Dynamic Bayesian Gaussian Graphical Models for Inferring Evolving Network Structure

Slav Kirov
Dept. of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213
skirov@andrew.cmu.edu

Micol Marchetti-Bowick
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
micolmb@cs.cmu.edu

Willie Neiswanger
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
willie@cs.cmu.edu

Abstract

We propose a method for learning the structure of evolving Gaussian graphical models via Bayesian inference. Gaussian graphical models (GGMs) are often used to model the structure of a network, where they have found applications in biology (gene networks), finance (relationships among stocks), computer vision (object pose) and a number of related fields. We develop a dynamic Bayesian model that is able to learn the structure of an evolving network composed of a sequence of GGMs with more accuracy than its static counterpart. We evaluate our method on synthetic data and then apply it to two real-world datasets.

1 Introduction

Learning the structure of a network is a problem of interest in a variety of fields. For a biologist, uncovering the structure of a gene network that charactarizes functional associations between genes can provide insight into how genes interact during the regulation of a biological process. Similarly, a financial analyst may be interested in understanding how the prices of different stocks relate to one another. In both of these settings, learning a dynamic network that evolves over time can provide even more valuable information about how these relationships vary under different conditions. Inferring the structure of a dynamic gene network—such as one that evolves over the course of a biological lineage—can provide insight into how gene associations change as a cell differentiates. Learning a dynamic network of stocks as the market evolves over time can provide valuable information about which stocks co-vary and how these relationships are influenced by external factors.

A variety of approaches have been used in the past to model static networks with hidden structure. One of the most popular is the Gaussian Graphical Model (GGM), which assumes that a set of observations $X_1, ..., X_n$ where $X_i \in \mathbb{R}^p$ are drawn IID from a multivariate normal distribution with mean 0 and unknown covariance $\Sigma \in \mathbb{R}^{p \times p}$. The conditional indepenence relationships between variables are encoded in the precision matrix, $\Omega = \Sigma^{-1}$. If the ij^{th} element of Ω is 0 then X_i and X_j are conditionally independent given all of the other variables. We can encode the (unweighted) conditional independence structure of this model in a graph, G, with p nodes and an edge between node X_i and node X_j if and only if $\Omega_{ij} = 0$.

Here, we propose a generative model for a time-evolving network. The model, which we call a Dynamic Bayesian Gaussian Graphical Model (DB-GGM), is constructed using a formulation of the GGM as a Bayesian network model that has been developed in the past for a static network. By introducing a dependence between the underlying graph at each time point, we convert the static model into a dynamic Bayesian network. We then derive two Bayesian inference procedures for our model, and evaluate their performance on a synthetic dataset. Finally, we apply our method to two real datasets: a gene network that encodes relationships between the activity levels of genes, and a network of stocks that encodes relationships between stock prices.

1.1 Previous Work

Recently, many techniques have emerged for estimating the structure of the graph G that encodes the conditional independence structure of a set of variables in a static GGM. Two of the most popular approaches are neighborhood selection [5] and glasso [2], both of which involve running a variant of Lasso regression, which uses an L1 penalty to encourage sparsity in the resulting graph. Although these discriminative methods are efficient for very large networks (over 1,000 nodes) and have good statistical guarantees, they lack the expressiveness and flexibility of generative models.

A few approaches have been developed for formulating a static GGM as a generative model and performing Bayesian inference to estimate the structure of the underlying graph. In many cases, though only efficient for relatively small networks (under 100 nodes), generative models have advantages over discriminative techniques because they can more easily be extended to incorporate both prior knowledge and additional observations from new data sources [8, 4, 3]. A class of these approaches involves using G-Wishart prior distributions for precision matrices of GGMs. Wang and Li, 2012 [10], develop an MCMC sampling procedure for inference in such a model, while a related birth-death MCMC inference approach is employed by Mohammadi and Wit [6].

Finally, some methods have been proposed for estimating the structure of a network that evolves over time. Parikh et al. [7] estimate networks that evolve over a biological lineage using an evolving GGM and learn the network structures by extending neighborhood selection to penalize differences between adjacent networks. In KELLER, Song et al. [9] estimate time-varying interactions of genes by using an MRF model at each time step and using logistic regression with a smoothing kernel to recover networks that change gradually over time.

2 Methods

In this section, we will discuss our model and inference procedures.

