Feedback Message Passing for Inference in Gaussian Graphical Models

Ying Liu, Venkat Chandrasekaran, Animashree Anandkumar, and Alan S. Willsky

Stochastic Systems Group, LIDS, MIT, Cambridge, MA 02139
Email:{liu_ying,venkatc,animakum,willsky}@mit.edu

***Abstract*—For Gaussian graphical models with cycles, loopy belief propagation often performs reasonably well, but its convergence is not guaranteed and the computation of variances is generally incorrect. In this paper, we identify a set of special vertices called a feedback vertex set whose removal results in a cycle-free graph. We propose a *feedback message passing* algorithm in which non-feedback nodes send out one set of messages while the feedback nodes use a different message update scheme. Exact inference results can be obtained in $\mathcal{O}(k^2n)$, where k is the number of feedback nodes and n is the total number of nodes. For graphs with large feedback vertex sets, we describe a tractable approximate feedback message passing algorithm. Experimental results show that this procedure converges more often, faster, and provides better results than loopy belief propagation.**

***Index Terms*—Gaussian graphical models, belief propagation, loopy graphs, feedback vertex set**

I. Introduction

In graphical models each node represents a random variable and the edge structure specifies the conditional independence or Markov properties of the underlying distribution [1]. Such models are widely used in many fields such as computer vision, gene regulatory networks, oceanography, and medical diagnostics. Although inference in Gaussian graphical models can be solved by direct matrix inversion, it is intractable for very large problems involving millions of random variables [2]. Therefore, it is of great importance to develop efficient inference algorithms.

Belief propagation (BP) is an efficient message passing algorithm that gives exact inference results in linear time for tree-structured graphs. However, trees possess limited modeling capabilities, and many real world processes cannot be modeled using graphs without cycles.

For inference in loopy graphs, loopy belief propagation (LBP) can be used as a direct extension of BP by following the same local message passing rules. It turns out that LBP performs reasonably well for certain loopy graphs [3]. However, the convergence and correctness of LBP are not guaranteed in general, and many studies have been conducted on the performance of LBP [4]–[7]. For Gaussian graphical models if LBP converges, the means converge to the correct values while the variances are generally incorrect [5]. In [7] an analysis framework based on walk-sums is proposed to analyze the performance of LBP in Gaussian graphical models.

A desirable property of LBP is that it is completely distributed. However, LBP has its limitations: only local information is used in updating messages and all nodes are treated equally. Global information of the cyclic structure of the graph is not captured and thus errors and convergence problems may occur. One can ask some natural questions: can we use some more memory to keep track of the messages or use some header information to denote the sources of the messages? Are there some nodes that are more important in terms of inference? Can we design an algorithm accordingly without losing too much decentralization?

We consider a particular set of "important" nodes called the feedback vertex set. A feedback vertex set is a subset of vertices that breaks all the cycles in the graph. Based on this concept, we propose an algorithm for Gaussian graphical models. The algorithm includes several message passing steps. The whole procedure takes linear time to obtain the exact means and variances for all nodes if the number of feedback nodes is bounded by a constant. When this number is large, we use an approximate feedback message passing algorithm to obtain approximate inference results, which trades off between efficiency and accuracy.

II. Background

A. Gaussian Graphical Models

A Gaussian distribution is given by $p(\mathbf{x}) \propto \exp\{-\frac{1}{2}\mathbf{x}^T J\mathbf{x} + \mathbf{h}^T\mathbf{x}\}$, where J is called the *information*, *precision* or *concentration matrix* and $\mathbf{h}$ is called the *potential vector*. The relationship with the mean $\boldsymbol{\mu} = \mathbb{E}\{\mathbf{x}\}$ and the covariance matrix $P = \mathbb{E}\{(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^T\}$ is given by $\boldsymbol{\mu} = J^{-1}\mathbf{h}$ and $P = J^{-1}$. For a valid probability distribution, J is symmetric and positive definite.

