Sampling from Gaussian graphical models using subgraph perturbations

Citation: Liu, Ying, Oliver Kosut, and Alan S. Willsky. "Sampling from Gaussian Graphical Models Using Subgraph Perturbations." 2013 IEEE International Symposium on Information Theory (July 2013).

As Published: http://dx.doi.org/10.1109/ISIT.2013.6620676

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: http://hdl.handle.net/1721.1/91051

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Sampling from Gaussian Graphical Models Using Subgraph Perturbations

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

Oliver Kosut
School of Electrical, Computer and Energy Engineering
Arizona State University
Email: okosut@asu.edu

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

***Abstract*—The problem of efficiently drawing samples from a Gaussian graphical model or Gaussian Markov random field is studied. We introduce the subgraph perturbation sampling algorithm, which makes use of any pre-existing tractable inference algorithm for a subgraph by perturbing this algorithm so as to yield asymptotically exact samples for the intended distribution. The subgraph can have any structure for which efficient inference algorithms exist: for example, tree-structured, low tree-width, or having a small feedback vertex set. The experimental results demonstrate that this subgraph perturbation algorithm efficiently yields accurate samples for many graph topologies.**

I. Introduction

An important family of Markov random fields (MRFs) is the family of Gaussian Markov random fields or Gaussian graphical models (GGMs). GGMs are often used to directly parametrize probabilistic networks and used as approximate models to circumvent the computational complexity inherent in many discrete models. This paper is devoted to developing efficient algorithms for drawing samples from a GGM.

Samples from a GGM can be drawn exactly by conducting a Cholesky decomposition of the covariance or inverse covariance matrix and then performing a linear transformation on *i.i.d.* Gaussian samples. However, the cubic complexity of this direct method precludes its use on large-scale models. GGMs with particular topologies (for example, tree-structured models) have well-known efficient sampling algorithms, but possess limited modeling power [1].

A popular sampling algorithm for general loopy graphs is Gibbs sampling, an Markov Chain Monte Carlo (MCMC) algorithm in which variables are sequentially drawn conditioned on the most recent sample of all other variables (or of the variables in the Markov blankets in the MRF setting) [2]. However, the Gibbs sampler can have slow mixing rate [3]. In [4], both exact methods and iterative sampling methods using blocking or divide-and-conquer strategies are studied. In [5], a local perturbation method is proposed from GGMs where a Cholesky decomposition is available. In [6], a blocked Gibbs sampler is proposed, where the induced subgraph within each block is cycle-free.

In this paper, we introduce the *subgraph perturbation algorithm*, which leverages any pre-existing efficient inference algorithm on a subgraph (that contains all the nodes but only a subset of the edges) and randomly perturbs the parameters so as to generate exact samples asymptotically. This algorithm can be considered as a general framework of converting a subgraph-based linear solver in numerical analysis to a sampling algorithm for GGMs. Using a tree-structured subgraph, our algorithm is a randomized extension of the embedded-tree algorithm [7], which has been shown to have excellent convergence properties when the spanning trees are selected adaptively [7], [8]. As our algorithm produces an exact sample from the target distribution asymptotically, the number of iterations required depends on the convergence rate. We provide theoretical characterization of the convergence rate, both exactly and via tractable bounds. We run experiments using GGMs of various structures and different sizes to demonstrate that the algorithm converges quickly on a wide variety of graphs. We also show the performance of the algorithm on a large-scale GGM built for estimating sea surface temperature.

II. Background

A. Gaussian Graphical Models

An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is used in an MRF to model the conditional independencies among a set of random variables [9], where $\mathcal{V}$ denotes the set of vertices (nodes) and $\mathcal{E}$ the set of edges. Each node $s \in \mathcal{V}$ corresponds to a random variable x_s. The random vector $\mathbf{x}_\mathcal{V}$ is Markov with respect to the graph if for any subset A, B, $S \subset \mathcal{V}$ where S separates A and B in the graph, we have that $\mathbf{x}_A$ and $\mathbf{x}_B$ are conditionally independent given the value of $\mathbf{x}_S$.

