i, j) \in E_T\},$$

$$N_{i,j} := N_{i,j} \setminus \{j\},$$

$$[\eta_{i,i}]_l := \begin{cases} 1 & \text{if } i = \nu_l, \\ 0 & \text{else.} \end{cases}$$

$$[\theta_{i,i}]_j := \begin{cases} \sum_{(k,i) \in \mathcal{E}} Z_{x_k, x_i}^{x_i, x_i} & \text{if } i = j \in \mathcal{V}, \\ Z_{x_i, x_j}^{x_i, x_j} & \text{if } i, j \in \mathcal{V} \text{ and } i \neq j, \\ 0 & \text{else.} \end{cases}$$

Repeat

For each node $i \in \{1, \dots, N\}$

For each node $j \in N_i$

If all messages from nodes in $N_{i,j}$ have been received, compute:

$$\begin{aligned} P_0^{i,j} &= Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_{i,j}} P_{k,i}, \\ P_0^{i,j} \mu_0^{i,j} &= -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_{i,j}} P_{k,i} \mu_{k,i}, \\ P_{i,j} &= Z_{x_i, x_j}^{x_j, x_j} - \frac{(Z_{x_i, x_j}^{x_i, x_j})^2}{Z_{x_i, x_j}^{x_i, x_i} + P_0^{i,j}}, \\ P_{i,j} \mu_{i,j} &= -\frac{Z_{x_i, x_j}^{x_i, x_j}}{Z_{x_i, x_j}^{x_i, x_i} + P_0^{i,j}} P_0^{i,j} \mu_0^{i,j}, \\ \theta_{i,j} &= \theta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \theta_{k,i}, \\ \eta_{i,j} &= \eta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \eta_{k,i}. \end{aligned}$$

Until every node has received messages from all its neighbors.

For each node $i \in \{1, \dots, N\}$ compute:

$$\begin{aligned} P_i &= Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_i} P_{k,i}, \\ P_i \mu_i &= -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_i} P_{k,i} \mu_{k,i}, \\ \theta_i &= \theta_{i,i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \theta_{k,i}, \\ \eta_i &= \eta_{i,i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \eta_{k,i}. \end{aligned}$$

For each node ν_l , $l \in \{1, \dots, L\}$ define $h_l, f_l \in \mathbb{R}^L$, and $\gamma_l \in \mathbb{R}$ through:

$$\begin{aligned} f_l &:= (0, 0, \dots, 0)^T, \\ \gamma_l &:= P_{\nu_l} \mu_{\nu_l}, \\ [h_l]_k &:= \begin{cases} P_{\nu_l} + [\theta_{\nu_l}]_l & \text{if } k = l, \\ [\theta_{\nu_l}]_k & \text{if } k \neq l. \end{cases} \end{aligned}$$

For $l = 1$ to L do

For each $k \in \{1, \dots, L\} \setminus \{l\}$ update:

$$\gamma_k \leftarrow \gamma_k - \gamma_l \frac{[h_k]_l}{[h_l]_l},$$

$$[f_k]_l \leftarrow -\frac{[h_k]_l}{[h_l]_l},$$

$$h_k \leftarrow h_k - h_l \frac{[h_k]_l}{[h_l]_l}.$$

For $l = 1$ to L compute:

$$[\hat{x}]_{\nu_l} = \frac{\gamma_l}{[h_l]_l}.$$

For each node $i \notin \mathcal{V}$ compute:

$$[\hat{x}]_i = \frac{1}{P_i} \left(P_i \mu_i - \sum_{l=1}^L [\theta_i]_l [\hat{x}]_{\nu_l} \right).$$

For $i = 1$ to N initialize v_i through:

$$[v_i]_l := \frac{P_{\nu_l}}{P_i} [\eta_i]_l.$$

For $l = 1$ to L do

For $k = 1$ to L update:

$$[v_i]_k \leftarrow [v_i]_k + [v_i]_l [f_k]_l.$$

For $l = 1$ to L update:

$$[v_i]_l \leftarrow \frac{[v_i]_l}{[h_l]_l}.$$

If $i = \nu_l \in \mathcal{V}$ compute:

$$[\hat{P}]_{i,i} = [v_i]_l,$$

else compute:

$$[\hat{P}]_{i,i} = \frac{1}{P_i} \left(1 - \theta_i^T v_i \right).$$

End algorithm.

10.2 Distributed algorithm

We now present a distributed algorithm which can be implemented by nodes passing messages to each other as in a sensor network.

The following algorithm is executed at each node:

Begin distributed algorithm

Input: An identity number i .

The identity of a prespecified root node r .