In a Gaussian graphical model, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is used to represent the underlying structure, where $\mathcal{V}$ indexes the variables and $\mathcal{E}$ specifies the conditional independence [1]. If there is no edge between two nodes, the corresponding variables are independent conditioned on all other variables. The information matrix J is sparse with respect to the graph $\mathcal{G}$: $\forall (i,j) \notin \mathcal{E}$, $J_{ij} = 0$, which means the conditional properties can be read immediately from the matrix J. Inference in Gaussian graphical models is the problem of calculating the variance P_{ii} and the mean μ_i for every node i given J and $\mathbf{h}$.

This work is supported in part by ARO under MURI Grant W911NF-06-1-0076, by AFOSR under Grant FA9550-08-1-1080 and MURI Grant FA9550-06-1-0324.

B. Belief Propagation on Trees and Loopy Graphs

BP is a distributed message passing algorithm that passes messages between neighboring nodes [8]. Each message is updated according to the messages received from other neighbors. After the messages converge, each node calculates its own variance and mean based on the incoming messages.

BP on tree-structured models is guaranteed to converge in a finite number of steps and gives the exact inference results in linear time. For Gaussian graphical models, the messages can be represented in terms of information parameters. Here is a summary:

Step 1: Message Passing

Each node i sends messages $\Delta J_{i\to j}$ and $\Delta h_{i\to j}$ to every $j \in \mathcal{N}(i)$, where $\mathcal{N}(i)$ denotes the set of i's neighbors:

$$\Delta J_{i\to j} = -J_{ji}\hat{J}_{i\backslash j}^{-1}J_{ij},\ \Delta h_{i\to j} = -J_{ji}\hat{J}_{i\backslash j}^{-1}\hat{h}_{i\backslash j},$$

where

$$\hat{J}_{i\backslash j} = J_{ii} + \sum_{k\in\mathcal{N}(i)\backslash j}\Delta J_{k\to i},\ \hat{h}_{i\backslash j} = h_i + \sum_{k\in\mathcal{N}(i)\backslash j}\Delta h_{k\to i}.$$

Step 2: Marginal Computation

After the messages converge, every node calculates

$$\hat{J}_i = J_{ii} + \sum_{k\in\mathcal{N}(i)}\Delta J_{k\to i},\quad \hat{h}_i = h_i + \sum_{k\in\mathcal{N}(i)}\Delta h_{k\to i},$$

which can be converted to the mean and variance by $\mu_i = \hat{J}_i^{-1}\hat{h}_i$ and $P_{ii} = \hat{J}_i^{-1}$.

Loopy belief propagation is a direct extension of BP for loopy graphs. It uses the same message update rule locally as BP and neglects the existence of cycles. LBP is not guaranteed to converge in general; if it does converge, it gives the exact means but inaccurate variances.

C. Feedback Vertex Set

A feedback vertex set (FVS), sometimes also called a loop cutset, is defined as a set of vertices whose removal results in an acyclic graph [9]. For instance, node 1 in Fig 1(a) forms an FVS by itself.

For a general graph, finding the FVS of the *minimum* size (the minimum FVS) is proved to be NP-complete [10]. However, for many special graph structures, optimal or near optimal solutions can be found efficiently or even in linear time [11]–[13]. In addition, for general graphs there exists an efficient approximate algorithm to find an FVS with size at most twice the minimum size [14].

In this paper we use $\mathcal{F}$ to denote an FVS and call the nodes in $\mathcal{F}$ the feedback nodes. We use $\mathcal{T} = \mathcal{V}\backslash\mathcal{F}$ to denote the non-feedback nodes. The subgraph induced by $\mathcal{T}$ can either be a tree or a forest.

III. Feedback Message Passing

The high level idea of the feedback message passing algorithm is to obtain inference results for the feedback nodes first and make corrections for the non-feedback nodes later. First we start with the case in which a single feedback node breaks all the cycles. Then we describe the general algorithm when multiple feedback nodes are used.