When the random vector $\mathbf{x}_\mathcal{V}$ is jointly Gaussian, the model is a GGM. 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 mean $\boldsymbol{\mu}$ and covariance matrix Σ are related to J and $\mathbf{h}$ by $\boldsymbol{\mu} = J^{-1}\mathbf{h}$ and $\Sigma = J^{-1}$. The structure of the underlying graph can be identified using the sparsity pattern of J, i.e., $J_{ij} \neq 0$ if and only if there is an edge

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

Algorithm 1 Sampling by Subgraph Correction

Input: J, $\mathbf{h}$, and $\mathcal{T}$
Output: samples with the asymptotic distribution $p(\mathbf{x}) \propto \exp\{-\frac{1}{2}\mathbf{x}^T J\mathbf{x} + \mathbf{h}^T\mathbf{x}\}$

1) Form $J_{\mathcal{T}}$ and K using (1)–(2).
2) Draw an initial sample $\mathbf{x}^{(0)}$ from a Gaussian distribution with potential vector $\mathbf{h}$ and information matrix being the diagonal of J.
3) In each iteration:
   a) Compute $\mathbf{e}^{(t+1)}$ using (3).
   b) Draw a new sample $\mathbf{x}^{(t+1)}$ from a Gaussian distribution with information matrix $J_{\mathcal{T}}$ and potential vector $\mathbf{h} + Kx^{(t)} + \mathbf{e}^{(t+1)}$.

between i and j. It can be shown that x_i and x_j are conditionally independent given all the other variables if and only if $J_{ij} = 0$. Hence, the conditional independencies can be read immediately from the sparsity pattern of the information matrix as well as the sparsity pattern the graph. The sampling problem refers to generating samples from the distribution $p(\mathbf{x})$ given J and $\mathbf{h}$.

B. Some Existing Sampling Algorithms

Cholesky Decomposition: The Cholesky decomposition is used to obtain a lower triangular matrix L such that $J = LL^T$. Let $\mathbf{z}$ be an n-dimensional random vector whose entries are drawn *i.i.d.* from the standard Gaussian distribution. An exact sample $\mathbf{x}$ can be obtained by computing $\mathbf{x} = (L^T)^{-1}\left(\mathbf{z} + L^{-1}\mathbf{h}\right)$. The total computational complexity of generating p exact samples is $\mathcal{O}(n^3 + pn^2)$. When a Cholesky decompostion of the model is given, the local perturbation algorithm in [5] can also be used to generate samples.

Forward Sampling for Tree-Structured Models: For a tree-structured GGM, an exact sample can be generated in linear time (with respect to the number of nodes) by first computing the variances and means for all nodes and covariances for the edges using belief propagation and then sampling the variables one by one in a root-to-leaf order [9].

Gibbs Sampling: Gibbs sampling is a commonly used MCMC method. At each iteration, following a predetermined order, the entries of a new sample are drawn sequentially conditioned on the most recent values of their neighbors. The Gibbs sampler is always convergent when $J \succ 0$; however, the convergence rate[1] can be very slow for many GGMs, including many tree-structured models [10]. More details on Gibbs sampling can be found in [11]. The nodes can be grouped into several blocks. In particular, when the each block induces a tree-structured subgraph, Gibbs sampling becomes the algorithm in [6]

[1]The convergence rate for the covariance matrix $\tau_\Sigma = -\ln\left(\liminf_{t\to\infty}\left(||\Sigma^{(t+1)} - \Sigma||/||\Sigma^{(t)} - \Sigma||\right)\right)$, where $\Sigma^{(t)}$ is the sample covariance at iteration t and the matrix norm is the Frobenius norm; the convergence rate for the mean, $\tau_{\boldsymbol{\mu}}$, is similarly defined. Here, the convergence rate refers to the smaller of the two.

III. Sampling GGMs by Subgraph Perturbation

The subgraph perturbation algorithm builds on the ideas used in graphical splitting preconditioning algorithms for solving large linear systems [12]. In our context, determining the means of the graphical model corresponds to solving the equation $J\boldsymbol{\mu} = \mathbf{h}$. A matrix splitting for solving this equation would be writing $J = J_{\mathcal{T}} - K$, where $J_{\mathcal{T}}$ corresponds to a tractable spanning subgraph and K corresponds to a graph consisting of the removed edges (Figure 1 shows an example where the subgraph $\mathcal{T}$ is chosen to be a spanning tree). Then the relationship $J_{\mathcal{T}}\boldsymbol{\mu} = K\boldsymbol{\mu} + \mathbf{h}$ is used as the basis for an iterative algorithm. The high level idea of our sampling algorithm is to add random perturbations at each iteration so that we convert linear solvers that converge to a fixed-point solution to sampling algorithms that give exact samples asymptotically.

