Recursive FMP for distributed inference in Gaussian graphical models

Recursive FMP for Distributed Inference in Gaussian Graphical Models

Ying Liu
Dept. of Electrical Engineering & Computer Science
Massachusetts Institute of Technology
Email: liu_ying@mit.edu

Alan S. Willsky
Dept. of Electrical Engineering & Computer Science
Massachusetts Institute of Technology
Email: willsky@mit.edu

Abstract*—For inference in Gaussian graphical models with cycles, loopy belief propagation (LBP) performs well for some graphs, but often diverges or has slow convergence. When LBP does converge, the variance estimates are incorrect in general. The feedback message passing (FMP) algorithm has been proposed to enhance the convergence and accuracy of inference. In FMP, standard LBP is run twice on the subgraph excluding the pseudo-FVS (a set of nodes that breaks most crucial cycles) while nodes in the pseudo-FVS use a different protocol. In this paper, we propose *recursive FMP*, a purely distributed extension of FMP, where all nodes use the same message-passing protocol. An inference problem on the entire graph is recursively reduced to those on smaller subgraphs in a distributed manner. One advantage of this recursive approach compared with FMP is that there is only one active feedback node at a time, so centralized communication among feedback nodes can be turned into message broadcasting from the single feedback node. We characterize this algorithm using walk-sum analysis and provide theoretical results for convergence and accuracy. We also demonstrate the performance using both simulated models on grids and large-scale sea surface height anomaly data.

I. Introduction

An important family of Markov random fields (MRFs) is the family of Gaussian Markov random fields (GMRFs) or Gaussian graphical models. Such models are widely used in medical diagnostics, oceanography, robotic mapping, and gene regulatory networks. For GMRFs of moderate size, exact inference can be solved by algorithms such as direct matrix inversion, Cholesky factorization, and nested dissection, but these algorithms cannot be used for large-scale problems due to the computational complexity [1], [2].

For tree-structured graphs, a message-passing algorithm called belief Propagation (BP) can give exact results in linear time. When there are cycles in the graphs, loopy belief propagation (LBP) is used, where the message-update protocol is the same as BP. LBP is distributed in nature: messages from all nodes are updated in parallel using only local information. However, LBP is not guaranteed to converge or give accurate results [3], [4], [5], [6]. When LBP does converge, only the computed means are exact while the computed variances are incorrect in general. Some extensions to LBP include generalized belief propagation [7], tree-reweighted message passing [8], double-loop belief propagation [9], and relaxed Gaussian belief propagation [10]. LBP in the context of quadratic minimization has also been studied in [11], [12].

In [13], the authors have proposed the feedback message passing (FMP) algorithm. FMP uses a different protocol among a special set of vertices called a *feedback vertex set* or FVS. When the size of the FVS is large, a pseudo-FVS is used instead of an FVS. By performing two rounds of standard LBP among the non-feedback nodes and solving a small inference problem among the feedback nodes, FMP improves the convergence and accuracy significantly compared with running LBP on the entire graph. In addition, choosing the size of the pseudo-FVS enables us to make the trade-off between efficiency and accuracy explicit. FMP is partially distributed, but the algorithm in [13] still requires centralized communication among the feedback nodes. One can ask some natural questions: Is it possible to select the feedback nodes in a purely distributed manner? Can we further eliminate the centralized computations among the feedback nodes in FMP without losing the improvements on convergence and accuracy?

In this paper, we propose *recursive FMP*, a recursive and purely distributed extension of FMP, where all nodes use the same message-passing protocol. In recursive FMP, an inference problem on the entire graph is recursively reduced to those on smaller and smaller subgraphs until the final inference problem can be solved efficiently by an exact or approximate message-passing algorithm. A purely distributed algorithm is of great importance because in many scenarios, such as wireless sensor networks, it is easy to implement the same protocol on all nodes while centralized computations are often expensive or impractical. In this recursive approach, there is only one active feedback node at a time, and thus centralized communication among feedback nodes in FMP is reduced to message broadcasting[1] from the single feedback node.