A. The Single Feedback Node Case

Consider the loopy graph shown in Fig. 1(a). Let J and $\mathbf{h}$ be the information matrix and potential vector respectively. In this graph every cycle passes through node 1, which is thus a feedback node for the graph. Let $\mathcal{N}(1)$ denote the set of neighboring nodes of node 1 and $\mathcal{T}$ denote the subgraph excluding node 1. The feedback message passing algorithm has the following steps:

Step 1: Initialization

A new potential vector $\mathbf{h}^1$ on $\mathcal{T}$ is constructed, where $h_i^1 = J_{1i}$, $\forall i \in \mathcal{N}(1)$ and $h_i^1 = 0$, $\forall i \notin \mathcal{N}(1), i \neq 1$. This new potential vector captures some of node 1's effects on $\mathcal{T}$ so that nodes in $\mathcal{T}$ can process this information. See Fig. 1(b) for illustration.

Step 2: First round of BP

BP is performed on $\mathcal{T}$ with $J_\mathcal{T}$ and $\mathbf{h}_\mathcal{T}$, where $J_\mathcal{T}$ and $\mathbf{h}_\mathcal{T}$ are the corresponding submatrix and subvector of J and $\mathbf{h}$ respectively. After convergence each node i obtains its "partial variance" $P_{ii}^\mathcal{T}$ and its "partial mean" $\mu_i^\mathcal{T}$. Note that these results are not accurate since they only capture local structures within $\mathcal{T}$ without considering the effects of node 1.

The information of node 1 is calculated by performing BP on $\mathcal{T}$ with the information matrix $J_\mathcal{T}$ and the new potential vector $\mathbf{h}^1$. Each node i on $\mathcal{T}$ will obtain a *feedback gain* g_i^1, where $g_i^1 = (J_\mathcal{T}^{-1}\mathbf{h}^1)_i$ given by BP.

In practice we run BP only once with one information matrix $J_\mathcal{T}$ and two potential vectors $\mathbf{h}_\mathcal{T}$ and $\mathbf{h}^1$. We also put the header information "1" into the messages related to $\mathbf{h}^1$ to denote the source of the messages. Therefore, each node on $\mathcal{T}$ knows the messages for the "partial variance" and "partial mean", as well as the messages for the feedback gain.

Step 3: Inference for the feedback node

The feedback node 1 collects the feedback gains from its neighbors as shown in Fig. 1(d). Node 1 then calculates the variance and mean for itself:

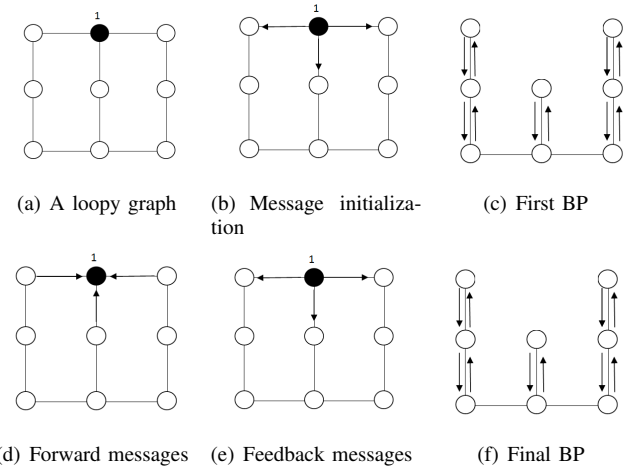
$$\begin{aligned} P_{11} &= (J_{11} - \sum_{j\in\mathcal{N}(1)} J_{1j}g_j^1)^{-1}, \\ \mu_1 &= P_{11}(h_1 - \sum_{j\in\mathcal{N}(1)} J_{1j}\mu_j^\mathcal{T}). \end{aligned}$$

These two results are the *exact* variance and *exact* mean for node 1. The exactness results from the fact that node 1 breaks all the cycles in the graph, and can be proved by matrix manipulation.