The subgraph perturbation algorithm runs as follows: We use the same graph decomposition as the linear solver but add a proper diagonal matrix to K (and the same diagonal matrix to $J_{\mathcal{T}}$) to make it positive semi-definite. Let $\mathcal{E}_{\mathcal{T}}$ denote the set of edges in the subgraph $\mathcal{T}$. The matrix K can be constructed as the sum of rank-one matrices:

$$K = \sum_{(i,j)\in\mathcal{E}\backslash\mathcal{E}_{\mathcal{T}}} \left[K^{(i,j)}\right]_{n\times n}, \tag{1}$$

where each $K^{(i,j)} = \begin{bmatrix} |J_{ij}| & -J_{ij} \\ -J_{ij} & |J_{ij}| \end{bmatrix}$ is a two-by-two matrix corresponding to edge (i,j), and $\left[K^{(i,j)}\right]_{n\times n}$ is an n-by-n matrix zero-padded from $K^{(i,j)}$, i.e., the principal submatrix corresponding to rows (and columns) i and j of $\left[K^{(i,j)}\right]_{n\times n}$ equals $K^{(i,j)}$ while other entries are zeros. The matrix $J_{\mathcal{T}}$ is obtained by

$$J_{\mathcal{T}} = J + K. \tag{2}$$

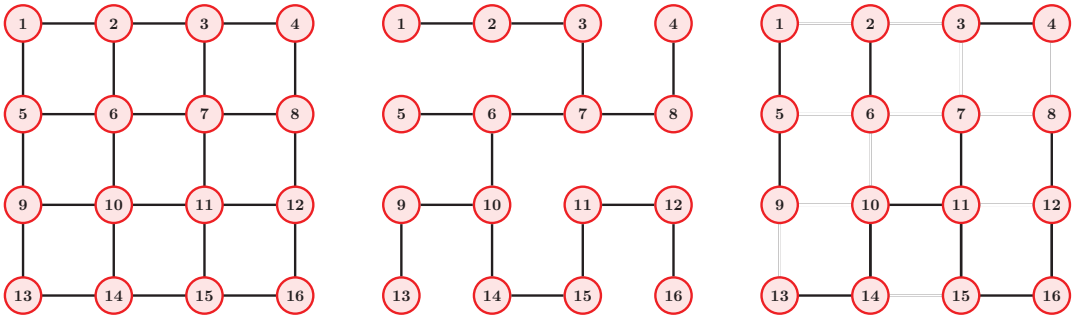
It is easy to see that, as required, K is positive semi-definite and $J_{\mathcal{T}}$ is positive definite (since J is positive definite for a valid model).

At iteration t+1, rather than solving $J_{\mathcal{T}}\boldsymbol{\mu}^{(t+1)} = K\boldsymbol{\mu}^{(t)} + \mathbf{h}$, instead we draw a sample from a Gaussian with information matrix $J_{\mathcal{T}}$ and potential vector $K\mathbf{x}^{(t)} + \mathbf{h} + \mathbf{e}^{(t+1)}$. The vector $\mathbf{e}^{(t+1)}$ is Gaussian with zero mean and covariance matrix K. It represents a perturbation to the potential vector that compensates for the discrepancy between the spanning subgraph and the full graph. Moreover, $\mathbf{e}^{(t+1)}$ can be computed efficiently and locally from our construction of K: For each $(i,j) \in \mathcal{E}\backslash\mathcal{E}_{\mathcal{T}}$, let the two-dimensional vector $\mathbf{e}^{(i,j)}$ be sampled from a zero-mean Gaussian distribution with covariance matrix $K^{(i,j)}$. We can obtain $\mathbf{e}^{(t+1)}$ by computing

$$\mathbf{e}^{(t+1)} = \sum_{(i,j)\in\mathcal{E}\backslash\mathcal{E}_{\mathcal{T}}} \left[\mathbf{e}^{(i,j)}\right]_n, \tag{3}$$

where $\left[\mathbf{e}^{(i,j)}\right]_n$ is an n-dimensional vector zero-padded from $\mathbf{e}^{(i,j)}$, i.e., the i-th and j-th entries of $\left[\mathbf{e}^{(i,j)}\right]_n$ take the two entries of $\mathbf{e}^{(i,j)}$and all other entries of $\left[\mathbf{e}^{(i,j)}\right]_n$ are zero.