II. Preliminaries

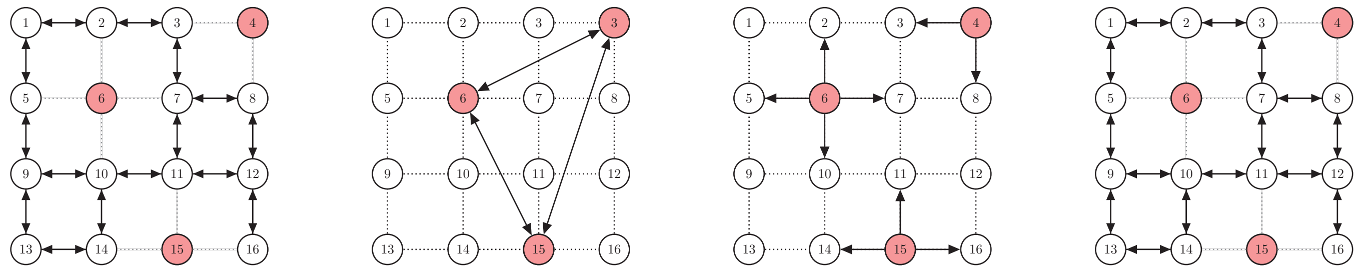
A. Gaussian Graphical Models

The conditional independencies among a set of random variables in an MRF can be modeled by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of vertices (nodes) and $\mathcal{E}$ the set of edges [14]. Each random variable $\mathbf{x}_s$ is represented by a node $s \in \mathcal{V}$ in the graph. The random vector $\mathbf{x}_\mathcal{V} = \{\mathbf{x}_s | s \in \mathcal{V}\}$ is Markov with respect to the graph: for any sets A, B, $S \subset \mathcal{V}$ where S separates A and B, $\mathbf{x}_A$ and $\mathbf{x}_B$ are conditionally independent given the value of $\mathbf{x}_S$.

The model is a Gaussian graphical model or GMRF when the random vector $\mathbf{x}_\mathcal{V}$ is jointly Gaussian. The probability density function 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 the *information matrix* or *precision matrix* and $\mathbf{h}$ is the *potential vector*. The parameters J and $\mathbf{h}$ are related

This research was supported in part by AFOSR under Grant FA9550-12-1-0287.

[1] Message passing is also called message broadcasting if messages are passed without being modified.



(a) LBP on the subgraph excluding the feedback nodes

(b) Solving a small inference problem among the feedback nodes

(c) Feedback nodes send feedback messages back to their neighbors

(d) Another round of LBP among the non-feedback nodes gives the final results

Fig. 1: Illustration for the FMP algorithm. Shaded nodes (4, 6, and 15) are selected feedback nodes.

to the mean $\boldsymbol{\mu}$ and covariance matrix P by $\boldsymbol{\mu} = J^{-1}\mathbf{h}$ and $P = J^{-1}$. The structure of the underlying graph is constructed such that there is an edge between i and j if and only if $J_{ij} \neq 0$. It can be shown that x_i and x_j are conditionally independent given all the other variables if and only if $J_{ij} = 0$. The inference problem in Gaussian graphical models refers to computing (exactly or approximately) the means μ_i and variances Σ_{ii} for all $i \in \mathcal{V}$ given J and $\mathbf{h}$.

B. Loopy Belief Propagation

LBP is a distributed inference algorithm for loopy graphs. Without loss of generality, we refer to both BP and LBP as LBP throughout the paper, as the protocols are the same. In LBP, each outgoing message from a node is updated using only incoming messages and local parameters. For tree-structured GMRFs, LBP runs in linear time (with respect to the number of nodes) and is exact. However, convergence and correctness are not guaranteed for LBP when the graphs have cycles. The equations for message update can be found in [15], [13].

C. Walk-sum Analysis

Walk-sum analysis is a framework in which the means and variances are interpreted as the sum of "walks" [15]. Walk-sum analysis allows us to characterize the errors of various algorithms in terms of missed walks and understand the algorithms better. As shown in [15], for loopy graphs, LBP collects all of the required walks for the computation of the means, but only some of the walks required for computing the variances.