Step 4: Revising the potential vector

After the feedback node 1 obtains its own variance and mean, it passes the results to other nodes in order to correct their inaccurate "partial variances" $P_{ii}^\mathcal{T}$ and "partial means" $\mu_i^\mathcal{T}$ as computed in Step 2 (see Fig. 1(e)). The neighbors of node 1 revise their node potentials as follows:

$$\widetilde{h}_j = \begin{cases} h_j - J_{1j}\mu_1, & \forall j \in \mathcal{N}(1) \\ h_j, & \forall j \notin \mathcal{N}(1) \end{cases}$$



(a) A loopy graph (b) Message initialization (c) First BP

(d) Forward messages (e) Feedback messages (f) Final BP

Fig. 1. A feedback message passing example

The revised potential vector $\widetilde{\mathbf{h}}_{\mathcal{T}}$ will be used in another round of BP.

Step 5: Final round of BP

BP is performed on $\mathcal{T}$ with $J_{\mathcal{T}}$ and the revised potential vector $\widetilde{\mathbf{h}}_{\mathcal{T}}$ (see Fig. 1(f)). The means we obtain are the *exact* means. The *exact* variances can be computed by

$$P_{ii} = P_{ii}^{\mathcal{T}} + P_{11}(g_i^1)^2, \forall i \in \mathcal{T},$$

where $P_{ii}^{\mathcal{T}}$ is the inaccurate "partial variance" computed in Step 2 and g_i^1 is the feedback gain computed in Step 3.

The results are exact because node 1 breaks all the cycles. The feedback messages from node 1 cancel out the cyclic effects caused by node 1 by revising the potential vector on $\mathcal{T}$ and adding correction terms.

B. Feedback Message Passing for General Graphs

For general graphs an FVS may have multiple nodes. In [14] a factor 2 approximate algorithm is proposed to find an FVS of size at most two times the minimum size.

The feedback message passing algorithm with multiple feedback nodes is essentially the same as the single feedback node case. Without loss of generality, we order the nodes such that the feedback nodes are the first k nodes, where k is the size of the FVS. Here we briefly explain the differences and summarize the algorithm in Fig. 2.

In Step 1 and Step 2, the difference is that k extra potential vectors similarly defined are used instead of just one. In Step 3, solving an inference problem on a graph with k nodes is required. In Step 4 and Step 5, the variances are corrected by adding multiple correction terms corresponding to all the feedback nodes.

C. Accuracy and Complexity

The feedback message passing algorithm described in Fig. 2 gives exact means and variances. We have the following result:

Result 1: The feedback message passing algorithm converges in $\mathcal{O}(k^2 n)$ time and gives the exact means and variances for all nodes, where k is the size of the FVS and n is the total number of nodes.

Input: information matrix J, potential vector $\mathbf{h}$ and feedback vertex set $\mathcal{F}$ of size k