The subgraph perturbation algorithm is summarized in Algorithm 1. The computational complexity of one iteration



(a) graph structure for J (b) graph structure for J_T (c) graph structure for K

Figure 1: the grid shown in (a) can be decomposed into a spanning tree (b) and a graph consisting of the removed edges (c)

is $C_{\mathcal{T}} + \mathcal{O}(|\mathcal{E}_K|)$, where $C_{\mathcal{T}}$ is the complexity of drawing a sample from the tractable subgraph $\mathcal{T}$ (which is the same as solving an inference problem on $\mathcal{T}$) and $|\mathcal{E}_K| = |\mathcal{E} - \mathcal{E}_{\mathcal{T}}|$ is the number of edges missing from $J_{\mathcal{T}}$.

IV. Theoretical Results

In this section, we prove some theoretical results on the convergence of the algorithm. We give both the exact convergence rate and its tractable bounds.

Proposition 1. *For a GGM with information matrix $J \succ 0$ and potential vector $\mathbf{h}$, the sample distribution generated by Algorithm 1 is guaranteed to converge to the exact distribution and the convergence rate is $-\ln \rho(J_{\mathcal{T}}^{-1}K)$, where $\rho(A)$ denotes the spectral radius of matrix A.*

Proof: The sample distribution at each iteration is a Gaussian because the initial sample $\mathbf{x}^{(0)}$ is drawn from a Gaussian distribution and the distribution at each iteration is Gaussian conditioned on the previous value. Hence, we can parametrize the sample distribution at iteration t by the mean $\boldsymbol{\mu}^{(t)}$ and the covariance matrix $\Sigma^{(t)}$.

From Step 3(c) in Algorithm (1), we have

$$\begin{aligned}\boldsymbol{\mu}^{(t+1)} &= \mathbb{E}\mathbf{x}^{(t+1)} = \mathbb{E}\left[J_{\mathcal{T}}^{-1}(\mathbf{h} + \mathbf{e}^{(t+1)} + K\mathbf{x}^{(t)})\right] \\ &= J_{\mathcal{T}}^{-1}(\mathbf{h} + K\boldsymbol{\mu}^{(t)}),\end{aligned}$$

where the equalities $\mathbb{E}\mathbf{e}^{(t+1)} = 0$ and $\mathbb{E}\mathbf{x}^{(t)} = \boldsymbol{\mu}^{(t)}$ are used. From Lemma 1 (see below), when $K \succeq 0$ and $J \succ 0$, we have $\rho(J_{\mathcal{T}}^{-1}K) < 1$. Hence, the mean $\boldsymbol{\mu}^{(t+1)}$ converges to the unique fixed-point $\hat{\boldsymbol{\mu}}$ satisfying

$$\hat{\boldsymbol{\mu}} = J_{\mathcal{T}}^{-1}\left(\mathbf{h} + K\hat{\boldsymbol{\mu}}\right).$$

Thus we have $\hat{\boldsymbol{\mu}} = (J_{\mathcal{T}} - K)^{-1}\mathbf{h} = J^{-1}\mathbf{h}$, so $\boldsymbol{\mu}^{(t)}$ converges to the exact mean $J^{-1}\mathbf{h}$ with convergence rate $-\ln \rho(J_{\mathcal{T}}^{-1}K)$.