We call $R = I - J$ the *edge-weight matrix*, where I is the identity matrix and J is normalized to have unit diagonal. Due to the page limit, we omit the precise definition of walk-sums here and only summarize some useful conclusions. Readers interested in walk-sum analysis can refer to [15] for details.

1) A GMRF is walk-summable if $\rho(\bar{R}) < 1$, where $\bar{R}$ is the matrix whose elements are the absolute values of the corresponding elements in R.
2) For a walk-summable Gaussian graphical model, LBP converges and gives the correct means.
3) In walk-summable models, the variance computed by LBP for each node is the sum of all backtracking walks[2], which is a subset of all self-return walks needed for computing the correct variance.

D. Feedback Message Passing

A *feedback vertex set* (FVS) is defined as a set of vertices whose removal (with the removal of the incident edges) results in an cycle-free graph [16]. A pseudo-FVS is a subset of an FVS that breaks not all but most crucial cycles.

The FMP algorithm described in [17] works as follows. Before running FMP, an FVS or a pseudo-FVS is selected by a greedy algorithm to break the most crucial cycles.[3] The selected nodes are called *feedback nodes*. In the first stage of FMP, LBP is employed in the subgraph excluding the feedback nodes to compute the initial inference results and "feedback gains" (Figure 1a). After convergence, the feedback nodes collect the feedback gains from their neighbors and solve a small inference problem involving only the feedback nodes (Figure 1b). In the third stage, the feedback nodes send feedback messages to their neighbors to modify local parameters (Figure 1c). Finally, another round of LBP among the non-feedback nodes gives the final results (Figure 1d). After convergence, FMP gives the exact means for all nodes as well as exact variances for the feedback nodes. The variance estimate for a non-feedback node i equals the sum of all backtracking walks plus all self-return walks that visit the feedback nodes. The message update equations of FMP and more theoretical results can be found in [13].

III. Recursive FMP

In many practical networks, each node has limited local memory and communication bandwidth. In addition, individual nodes often do not know the diameter of the whole networks. In recursive FMP, each node i has a local list of feedback nodes, denoted by L_i. The set L_i is initially empty and has a maximum size of K_i. As will be explained later, the number of messages node i sends out is proportional to the current size of L_i. Another parameter d, called the *effective diameter,* indicates the default estimate of the network diameter. There are three stages in recursive FMP: In the first

[2] A backtracking walk of a node is a self-return walk that can be reduced consecutively to a single node. Each reduction is to replace a subwalk of the form $\{i, j, i\}$ by the single node $\{i\}$. For example, a self-return walk of the form 12321 is backtracking, but a walk of the form 1231 is not.

[3] In [13], the algorithm is called approximate FMP when a pseudo-FVS is used. In this paper, we refer to both exact and approximate as FMP for conciseness.

stage, feedback nodes are elected using a distributed algorithm similar to the "leader election" algorithm [18]. The current feedback nodes are also called *inactive nodes* since they do not pass any messages; the non-feedback nodes are referred to as *active nodes*. The inactive nodes later "wake up" to become active nodes again. We denote the set of feedback nodes by $\mathcal{F}$ and the set of active nodes by $\mathcal{A}$. The subgraph induced by all active nodes is called the *active subgraph* (the active subgraph before any feedback node wakes up is called the *initial active subgraph*). In the second stage, LBP is run on the initial active subgraph while the feedback nodes remain inactive. In the third stage, each of the feedback nodes wakes up to become an active node when some local conditions are satisfied and broadcasts correction messages. In practice, the three stages are integrated together and have no clear separation; however, for clarity, we present the protocols in three separate stages.