A graph $G = (V_G, E_G)$ and the identity of every other node in G .

Observation $y_i \in \mathcal{R}$.

A Gaussian kernel: $\Psi_{i,i}(x_i, y_i)$.

For each $(i, j) \in E_G$, a Gaussian kernel: $\Psi_{i,j}(x_i, x_j)$ (equation 3).

Output: The posterior mean $[\hat{x}]_i$, and variance $[\hat{P}]_{i,i}$ (equation 3).

Find a spanning subtree $T = (V_G, E_T)$ of G , using, for example Dijkstra's algorithm on T with root node r .

Define:

$$\mathcal{E} := E_G \setminus E_T,$$

$$\mathcal{V} := \text{non-isolated vertices in } G - T,$$

$$N_i := \{1 \leq j \leq N : (i, j) \in E_T\},$$

$$N_{i,j} := N_i \setminus \{j\},$$

$$[\theta_{i,i}]_j := \begin{cases} \sum_{(k,i) \in \mathcal{E}} Z_{x_k, x_i}^{x_i, x_i} & \text{if } i = j \in \mathcal{V}, \\ Z_{x_i, x_j}^{x_i, x_j} & \text{if } i, j \in \mathcal{V} \text{ and } i \neq j, \\ 0 & \text{else.} \end{cases}$$

$$[\eta_{i,i}]_l := \begin{cases} 1 & \text{if } i = \nu_l, \\ 0 & \text{else.} \end{cases}$$

Repeat

For $j \in N_i$ do

If all messages from nodes in $N_{i,j}$ have been received, compute:

$$P_0^{i,j} = Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_{i,j}} P_{k,i},$$

$$P_0^{i,j} \mu_0^{i,j} = -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_{i,j}} P_{k,i} \mu_{k,i},$$

$$P_{i,j} = Z_{x_i, x_j}^{x_j, x_j} - \frac{(Z_{x_i, x_j}^{x_i, x_j})^2}{Z_{x_i, x_j}^{x_i, x_i} + P_0^{i,j}},$$

$$P_{i,j} \mu_{i,j} = -\frac{Z_{x_i, x_j}^{x_i, x_j}}{Z_{x_i, x_j}^{x_i, x_i} + P_0^{i,j}} P_0^{i,j} \mu_0^{i,j},$$

$$\theta_{i,j} = \theta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \theta_{k,i},$$

$$\eta_{i,j} = \eta_{i,i} - \sum_{k \in N_{i,j}} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k,i}} \eta_{k,i}.$$

Send these quantities to j .

Until all messages from nodes in N_i have been received.

Compute:

$$P_i = Z_{x_i, y_i}^{x_i, x_i} + \sum_{k \in N_i} P_{k, i},$$

$$P_i \mu_i = -Z_{x_i, y_i}^{x_i, y_i} y_i + \sum_{k \in N_i} P_{k, i} \mu_{k, i},$$

$$\theta_i = \theta_{i, i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k, i}} \theta_{k, i},$$

$$\eta_i = \eta_{i, i} - \sum_{k \in N_i} \frac{Z_{x_k, x_i}^{x_k, x_i}}{Z_{x_k, x_i}^{x_k, x_k} + P_0^{k, i}} \eta_{k, i}.$$

If $i = \nu_k \in \mathcal{V}$ is special, define $h_l, f_l \in \mathbb{R}^L$, and $\gamma_l \in \mathbb{R}$ through:

$$f_l := (0, 0, \dots, 0)^T,$$

$$\gamma_l := P_{\nu_l} \mu_{\nu_l},$$

$$[h_l]_k := \begin{cases} P_{\nu_l} + [\theta_{\nu_l}]_l & \text{if } k = l, \\ [\theta_{\nu_l}]_k & \text{if } k \neq l. \end{cases}$$

Repeat

Wait for γ_l, f_l , and h_l sent by special nodes with $l < k$.

Update

$$\gamma_k \leftarrow \gamma_k - \gamma_l \frac{[h_k]_l}{[h_l]_l},$$

$$f_k \leftarrow -\frac{[h_k]_l}{[h_l]_l},$$

$$h_k \leftarrow h_k - h_l \frac{[h_k]_l}{[h_l]_l}.$$

Until x_{ν_k} is the special node with least index that has not sent its information.

Send γ_k, f_k , and h_k to every other special node.

Repeat

Wait for γ_l, f_l , and h_l sent by special nodes with $l > k$.

Update

$$\gamma_k \leftarrow \gamma_k - \gamma_l \frac{[h_k]_l}{[h_l]_l},$$

$$f_k \leftarrow -\frac{[h_k]_l}{[h_l]_l},$$

$$h_k \leftarrow h_k - h_l \frac{[h_k]_l}{[h_l]_l}.$$

Until messages from all nodes in \mathcal{V} have been received.