Now we consider the convergence of the covariance matrix. From Step 3(c) in Algorithm (1), we have

$$\begin{aligned}\Sigma^{(t+1)} &= \mathrm{Cov}\left\{\mathbf{x}^{(t+1)}\right\} \\ &= J_{\mathcal{T}}^{-1} + \mathrm{Cov}\left\{J_{\mathcal{T}}^{-1}(\mathbf{h} + \mathbf{e}^{(t+1)} + K\mathbf{x}^{(t)})\right\} \\ &= J_{\mathcal{T}}^{-1} + J_{\mathcal{T}}^{-1}\left(K + K\Sigma^{(t)}K\right)J_{\mathcal{T}}^{-1} \\ &= (J_{\mathcal{T}}^{-1}K)\Sigma^{(t)}(J_{\mathcal{T}}^{-1}K)^T + J_{\mathcal{T}}^{-1}(J_{\mathcal{T}} + K)J_{\mathcal{T}}^{-1}.\end{aligned} \tag{4}$$

This equation can be rewritten in vector form as

$$\begin{aligned}\mathrm{vec}(\Sigma^{(t+1)}) &= \left[(J_{\mathcal{T}}^{-1}K) \otimes (J_{\mathcal{T}}^{-1}K)\right]\mathrm{vec}(\Sigma^{(t)}) \\ &\quad + \mathrm{vec}(J_{\mathcal{T}}^{-1}(J_{\mathcal{T}} + K)J_{\mathcal{T}}^{-1}),\end{aligned} \tag{5}$$

where $\mathrm{vec}(\cdot)$ is a column vector obtained by stacking all the columns in its argument and $A \otimes B$ is the Kronecker product of matrices A and B, i.e.,

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix},$$

where A is an m-by-n matrix $[a_{ij}]_{m\times n}$.

From [13], $\rho\left((J_{\mathcal{T}}^{-1}K) \otimes (J_{\mathcal{T}}^{-1}K)\right)$, the spectral radius of the matrix $(J_{\mathcal{T}}^{-1}K) \otimes (J_{\mathcal{T}}^{-1}K)$, is $\rho^2(J_{\mathcal{T}}^{-1}K)$. By Lemma 1 we have $\rho^2(J_{\mathcal{T}}^{-1}K) < 1$. Hence the iterative equation (5) is guaranteed to converge to a unique fixed-point, denoted by $\mathrm{vec}(\hat{\Sigma})$, and thus the equation (4) converges to a unique fixed point $\hat{\Sigma}$.

By Lemma 2 (see below), $\hat{\Sigma} = (J_{\mathcal{T}} - K)^{-1} = J^{-1}$ is the fixed-point solution. Therefore, the sample covariance $\Sigma^{(t)}$ converges to the target covariance J^{-1} with convergence rate $-\ln \rho^2(J_{\mathcal{T}}^{-1}K)$. This completes the proof. ∎

Lemma 1. *Let J, $J_{\mathcal{T}}$ and K be real symmetric matrices satisfying $J = J_{\mathcal{T}} - K$. If $J \succ 0$ and $K \succeq 0$, then $\rho(J_{\mathcal{T}}^{-1}K) < 1$.*

Proof: By Theorem 7.7.3 in [14] we have that $\rho(KJ_{\mathcal{T}}^{-1}) < 1$ when $J_{\mathcal{T}} - K = J \succ 0$. The eigenvalues of $KJ_{\mathcal{T}}^{-1}$ remain the same after the similarity transformation $J_{\mathcal{T}}^{-1}(KJ_{\mathcal{T}}^{-1})J_{\mathcal{T}} = J_{\mathcal{T}}^{-1}K$. Hence $\rho(J_{\mathcal{T}}^{-1}K) = \rho(KJ_{\mathcal{T}}^{-1}) < 1$. ∎

Lemma 2. *Let A and B be square matrices. If A is invertible and $A + B$ is symmetric and invertible, then $\Sigma = (A + B)^{-1}$ is a solution of the equation $A\Sigma A^T = B\Sigma B^T + A^T - B$.*