A. First Stage: Election of Feedback Nodes

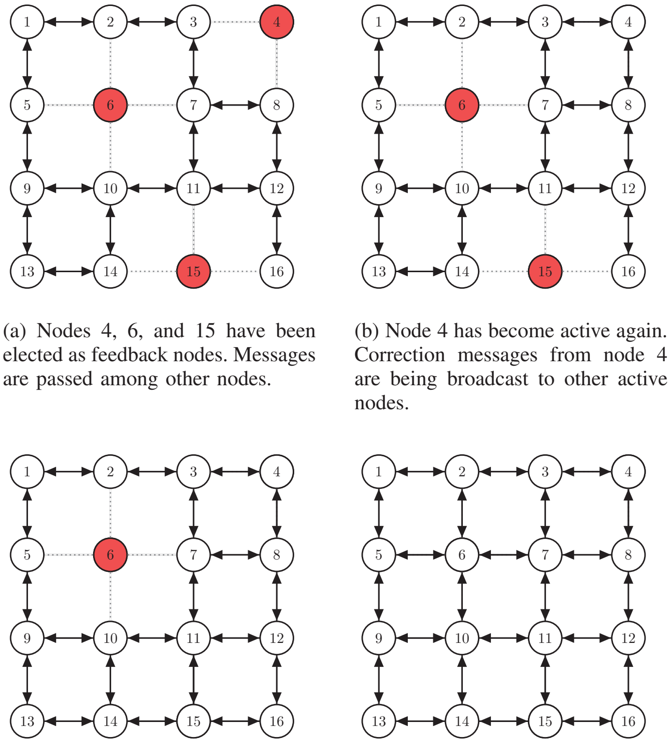
The election algorithm favors nodes that have high priority scores and break many cycles. The priority score for node i is denoted by $s(i)$, and the definition is motivated by the criterion in [13] to enhance convergence. At the beginning, all nodes have status U (undecided), i.e., $S_i = \text{U}$ for all $i \in \mathcal{V}$. The status of a node will change to either A (active) or F (feedback) under different conditions. Each node stores the largest score (denoted by MaxScore(i)) it has seen and the corresponding node index (denoted by MaxIndex(i)). These values are passed to its undecided neighbors (neighbors with status U) as messages. Throughout this paper, we use $\mathcal{N}_{\text{U}}(i)$ to denote the set of i's neighbors with status U, $\mathcal{N}_{\text{F}}(i)$ the set of i's neighbors with status F, and $\mathcal{N}_{\text{A}}(i)$ those with status A. The distributed algorithm for electing the feedback nodes is as follows.

```
For each node $i$ with status U,
  1) Repeat for $K_i$ times
     a) Compute $s(i) = \sum_{j \in \mathcal{N}_{\text{U}}(i)} |J_{ij}| / \sqrt{J_{ii} J_{jj}}$. Set
        MaxScore$(i) \leftarrow s(i)$ and MaxIndex$(i) \leftarrow i$.
     b) Repeat for $d$ iterations
         i) If $|\mathcal{N}_{\text{U}}(i)| \leq 1$, then $S_i \leftarrow$ A.
        ii) Send [MaxIndex$(i)$ MaxScore$(i)$] to all
            neighbors with status U.
       iii) if MaxScore$(i)$ is less than the maximum received
            score, then replace MaxScore$(i)$ with
            the maximal received score and MaxIndex$(i)$
            with the corresponding node index.
     c) If MaxIndex$(i) = i$, then $S_i \leftarrow$ F.
  2) If $S_i =$ U, then $S_i \leftarrow$ A.
```

B. Second Stage: Initial Estimation

In the second stage, initial estimates of the means and variances are computed for the initial active subgraph by passing messages only among the active nodes. Each active node i stores the following values: P_i, the current estimate of the variance; μ_i, the current estimate of the mean; L_i, the set of feedback nodes stored at node i; and the current feedback gains g_i^k for all $k \in L_i$. The messages from node i to node j include $\Delta J_{i\to j}$, $\Delta h_{i\to j}$, and $\Delta g_{i\to j}^k$ for all $k \in L_i$.