Compute:

$$[\hat{x}]_i = \frac{\gamma_k}{[h_k]_k}.$$

Send $[\hat{x}]_i$ to every nonspecial node in G .

Send $P_i, [h_k]_k$, and f_k to every other node in G .

else

Computation	Number of operations		
	M.P. in tree	E.M.P. means	E.M.P. means and variances
$P_0^{i,j}, P_0^{i,j} \mu_0^{i,j},$ $P_{i,j}, P_{i,j} \mu_{i,j}$	$O(\Delta N)$	$O(\Delta N)$	$O(\Delta N)$
$\theta_{i,j}$	-	$O(\Delta NL)$	$O(\Delta NL)$
$\eta_{i,j}$	-	-	$O(\Delta NL)$
$P_i, P_i \mu_i$	$O(N)$	$O(N)$	$O(N)$
θ_i	-	$O(NL)$	$O(NL)$
η_i	-	-	$O(NL)$
Solution of subsystem	-	$O(L^2)$	$O(NL^2)$
Backsubstitution	-	$O((N-L)L)$	$O((N-L)L)$
TOTAL	$O(\Delta N)$	$O(\Delta NL)$	$O(NL^2)$

Table 1

Comparison of complexities of message passing (M.P.), extended message passing (E.M.P.) for means, and extended message passing for means and variances.

Wait for $[\hat{x}]_{\nu_l}$ sent by each special node ν_l .

Compute:

$$[\hat{x}]_i = \frac{1}{P_i} \left(P_i \mu_i - \sum_{j=1}^L [\theta_i]_j [\hat{x}]_{\nu_j} \right).$$

Wait to receive P_{ν_l} , $[h_l]_l$, and f_l from every special node $\nu_l \in \mathcal{V}$.

Define v_i through:

$$[v_i]_l := \frac{P_{\nu_l}}{P_i} [\eta_i]_l.$$

For $l = 1$ to L do

For $k = 1$ to L update:

$$[v_i]_k \leftarrow [v_i]_k + [v_i]_l [f_k]_l.$$

For $l = 1$ to L update:

$$[v_i]_l \leftarrow \frac{[v_i]_l}{[h_l]_l}.$$

If $i = \nu_k \in \mathcal{V}$ compute:

$$[\hat{P}]_{i,i} = [v_i]_k,$$

else compute:

$$[\hat{P}]_{i,i} = \frac{1}{P_i} (1 - \theta_i^T v_i).$$

End algorithm.

11 Complexity of the algorithms

In this section we consider the complexity of the complete algorithms and compare them to the complexity of message passing in trees.

Table 1 shows the order of the number of operations required by the relevant

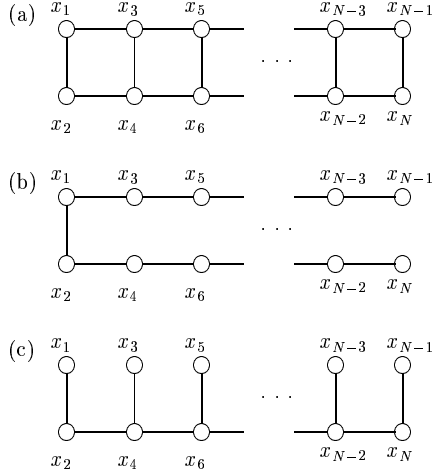


Fig. 9. Example of a judicious choice of spanning subtree. The tree in Figure (b) produces roughly twice the number of special nodes compared to the tree in Figure (c).

operations in each algorithm. In this table Δ represents the maximum degree of nodes in the spanning subtree T . Note that constants have been neglected. For example, there are two vectors $\theta_{i,j}$ to be computed for each edge. Since T , is a tree, there are $N - 1$ edges. Each of these computations requires of the order of $O(\Delta L)$ operations. Thus, the number of operations necessary to compute all vectors $\theta_{i,j}$ is $O(\Delta NL)$.

From Table 1 we see that the total complexity of message passing in a tree is $O(\Delta N)$, while the complexities of the extended algorithms are $O(\Delta NL)$ for the means, and $O(NL^2)$ for means and variances. We see thus that the complexity of computing the means grows linearly in the number of special nodes, while the complexity of computing the variances grows quadratically.

The fact that the number of operations required to solve the inference problem grows gradually when the complexity of the graph is increased is intuitively satisfying, and justifies the use of the extended algorithms in graphical models whose underlying graph is “close” to a tree.