Proof: It is equivalent to showing

$$A(A + B)^{-1}A^T = B(A + B)^{-1}B^T + A^T - B.$$

To do so, consider

$$\begin{aligned}\text{LHS} &= ((A+B)-B)(A+B)^{-1}A^T \\ &= A^T - B(A+B)^{-1}A^T \\ &= A^T - B(A+B)^{-1}\left((A^T+B^T)-B^T\right) \\ &= A^T - B(A+B)^{-1}(A+B)^T + B(A+B)^{-1}B^T \\ &\overset{(a)}{=} A^T - B + B(A+B)^{-1}B^T = \text{RHS},\end{aligned}$$

where (a) is due to the assumption that $A+B$ is symmetric. ■

Proposition 2. *Consider symmetric matrices J, $J_{\mathcal{T}}$, and K that satisfy $J = J_{\mathcal{T}} - K$. If $J \succ 0$ and $K \succeq 0$, then* $\frac{\lambda_{\max}(K)}{\lambda_{\max}(K)+\lambda_{\max}(J)} \leq \rho(J_{\mathcal{T}}^{-1}K) \leq \frac{\lambda_{\max}(K)}{\lambda_{\max}(K)+\lambda_{\min}(J)} < 1$.

Proof: Use Theorem 2.2 in [15] and let $\mu = 1$. ■

Proposition 2 is adopted from [7] and gives bounds on the convergence rate.

We can further bound $\lambda_{\max}(K)$, the largest eigenvalue of K, using a simple function of its entries. We define the weight of node i in a GGM with information matrix K as $w(i) = \sum_j |K_{ij}|$ and the weight of the model as $w(K) = \max_i w(i)$. Corollary 1 follows immediately.

Corollary 1. *In the same setting as in Proposition 2, we have* $\rho(J_{\mathcal{T}}^{-1}K) \leq \frac{w(K)}{w(K)+\lambda_{\min}(J)}$.

Proof: As K is symmetric positive semi-definite, we have $\lambda_{\max}(K) = \rho(K)$. By Corollary 8.1.18 in [14], we have that $\rho(K) \leq \rho(\overline{K})$, where $\overline{K}$ takes the entry-wise absolute values of K. By Corollary 8.1.22 in [14], $\lambda_{\max}(K) = \rho(K) \leq \rho(\overline{K}) \leq w(K)$. The corollary thus follows. ■

V. On Selecting Tractable Subgraphs

Our algorithm does not restrict the subgraph to be tree-structured. Any subgraph with fast inference methods can be used, such as subgraphs with low tree-width [16], or subgraphs with small feedback vertex sets (FVS[2]) [17]. The computational complexity of one iteration is the complexity of generating one sample from the tractable subgraph, plus a term proportional to the number of missing edges from the subgraph. Although we have presented the algorithm using a single, constant splitting for clarity, using different trees or other tractable structures at different iterations can be very beneficial, similarly as it is useful when calculating the means in the inference case [8]. This sequence of subgraphs can be selected *a priori* or on the fly. By Proposition 1, to speed convergence, $J_{\mathcal{T}}$ should be selected to minimize $\rho(J_{\mathcal{T}}^{-1}K)$. In this section, we give brief references to the selection algorithms for different families of tractable subgraphs.

Tree-Structured Subgraphs: The idea of using a maximum-weighted spanning tree (MST) has been discussed in the support graph preconditioner literature [12] as well as in the studies of the embedded tree algorithm for inference [8], where multiple embedded trees are selected adaptively.

[2]An FVS is a set of nodes whose removal results in a cycle-free graph.

Subgraphs with Low Tree-width: Graphical models with low tree-width have efficient inference and sampling algorithms and have been widely studied. We can compute a low tree-width approximation $J_{\mathcal{T}}$ to J using algorithms such as those in [18], [19], [16].

Subgraphs with Small FVS: When a subgraph with a small FVS is used, there is a trade-off in choosing the FVS size (a larger FVS means more computation per iteration but faster convergence). The key step of obtaining the subgraph is the selection of the FVS. We can first use the algorithm in [17] to select a pseudo-FVS of the entire graph (the pseudo-FVS does not break all the cycles in the entire graph, but will be an FVS of the subgraph to be constructed). Then we compute the MST among the other nodes. The tractable subgraph is constructed by combining the nodes in the FVS (with all their incident edges) and the MST of the remaining graph.

Spectrally Sparsified Subraphs: Many common GGMs such as thin-membrane or thin-plate models have diagonally dominant information matrices. Some recent studies show that the graph Laplacian of a dense graph can be well-approximated by the graph Laplacian of graphs with nearly-linear number of edges [20]. These spectrally sparsified graphs have efficient inference and sampling algorithms and can be used as tractable subgraphs.