At the beginning, each feedback node k constructs an extra potential vector $\mathbf{h}^k$ with $(\mathbf{h}^k)_i = J_{ik}$ for all $i \in \mathcal{N}_{\text{A}}(k)$. Note that the only non-zero entries of $\mathbf{h}^k$ correspond to the active neighbors of k, and thus can be constructed locally. We also initialize L_i as $L_i = \mathcal{N}_F(i)$ for every active node i. We refer to the message $\Delta g_{i\to j}^k$ as a BP message *for* feedback node k. The priority score $s(k)$ is passed with the BP message for k to be used in the third stage. The message-update protocol is described as follows and illustrated by Figure 2a.



(a) Nodes 4, 6, and 15 have been elected as feedback nodes. Messages are passed among other nodes.

(b) Node 4 has become active again. Correction messages from node 4 are being broadcast to other active nodes.

(c) Node 15 has become active again. Correction messages from node 15 are being broadcast to other active nodes.

(d) All nodes are active nodes now. Final correction messages from node 6 are being broadcast.

Fig. 2: Second stage (a) and third stage (b-d) of recursive FMP. Shaded nodes represent elected feedback nodes. Solid lines with arrows denote the edges where messages are being passed.

```
At each iteration $t$, for each active node $i$ :
  1) send messages to all $j \in \mathcal{N}_A(i)$,
     $\Delta J_{i\to j}^{(t)} = -J_{ji} \hat{J}_{i\backslash j}^{(t-1)} J_{ij}$ and $\Delta h_{i\to j}^{(t)} =
     -J_{ji} \hat{J}_{i\backslash j}^{(t-1)} \hat{h}_{i\backslash j}^{(t-1)}$, where
        $\hat{J}_{i\backslash j}^{(t-1)} = J_{ii} + \sum_{l \in \mathcal{N}_A(i)\backslash\{j\}} \Delta J_{l\to i}^{(t-1)}$
        $\hat{h}_{i\backslash j}^{(t-1)} = h_i + \sum_{l \in \mathcal{N}_A(i)\backslash\{j\}} \Delta h_{l\to i}^{(t-1)}$.
  2) When receiving a BP message for feedback node $k$,
     update the local list by $L_i \leftarrow L_i \cup \{k\}$. Only keep the
     nodes with the top $K_i$ priority scores if $|L_i| > K_i$.
  3) For all $k \in L_i$ and all $j \in \mathcal{N}_A(i)$, send messages
     $\left(\Delta g_{i\to j}^k\right)^{(t)} = -J_{ji} \left(\hat{J}_{i\backslash j}^{(t-1)}\right)^{-1} \left(\hat{g}_{i\backslash j}^k\right)^{(t-1)}$, where
        $\left(\hat{g}_{i\backslash j}^k\right)^{(t-1)} = (\mathbf{h}^k)_i + \sum_{l \in \mathcal{N}_A(i)\backslash\{j\}} \left(\Delta g_{l\to i}^k\right)^{(t-1)}$
```

and $\left(\Delta g^k_{l\to i}\right)^{(t-1)} = 0$ if $k \notin L_l$.

4) Update the local values by

$$\begin{aligned} P_i &\leftarrow \left(J_{ii} + \sum_{j\in\mathcal{N}_A(i)} \Delta J^{(t)}_{j\to i}\right)^{-1} \\ \mu_i &\leftarrow P_i\left(h_i + \sum_{j\in\mathcal{N}_A(i)} \Delta h^{(t)}_{j\to i}\right) \\ g^k_i &\leftarrow P_i\left((\mathbf{h}^k)_i + \sum_{j\in\mathcal{N}_A(i)} \left(\Delta g^k_{j\to i}\right)^{(t)}\right) \quad \text{for } k \in L_i. \end{aligned}$$

C. Third Stage: Recursive Correction

In this stage, each feedback node k becomes active again when some local conditions are satisfied and broadcasts correction messages (which are called correction messages *about* feedback node k). The third stage ends when all nodes are active and all correction messages have been broadcast. The message protocol is as follows (see Figures 2b–2d for illustration).