Output: mean μ_i and variance P_{ii} for every node i

1. Construct k extra potential vectors: $\forall p \in \mathcal{F}, \mathbf{h}^p = J_{\mathcal{T},p}$, each corresponding to one feedback node.
2. Perform BP on $\mathcal{T}$ with $J_{\mathcal{T}}$, $h_{\mathcal{T}}$ to obtain $P_{ii}^{\mathcal{T}} = (J_{\mathcal{T}}^{-1})_{ii}$ and $\mu_i^{\mathcal{T}} = (J_{\mathcal{T}}^{-1}\mathbf{h}_{\mathcal{T}})_i$ for each $i \in \mathcal{T}$. With the k extra potential vectors, calculate the feedback gains $g_i^1 = (J_{\mathcal{T}}^{-1}\mathbf{h}^1)_i, g_i^2 = (J_{\mathcal{T}}^{-1}\mathbf{h}^2)_i, \ldots, g_i^k = (J_{\mathcal{T}}^{-1}\mathbf{h}^k)_i$ for $i \in \mathcal{T}$ by BP .
3. Obtain a size k subgraph with $\widehat{J}_{\mathcal{F}}$ and $\widehat{\mathbf{h}}_{\mathcal{F}}$ given by
$$(\widehat{J}_{\mathcal{F}})_{pq} = J_{pq} - \sum_{j \in \mathcal{N}(p) \cap \mathcal{T}} J_{pj} g_j^q, \ \forall p, q \in \mathcal{F}$$
$$(\widehat{\mathbf{h}}_{\mathcal{F}})_p = h_p - \sum_{j \in \mathcal{N}(p) \cap \mathcal{T}} J_{pj} \mu_j^{\mathcal{T}}, \ \forall p \in \mathcal{F},$$
and solve the inference problem on the small graph by $P^{\mathcal{F}} = \widehat{J}_{\mathcal{F}}^{-1}, \quad \boldsymbol{\mu}^{\mathcal{F}} = \widehat{J}_{\mathcal{F}}^{-1}\widehat{\mathbf{h}}_{\mathcal{F}}$.
4. Revise the potential vector on $\mathcal{T}$ by
$$\widetilde{h}_i = h_i - \sum_{j \in \mathcal{N}(i) \cap \mathcal{F}} J_{ij} \boldsymbol{\mu}_j^{\mathcal{F}}, \ \forall i \in \mathcal{T}.$$
5. Another round of BP with the revised potential vector $\widetilde{\mathbf{h}}_{\mathcal{T}}$ gives the exact means for nodes on $\mathcal{T}$.
Add correction terms to obtain the exact variances for nodes in $\mathcal{T}$:
$$P_{ii} = P_{ii}^{\mathcal{T}} + \sum_{p \in \mathcal{F}} \sum_{q \in \mathcal{F}} g_i^p P_{pq}^{\mathcal{F}} g_i^q, \ \forall i \in \mathcal{T}.$$

Fig. 2. Feedback message passing algorithm for general graphs

The proof is provided in a longer version of this paper, and essentially follows from Gaussian elimination in a carefully designed order.

If the size of the FVS is bounded by a constant, the means and variances can be computed exactly in linear time. If the size of the FVS is unbounded but grows much slower than the graph size, e.g., if the size of the FVS is $\mathcal{O}(\log n)$, the algorithm is still much faster than direct matrix inversion .

As stated in [14] the complexity of the factor 2 approximate algorithm to find an FVS is $\mathcal{O}(\min\{m \log n, n^2\})$, where m is the number of edges. For any graph in which the number of edges grows linearly with the number of nodes, this algorithm takes $\mathcal{O}(n \log n)$ time.

IV. Approximate Feedback Message Passing

For graphs with many cycles, the FVS may have a large size. In such problems the feedback message passing algorithm may not be tractable. A grid graph is such an example, where the size of the FVS grows linearly with the size of the graph [15].

To better trade off between accuracy and efficiency, we consider a subset of an FVS (pseudo-FVS) of some specified

Input: information matrix J and maximum size k
Output: a pseudo-FVS $\widetilde{\mathcal{F}}$ of size at most k

1. Let $\widetilde{\mathcal{F}} = \emptyset$ and normalize J.
2. Repeat until $|\mathcal{F}| = k$ or the remaining graph is empty.
   (a) Clean up the graph by eliminating all the tree branches.
   (b) Update the scores $s(i) = \sum_{j \in \mathcal{N}(i)} |J_{ij}|$.
   (c) Put the node with the largest weight into $\widetilde{\mathcal{F}}$.

Fig. 3. Selecting a pseudo-FVS with bounded size

size. We denote a pseudo-FVS by $\widetilde{\mathcal{F}}$ and still call the nodes in $\widetilde{\mathcal{F}}$ the feedback nodes although $\widetilde{\mathcal{F}}$ is not necessarily an FVS. Similarly, we use $\widetilde{\mathcal{T}}$ to denote $\mathcal{V} \backslash \widetilde{\mathcal{F}}$ although $\widetilde{\mathcal{T}}$ may still have cycles. The only change in the feedback message passing algorithm is to use LBP instead of BP on $\widetilde{\mathcal{T}}$ in Step 2 and Step 5.