It is useful to note that by a judicious choice of the spanning subtree T , one can reduce L significantly. The following is an example.

Example

Consider the grid in Figure 9(a). Figures 9(b) and 9(c) show two different spanning subtrees for the graph in Figure 9(a). The number of special edges for the tree in Figure 9(b) is $N - 2$, while the number of special edges for

the tree in Figure 9(c) is $N/2$. We can see that the choice of T can affect the complexity of the algorithm.

12 Concluding remarks

The use of the standard message passing for a tree, directly in a graph with cycles, yields a recursive algorithm whose convergence is not guaranteed. In fact, whether it converges and, if it converges the rate of convergence, depend not only on the topology of the graph, but also on the parametrization of the joint probability distribution. As opposed to this, the extended message passing algorithms presented are guaranteed to terminate in finite time for *any* parametrization, and for *any* graph. Moreover, the complexity of the algorithms increases gradually with the number of special nodes L . A judicious choice of the spanning subtree can further significantly reduce L . The complexity of the extended message passing algorithm for computing the posterior means only in loopy graphs is $O(\Delta NL)$, while the complexity of the algorithm for computing both the posterior means and variances is $O(NL^2)$.

A drawback of the new algorithms is that each node needs information about the structure of the graph. To apply traditional message passing, each node needs to know only its neighbors. In the extended algorithms, each node needs to know which of its incident edges are in the spanning subtree T and which nodes are special. Moreover, when solving the subsystem of equations we assumed a fixed ordering among the special nodes. These issues do not represent difficulties when the algorithm is applied on a single computer, but proper scheduling of messages is required when the algorithm is to be applied in a distributed fashion over a network.

It is an intriguing question whether ideas such as these can be applied to solve the problem of inference for general loopy graphical models without Gaussianity assumptions.

References

- [1] S. M. Aji, J. McEliece, The generalized distributive law, *IEEE Transactions on Information Theory* 46 (1999) 325–343.
- [2] D. Bertsekas, J. Tsitsiklis, *Parallel and distributed computation*, Englewood Cliffs, N.J. :Prentice Hall, 1989.

- [3] J. Besag, Spatial interaction and the statistical analysis of lattice systems, *Journal of Royal Statistical Society, Series B* 36 (1974) 192–223.
- [4] K. Chou, A. S. Willsky, A. B. Beveniste, Multiscale recursive estimation, data fusion, and regularization, *IEEE Trans. Automat. Contr.* 39 (1994) 464–478.
- [5] K. Chou, A. S. Willsky, R. Nikoukhah, Multiscale systems, Kalman filters, and Riccatti equations, *IEEE Trans. Automat. Contr.* 39 (1994) 479–492.
- [6] B. Frey, A revolution: Belief propagation in graphs with cycles, *Advances in Neural Processing Systems* 10.
- [7] B. J. Frey, *Graphical models for machine learning and digital communication*, Cambridge, MA: MIT Press, 1998.
- [8] G. H. Golub, C. F. V. Loan, *Matrix Computations*, Baltimore: Johns Hopkins University Press, 1983.
- [9] M. Jordan, C. Bishop, *Introduction to Graphical Models*, Preprint 2001.
- [10] P. R. Kumar, P. Varaiya, *Stochastic systems : estimation, identification, and adaptive control*, Englewood Cliffs, N.J. : Prentice Hall, 1986.
- [11] R. Nokoukhah, A. S. Willsky, B. Levy, Boundary value descriptor systems: well posedness, reachability and observability, *International Journal of Control*, 46 (5) (1987) 1715–1737.
- [12] J. Pearl, *Probabilistic reasoning in intelligent systems. Networks of plausible inference*, Morgan Kaufmann, 1988.
- [13] P. Rusmevichientong, B. Van Roy, An analysis of belief propagation on the turbo decoding graph with Gaussian densities, *IEEE Transactions on Information Theory* 47 (2) (2001) 745–765.
- [14] E. Sudderth, *Embedded trees: Estimation of Gaussian processes on graphs with cycles*. MS thesis, Massachussets Institute of Technology, February 2002.
- [15] M. Wainwright, E. Sudderth, A. S. Willsky, Tree-based modeling and estimation of Gaussian processes on graphs with cycles, *Advances in Neural Information Processing Systems* 13.
- [16] Y. Weiss, W. T. Freeman, Correctness of belief propagation in Gaussian graphical models of arbitrary topology, *Neural Computation* 13 (2001) 2173–2200.
- [17] A. S. Willsky, Multiresolution Markov models for signal and image processing, *Proceedings of the IEEE* 90 (8) (2002) 1396–1458.