For a feedback node k:

Let $L_k \leftarrow \cup_{j\in\mathcal{N}_A(k)} L_j$. If the following two conditions are satisfied: 1) k itself has the lowest priority score in L_k; and 2) BP messages for k have converged at k's active neighbors (or d iterations have passed since the last change of L_k), then

1) $S_k \leftarrow$ A, $L_k \leftarrow L_k \backslash \{k\}$.
2) Compute the current estimates of the variance, the mean, and the feedback gains at k by

$$\begin{aligned} P_k &\leftarrow \left(J_{kk} - \sum_{j\in\mathcal{N}_A(k)} J_{kj} g^k_j\right)^{-1} \\ \mu_k &\leftarrow P_k\left(h_k - \sum_{j\in\mathcal{N}_A(k)} J_{kj}\mu_j\right) \\ g^p_k &\leftarrow P_k\left((\mathbf{h}^p)_k - \sum_{j\in\mathcal{N}_A(k)} J_{kj} g^p_j\right), \text{ for } p \in L_k, \end{aligned}$$

where $g^p_j = 0$ if $p \notin L_j$.

3) Send correction messages including P_k, μ_k, and g^p_k, $\forall p \in L_k$ to active neighbors.

For an active node i:

When correction messages about k are received, then

1) $L_i \leftarrow (L_i \cup L_k) \backslash \{k\}$. Only keep the nodes with the top K_i priority scores if $|L_i| > K_i$.
2) Update the local values by

$$\begin{aligned} P_i &\leftarrow P_i + (g^k_i)^2 P_k \\ \mu_i &\leftarrow \mu_i - g^k_i \mu_k \\ g^p_i &\leftarrow g^p_i - g^k_i g^p_k, \text{ for } p \in L_i, \end{aligned}$$

where $g^p_i = 0$ and $g^k_i = 0$ if $p, k \notin L_i$ before the update of L_i.

3) Pass the same correction messages to other active neighbors

D. Discussions on Local Lists of Feedback Nodes

In recursive FMP, it is entirely possible (and very likely) that different nodes have different lists of feedback nodes. At the beginning of the second stage, the list stored at an active node includes only neighboring feedback nodes. The lists are then exchanged only within distance d, and thus a node may not know feedback nodes located far away. In addition, if the total number of feedback nodes exceeds K_i for some i, then the lists may be different even if d is as large as the network diameter. Moreover, if the initial active subgraph is disconnected, then the lists may be different even if d and all of the K_i's are sufficiently large, because some message pathways are broken. However, in this case, as will be stated in Proposition 2, the inference results are exact.

IV. Theoretical Results

In this section, we provide some theoretical results on the convergence and accuracy of recursive FMP. Due to the page limit, the proofs are omitted here and provided in the journal version of the paper in preparation.

Proposition 1. *If the effective distance d is equal to or greater than the diameter of the graph and $\forall i$, $K_i \geq K$ for some K, then the election algorithm in the first stage gives the same set of feedback nodes as Algorithm 2 in [13] with parameter K.*

Proposition 1 shows that the distributed election algorithm is consistent with the non-distributed version when the effective distance is large enough.

Proposition 2. *If 1) $\mathcal{F}$, the set of elected feedback nodes, is an FVS; 2) the effective distance d is at least the diameter of the initial active subgraph; and 3) $|\mathcal{F}| \leq K_i$, $\forall i$, then recursive FMP gives exact means and variances for all nodes in time $\mathcal{O}(|\mathcal{F}|N)$ with a total computational cost of $\mathcal{O}(|\mathcal{F}|^2 N)$, where N is the total number of nodes. The memory cost is $\mathcal{O}(|\mathcal{F}|)$ per node, and the communication cost is $\mathcal{O}(|\mathcal{F}|)$ per edge per iteration.*

Propositions 2 guarantees the correctness of recursive FMP when d and K_i are sufficiently large and gives the cost of the algorithm.