A. Finding a Bounded Size pseudo-FVS

We begin by addressing the problem of finding a good pseudo-FVS $\widetilde{\mathcal{F}}$. We should keep in mind that in a loopy graph the problems of divergence and inaccuracy of LBP are caused by the existence of cycles. Therefore, one goal is to ensure convergence while the other goal is to obtain smaller errors. Breaking all the cycles by using a complete but large FVS can clearly achieve both goals together. However, it may lead to intractable algorithms. Therefore, we want to select a small set of nodes whose removal breaks most cycles. We will see later that there is a huge performance difference between a good selection and a bad selection of $\widetilde{\mathcal{F}}$. The approach here is motivated by a sufficient condition for LBP convergence.

Consider a normalized information matrix J with the diagonal entries equal to one. Define $R = I - J$ where I is the identity matrix. Therefore R has zero diagonal entries. Let $\bar{R}$ be the matrix formed with the absolute values of entries in R. A sufficient condition for LBP to converge is $\rho(\bar{R}) < 1$, where $\rho(\bar{R})$ is the spectral radius of $\bar{R}$. A Gaussian graphical model with $\rho(\bar{R}) < 1$ is called *walk-summable* [7].

If we have a subgraph $\widetilde{\mathcal{T}}$ with smaller $\rho(\bar{R}_{\widetilde{\mathcal{T}}})$, where $\bar{R}_{\widetilde{\mathcal{T}}}$ is the corresponding submatrix of $\bar{R}$, LBP on $\widetilde{\mathcal{T}}$ is more likely to converge. A bound on the spectral radius of a nonnegative matrix [16] is given by

$$\min_i \sum_j \bar{R}_{ij} \leq \rho(\bar{R}) \leq \max_i \sum_j \bar{R}_{ij}.$$

Motivated by this inequality, we remove the node i with the largest score $s(i) = \sum_{j \in \mathcal{N}(i)} \bar{R}_{ij}$ from the graph and put it into $\widetilde{\mathcal{F}}$. The remaining graph $\widetilde{\mathcal{T}}$ thus has a smaller upper bound on the spectral radius of the corresponding $\bar{R}_{\widetilde{\mathcal{T}}}$. We continue this procedure on the remaining graph until the maximum allowed size k of $\widetilde{\mathcal{F}}$ is reached or the remaining graph does not have any cycles. The algorithm is summarized in Fig. 3. The complexity of this procedure is $\mathcal{O}(km)$, where m is the number of edges.

B. Convergence and Accuracy

For the convergence and accuracy of the approximate feedback message passing algorithm, we have the following results:

Result 2: If a Gaussian graphical model is walk-summable, the approximate feedback message passing algorithm converges with any selection of feedback nodes.

If the model is not walk-summable, the approximate feedback message passing algorithm with a suitable set of feedback nodes often converges even though LBP does not converge. When both algorithms converge, the approximate feedback message passing algorithm often converges faster.

Result 3: When the approximate feedback message passing algorithm converges, it always gives the correct means for all nodes.

This result is a natural extension of the fact that LBP gives the correct means when it converges.

Result 4: For attractive Gaussian graphical models (i.e. models with only non-negative partial correlation), the approximate feedback message passing algorithm converges with any selection of $\widetilde{\mathcal{F}}$ and the variance estimations are lower bounds of the true variances. With a sequence of increasing pseudo-FVS $\widetilde{\mathcal{F}}_1 \subset \widetilde{\mathcal{F}}_2 \subset \widetilde{\mathcal{F}}_3 \ldots$, the estimated lower bounds also increase and eventually reach the exact variances after a pseudo-FVS becomes an FVS.