VI. Experimental Results

In this section, we present some experimental results using the subgraph perturbation algorithm using various subgraphs. We compare the performances on both simulated models and on models for sea surface temperature data.

For simulated models, the model parameters J and $\mathbf{h}$ are randomly generated as follows: the entries of the potential vector $\mathbf{h}$ are generated $i.i.d.$ from a uniform distribution $U[-1,1]$; the non-zero entries of J are also generated *i.i.d.* from $U[-1,1]$ with a multiple of the identity matrix added to ensure $J \succ 0$. We compute the numbers of iterations needed to achieve an approximating error of $\epsilon = 10^{-5}$, i.e., the minimum t such that $\left\|\Sigma^{(t)} - \Sigma\right\| \leq \epsilon$. We run the subgraph perturbation algorithm on l-by-l grids with l ranging from 3 to 30. For each grid, two different subgraphs are used: one is a tree-structured subgraph, the other is a graph with an FVS of size $\lceil \log l^2 \rceil$. For each size, we repeat the algorithm for 100 sets of random model parameters and the results shown are averaged over the 100 runs. Note that since the sizes of the simulated models are moderate, we are able to compute and compare with the exact solutions. Figure 2 shows that both kinds of subgraphs give better convergence than the Gibbs sampler while the subgraphs with small FVSs perform the best consistently.

We also run the algorithm on a large-scale GGM built to estimate the sea surface temperature (the dataset is publicly available at http://podaac.jpl.nasa.gov/dataset/). The raw data is preprocessed to have raw measurements at 720×1440 different locations. We construct a grid of 1,036,800 nodes (we also connect the eastmost and westmost nodes at the same latitudes as they are geographical neighbors). We then remove the nodes that have invalid measurements corresponding to land areas. We build an GGM with this underlying structure using the thin-plate

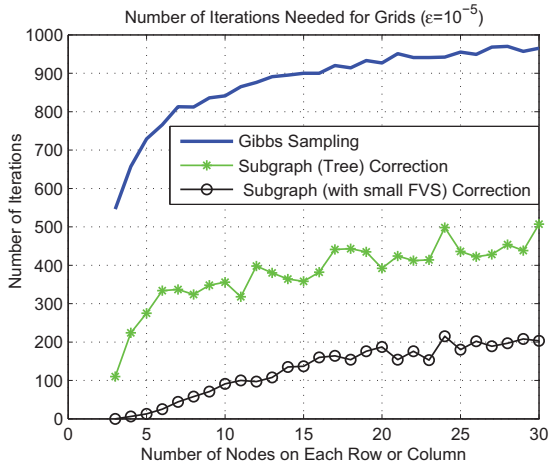
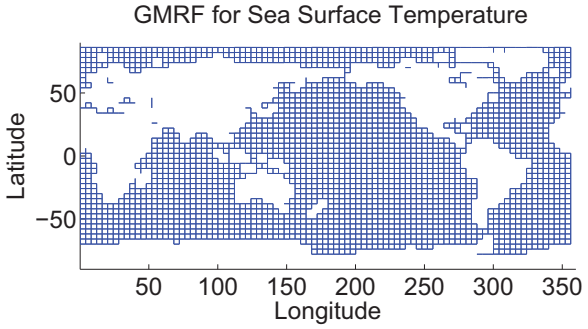
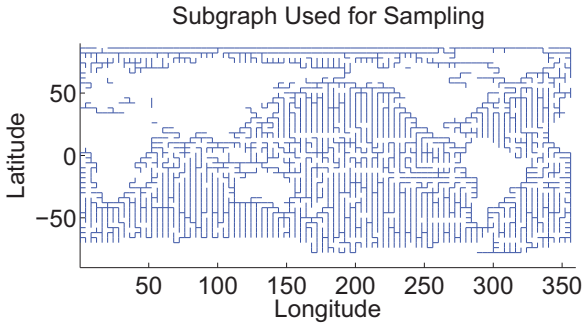


Figure 2: the performance on grids of size 3-by-3 to 30-by-30. The tractable subgraphs used include tree-structured graphs and graphs with small FVSs.