Proposition 3. *If $\mathcal{F}$ is a pseudo-FVS, then recursive FMP converges when the initial active subgraph is walk-summable. If $|\mathcal{F}| \leq K_i$ for all i, then after convergence, recursive FMP gives the exact means for all nodes and exact variances for all nodes in $\mathcal{F}$. The variance estimate for a non-feedback node equals the sum of all backtracking walks in the initial active subgraph plus all walks that visit at least one feedback node.*

Proposition 3 shows that under mild conditions recursive FMP has the same improvements on convergence and accuracy compared with FMP while being a purely distributed algorithm.

Proposition 4. *When node i receives correction messages about k, the correction term added to the P_i equals the sum of all self-return walks that stay within the current active subgraph and visit k at least once.*

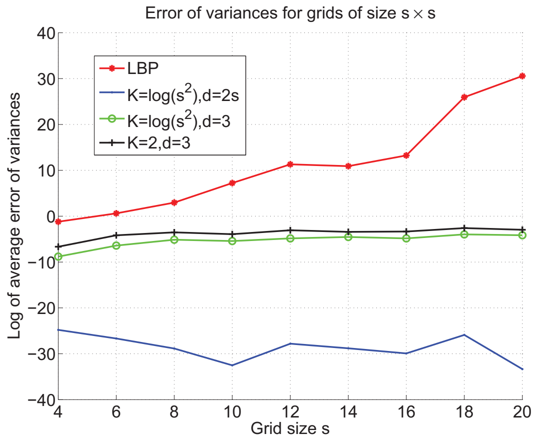


Fig. 3: Recursive FMP with different parameters performed on grids of various sizes

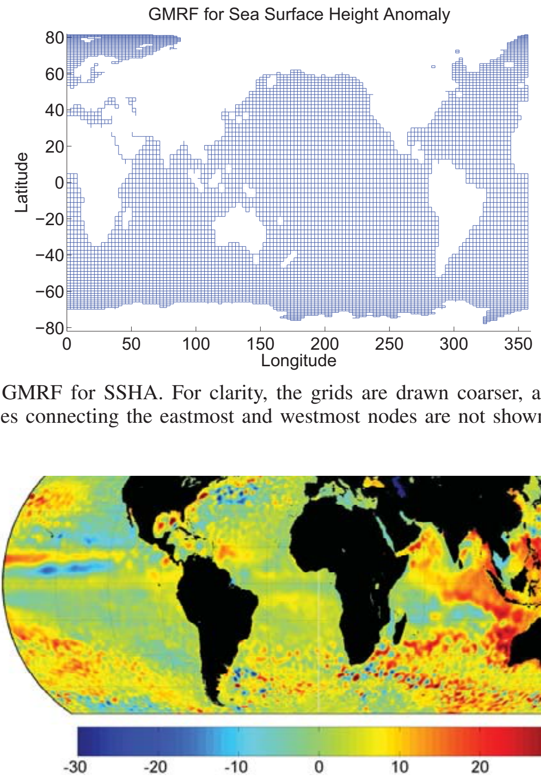
Proposition 4 provides a precise characterization of the correction terms using a walk-sum interpretation.

V. Experimental Results

In this section, we demonstrate the performance of recursive FMP using both simulated models on grids and a large-scale GMRF with about a million variables.

We simulate GMRFs defined on grids of size 4×4 to 20×20. For each size, we generate 50 models with random parameters. We solve the inference problems using LBP, and recursive FMP with different parameters of d and K_i. For each algorithm and each grid size, the average errors of variances are computed. As shown in Figure 3, LBP has large estimation errors of the variances while recursive FMP gives significant improvements. The inference results are more accurate when K_i and d are larger, showing the trade-off between memory capacity and accuracy.

We also use sea surface height anomaly (SSHA) data, which is measured relative to seasonal, space-variant mean-sea level (the dataset is publicly available at http://podaac.jpl.nasa.gov/dataset/). The raw data is preprocessed to have measurements at 915×1080 different locations with latitudes between $\pm 82^\circ$ and a full 360° of longitude. We construct a grid of 988,200 nodes and connect the eastmost and westmost nodes at the same latitudes (since they are geographical neighbors). We then remove the nodes that have invalid measurements (most of which correspond to land areas) and obtain the final graph structure shown in Figure 4a. With this underlying structure, we build an GMRF using the thin-membrane model [19]. The final estimates by recursive FMP with $d = 200$ and $K_i = 15$ for all i are plotted in Figure 4b.