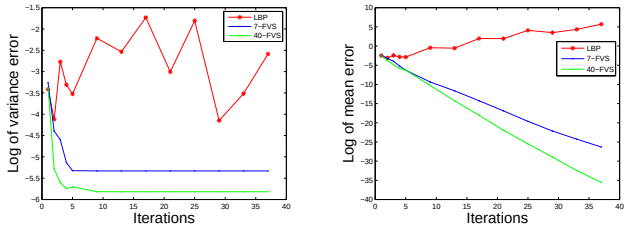
For non-attractive Gaussian graphical models, the situation is more subtle. Based on the walk-sum interpretation of inference in Gaussian graphical models [7], the correct variance at each node corresponds to the sum over a certain set of "walks" in the graph. LBP only captures a subset of these walks, and thus gives inaccurate variance estimates. Our approximate feedback message passing algorithm with any set of pseudo-FVS nodes calculates the sum of a strictly larger set of walks than LBP. In practice, we have observed that variance estimates improve significantly even for non-attractive models. However, since the walks may have both positive and negative weights, capturing more walks does not directly lead to better estimates.

The proofs of the results are omitted here. They are provided in a longer version of this paper.

V. Numerical Results

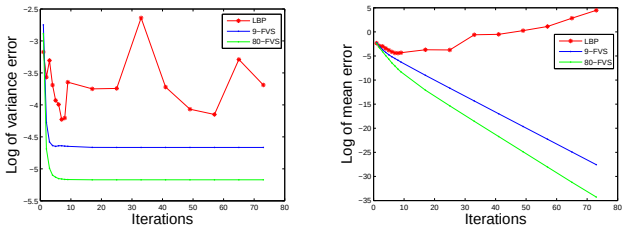
From our Result 1 exact inference on any graph with a small FVS can be solved efficiently. In this section, we focus on the case when the size of the FVS is large. Grid graphs are widely used in computer vision, seismic data modeling, and many other applications. Inference on a grid graph is in general not easy even though the graph is sparse. Here we apply the approximate feedback message passing algorithm and show that it gives good approximate results in a tractable procedure.

Consider $l \times l$ grid graphs with different values of l. The graph size is $n = l^2$. Given a fixed graph structure, we randomly generate an information matrix J, which is sparse with respect to the graph. Its nonzero entries are drawn from an i.i.d. uniform distribution ranging between -1 and 1. We



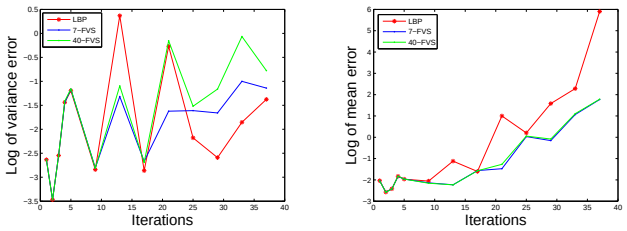
(a) Iterations versus variance errors (b) Iterations versus mean errors

Fig. 4. Inference errors of a 40×40 grid graph



(a) Iterations versus variance errors (b) Iterations versus mean errors

Fig. 5. Inference errors of an 80×80 grid graph



(a) Iterations versus variance errors (b) Iterations versus mean errors

Fig. 6. Inference errors with a bad selection of feedback nodes

also generate a potential vector $\mathbf{h}$ whose entries are also drawn from an i.i.d. uniform distribution ranging between -1 and 1. We ensure the information matrix J is positive definite by adding proper diagonal values. We perform experiments on models with different parameters including many ill-conditioned models (e.g. those whose smallest eigenvalue of J equals 0.02). On each grid graph, LBP and the approximate feedback message passing algorithm with two different feedback sets are used. One set has $k = \lceil \log n \rceil$ feedback nodes while the other has $k = \sqrt{n}$ nodes. The feedback nodes are selected by the algorithm described in Fig. 3. We plot the average errors for both variances and means on a logarithmic scale. We use "k-FVS" to denote the algorithm with k feedback nodes in the figures.