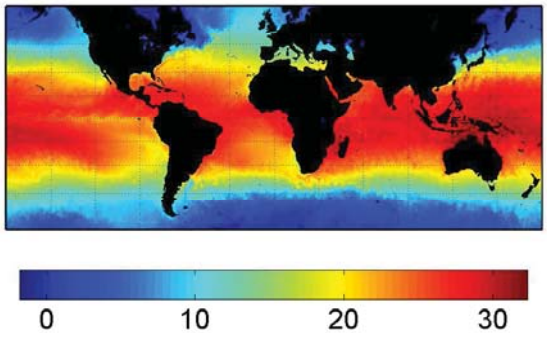
model [21]. The resulting GGM is shown in Figure 3a and the subgraph used for the sampling algorithm is shown in Figure 3b (for clarity, we plot a much coarser version and hide the edges connecting the eastmost and westmost nodes). A sampled estimate after 200 iterations from the posterior distribution is shown in Figure 3c.



(a) the entire GGM for sea surface temperature



(b) the spanning tree used as a tractable subgraph



(c) sea surface temperature in degrees (Celsius)

Figure 3: GGM for estimating sea surface temperature

REFERENCES

[1] D. Batra, A. Gallagher, D. Parikh, and T. Chen, "Beyond trees: MRF inference via outer-planar decomposition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 2496–2503.

[2] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[3] C. K. Carter and R. Kohn, "On Gibbs sampling for state space models," *Biometrika*, vol. 81, no. 3, pp. 541–553, 1994.

[4] H. Rue, "Fast sampling of Gaussian Markov random fields," *J. Roy. Statistical Soc.: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 325–338, 2001.

[5] G. Papandreou and A. Yuille, "Gaussian sampling by local perturbations," in *Proc. NIPS*, 2010.

[6] F. Hamze and N. de Freitas, "From fields to trees," in *Proc. 20th Conf. Uncertainty in Artificial Intell. (UAI)*, 2004, pp. 243–250.

[7] E. Sudderth, M. Wainwright, and A. Willsky, "Embedded trees: estimation of Gaussian processes on graphs with cycles," *IEEE Trans. Signal Process.*, vol. 52, no. 11, pp. 3136–3150, 2004.

[8] V. Chandrasekaran, J. Johnson, and A. Willsky, "Estimation in Gaussian graphical models using tractable subgraphs: A walk-sum analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1916–1930, 2008.

[9] M. Jordan, "Graphical models," *Statistical Sci.*, pp. 140–155, 2004.

[10] A. Thomas, A. Gutin, V. Abkevich, and A. Bansal, "Multilocus linkage analysis by blocked Gibbs sampling," *Stat. and Computing*, vol. 10, no. 3, pp. 259–269, 2000.

[11] C. Robert and G. Casella, *Monte Carlo statistical methods*. Springer Verlag, 2004.

[12] M. Bern, J. Gilbert, B. Hendrickson, N. Nguyen, and S. Toledo, "Support-graph preconditioners," *SIAM J. Matrix Anal. and Applicat.*, vol. 27, no. 4, pp. 930–951, 2006.

[13] A. Laub, *Matrix Alysis for Scientists and Engineers*. Society of Industrial Mathematics, 2005.

[14] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge University Press, 1990.

[15] O. Axelsson, "Bounds of eigenvalues of preconditioned matrices," *SIAM J. Matrix Anal. and Applicat.*, vol. 13, no. 3, pp. 847–862, 1992.

[16] D. Karger and N. Srebro, "Learning Markov networks: Maximum bounded tree-width graphs," in *Proc. 12th Annu. ACM-SIAM Symp. on Discrete Algorithms*, 2001, pp. 392–401.

[17] 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.

[18] N. Srebro, "Maximum likelihood bounded tree-width Markov networks," in *Proc. 17th Conf. Uncertainty in Artificial Intell. (UAI)*, 2001, pp. 504–511.

[19] D. Shahaf, A. Chechetka, and C. Guestrin, "Learning thin junction trees via graph cuts," in *Conf. Artificial Intell. and Stat.*, 2009.

[20] D. Spielman and S. Teng, "Spectral sparsification of graphs," *SIAM J. Computing*, vol. 40, p. 981, 2011.

[21] 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.