(a) GMRF for SSHA. For clarity, the grids are drawn coarser, and the edges connecting the eastmost and westmost nodes are not shown.

(b) Estimated SSHA with $d = 200$ and $K_i = 15$ for all i.

Fig. 4: Estimating SSHA using recursive FMP

References

[1] C. Wunsch and P. Heimbach, “Practical global oceanic state estimation,” *Physica D: Nonlinear Phenomena*, vol. 230, no. 1-2, pp. 197–208, 2007.

[2] L. Yang, X. Liu, C. Jursa, M. Holliman, A. Rader, H. Karimi, and I. Bahar, “iGNM: a database of protein functional motions based on Gaussian Network Model,” *Bioinformatics*, vol. 21, no. 13, p. 2978, 2005.

[3] 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.

[4] K. Murphy, Y. Weiss, and M. Jordan, “Loopy belief propagation for approximate inference: an empirical study,” in *Proc. of Uncertainty in Artificial Intell. (UAI)*, 1999, pp. 467–475.

[5] D. Bickson, O. Shental, and D. Dolev, “Distributed Kalman filter via Gaussian belief propagation,” in *46th Annu. Allerton Conf. Commun., Control, and Computing*, 2008, pp. 628–635.

[6] M. Wainwright and M. Jordan, “Graphical models, exponential families, and variational inference,” *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[7] J. Yedidia, W. Freeman, and Y. Weiss, “Constructing free-energy approximations and generalized belief propagation algorithms,” *IEEE Trans. Inform. Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.

[8] M. Wainwright, T. Jaakkola, and A. Willsky, “Tree-based reparameterization framework for analysis of sum-product and related algorithms,” *IEEE Trans. Inform. Theory*, vol. 49, no. 5, pp. 1120–1146, 2003.

[9] D. Dolev, D. Bickson, and J. Johnson, “Fixing convergence of Gaussian belief propagation,” in *Proc. IEEE Int. Symp. Inform. Theory*, 2009, pp. 1674–1678.

[10] Y. El-Kurdi, D. Giannacopoulos, and W. Gross, “Relaxed Gaussian Belief Propagation,” in *Proc. IEEE Int. Symp. Inform. Theory*, 2012, pp. 2002–2006.

[11] C. C. Moallemi and B. Van Roy, “Convergence of min-sum message passing for quadratic optimization,” *IEEE Trans. Inform. Theory*, vol. 55, no. 5, pp. 2413–2423, 2009.

[12] N. Ruozzi and S. Tatikonda, “Message-passing algorithms for quadratic minimization,” *arXiv preprint arXiv:1212.0171*, 2012.

[13] Y. Liu, V. Chandrasekaran, A. Anandkumar, and A. Willsky, “Feedback message passing for inference in Gaussian graphical models,” *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4135–4150, 2012.

[14] S. Lauritzen, *Graphical Models*. New York: Oxford University Press, 1996.

[15] D. Malioutov, J. Johnson, and A. Willsky, “Walk-sums and belief propagation in Gaussian graphical models,” *J. Machine Learning Research*, vol. 7, pp. 2031–2064, 2006.

[16] V. Vazirani, *Approximation Algorithms*. New York: Springer, 2004.

[17] Y. Liu, “Feedback message passing for inference in Gaussian graphical models,” Master’s thesis, Massachusetts Institute of Technology, 2010.

[18] N. A. Lynch, *Distributed Algorithms*. San Francisco: Morgan Kaufmann, 1996.

[19] D. Malioutov, J. Johnson, M. Choi, and A. Willsky, “Low-rank variance approximation in GMRF models: Single and multiscale approaches,” *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4621–4634, 2008.