In Fig. 4 and Fig. 5, numerical results are shown for 40×40 and 80×80 grids respectively. In each case, direct LBP fails to converge. With $k = \lceil \log n \rceil$ feedback nodes, our algorithm converges for both grids and gives much better approximations than LBP in fewer iterations. If $k = \sqrt{n}$ feedback nodes are used, we obtain even better approximations but with more computations in each iteration. By performing many more experiments, $k = \lceil \log n \rceil$ feedback nodes seem to be sufficient to give a convergent algorithm and good approximations. The complexity of such a method is thus $\mathcal{O}(n \log^2 n)$.

We also note that making a good selection of feedback nodes is important. In Fig. 6, an opposite criterion is used in selecting the feedback nodes: we choose the node with the smallest score as defined in Fig. 3 instead of the largest. LBP, 7-FVS and 40-FVS algorithms all fail to converge with feedback nodes selected by this criterion. This phenomenon in some sense shows the importance of selecting suitable feedback nodes and the effectiveness of our selection criterion.

VI. Conclusions

The feedback message passing algorithm solves inference problems in a Gaussian graphical model in linear time if the graph has a small FVS. For a graph with a large FVS, the approximate feedback message passing algorithm can be used. By carefully choosing a small number of feedback nodes, very good inference results can be obtained efficiently.

VII. Acknowledgment

We thank Professor Devavrat Shah and Justin Dauwels for many helpful discussions.

References

[1] M. Jordan, "Graphical models," *Statistical Science*, pp. 140–155, 2004.
[2] C. Wunsch and P. Heimbach, "Practical global oceanic state estimation," *Physica D: Nonlinear Phenomena*, vol. 230, no. 1-2, pp. 197–208, 2007.
[3] K. Murphy, Y. Weiss, and M. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *Proceedings of Uncertainty in AI*, 1999, pp. 467–475.
[4] A. Ihler, J. Fisher, and A. Willsky, "Loopy belief propagation: Convergence and effects of message errors," *Journal of Machine Learning Research*, vol. 6, no. 1, p. 905, 2006.
[5] Y. Weiss and W. Freeman, "Correctness of belief propagation in Gaussian graphical models of arbitrary topology," *Neural Computation*, vol. 13, no. 10, pp. 2173–2200, 2001.
[6] S. Tatikonda and M. Jordan, "Loopy belief propagation and Gibbs measures," in *Uncertainty in Artificial Intelligence*, vol. 18, 2002, pp. 493–500.
[7] D. Malioutov, J. Johnson, and A. Willsky, "Walk-sums and belief propagation in Gaussian graphical models," *The Journal of Machine Learning Research*, vol. 7, pp. 2031–2064, 2006.
[8] S. Lauritzen, *Graphical models*. Oxford University Press, USA, 1996.
[9] V. Vazirani, *Approximation algorithms*. Springer, 2004.
[10] R. Karp, "Reducibility among combinatorial problems," *Complexity of computer computations*, vol. 43, pp. 85–103, 1972.
[11] A. Shamir, "A linear time algorithm for finding minimum cutsets in reducible graphs," *SIAM Journal on Computing*, vol. 8, p. 645, 1979.
[12] C. Wang, E. Lloyd, and M. Soffa, "Feedback vertex sets and cyclically reducible graphs," *Journal of the ACM (JACM)*, vol. 32, no. 2, pp. 296–313, 1985.
[13] D. Kratsch, H. Müller, and I. Todinca, "Feedback vertex set on AT-free graphs," *Discrete Applied Mathematics*, vol. 156, no. 10, pp. 1936–1947, 2008.
[14] V. Bafna, P. Berman, and T. Fujito, "A 2-approximation algorithm for the undirected feedback vertex set problem," *SIAM Journal on Discrete Mathematics*, vol. 12, p. 289, 1999.
[15] F. Madelaine and I. Stewart, "Improved upper and lower bounds on the feedback vertex numbers of grids and butterflies," *Discrete mathematics*, vol. 308, no. 18, pp. 4144–4164, 2008.
[16] R. Horn and C. Johnson, *Matrix analysis*. Cambridge University Press, 1990.