2.1 Model

Our generative model is a dynamic Bayesian network (DBN) that incorporates a Gaussian graphical model (GGM) at each time step. The model consists of hidden variables $\{A_t\}_{t=1}^T$ and $\{K_t\}_{t=1}^T$ and observed variables $\{X_t\}_{t=1}^T$. $A_t = \{A_{ij}^t : i < j\}$ is a p-by-p adjacency matrix that represents the graph structure at time t. K_t is a precision matrix for a multivariate Normal distribution with covariance $\Sigma_t = K_t^{-1}$. $K_t \mid A_t$ is drawn from a G-Wishart distribution, which is a conjugate prior for the precision matrix of a multivariate Normal for a given graph structure A_t. X_t is a vector $\in \mathbb{R}^p$ that is drawn from a multivariate Normal distribution with mean 0 and covariance Σ_t. At time t, each A_{ij}^t has a value of either 1 (edge present) or 0 (edge not present). The distribution of A_{ij}^t depends on the value of the same edge at the previous time point. Each edge has a "flip on" and "flip off" probability, given by p_{ij}^0 and p_{ij}^1, respectively. These probabilities are defined as follows:

$$p_{ij}^0 = P(A_{ij}^t = 1 \mid A_{ij}^{t-1} = 0) \qquad 1 - p_{ij}^0 = P(A_{ij}^t = 0 \mid A_{ij}^{t-1} = 0)$$
$$p_{ij}^1 = P(A_{ij}^t = 0 \mid A_{ij}^{t-1} = 1) \qquad 1 - p_{ij}^1 = P(A_{ij}^t = 1 \mid A_{ij}^{t-1} = 1)$$

The specific parameterization of our model is given below and the associated graphical model is provided in Figure 1. Note that the values of the prior parameters α_0, β_0, α_1, β_1, b, and D are set manually.

$$\begin{aligned}
p_{ij}^0 &\sim \text{Beta}(\alpha_0, \beta_0) \\
p_{ij}^1 &\sim \text{Beta}(\alpha_1, \beta_1) \\
A_{ij}^t \mid (A_{ij}^{t-1} = 0) &\sim \text{Bernoulli}(p_{ij}^0) \\
A_{ij}^t \mid (A_{ij}^{t-1} = 1) &\sim \text{Bernoulli}(p_{ij}^1) \\
K_t \mid A_t &\sim \text{G-Wishart}_{A_t}(b, D) \\
X_t \mid K_t &\sim \text{Normal}(0, K_t^{-1})
\end{aligned}$$

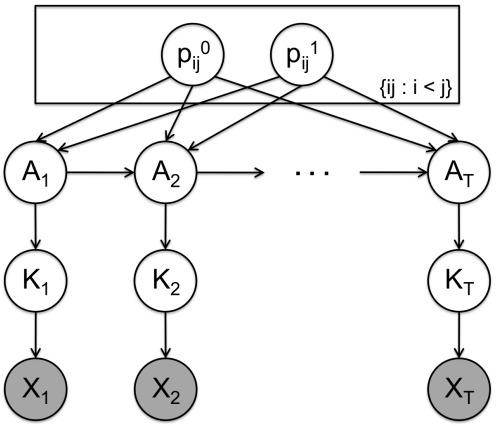

Figure 1: Illustration of the graphical model for our dynamic Bayesian network. Note that each edge A_{ij}^t only depends on one Bernoulli parameter, p_{ij}^a, given that its predecessor $A_{ij}^{t-1} = a$.

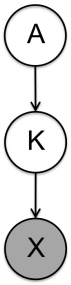

Figure 2: Illustration of the graphical model for the static counterpart to our Bayesian network.

2.2 Inference

A number of previous works have focused on developing inference methods for a static Bayesian GGM, such as the one shown in Figure 2, which is the direct static analog to our dynamic model.

Most recently, Mohammadi and Wit [6], proposed a technique called Birth-Death MCMC (BD-MCMC) for jointly sampling from the posterior over the graph structure A and the precision matrix K (Algorithm 1). Briefly, BD-MCMC involves constructing a Markov chain whose stationary distribution is $P(A, K \mid X)$ by formulating it as a birth-death process, where each state corresponds to a graph with a specific number of edges, E, and its associated precision matrix. The transition probabilities are encoded as birth rates and death rates, and it is the specification of these rates which determines the stationary distribution. Inference is performed by traversing the space of possible assignments to the hidden variables by executing birth moves (adding an edge) and death moves (removing an edge) and resampling the precision matrix after each move. Sampling the precision matrix from a G-Wishart distribution is done by relying on properties of the Choleski decomposition of the precision matrix [6]. BD-MCMC runs efficiently on graphs with up to 100 nodes.

Here, we build on prior work to derive an inference method for the dynamic Bayesian GGM described in Section 2.1. We experiment with two different inference procedures: Sequential Monte Carlo (SMC) and collapsed Gibbs sampling, a type of MCMC.

2.2.1 Sequential Monte Carlo

SMC is a type of inference that is well suited for dynamic models, as it performs Bayesian inference in an online manner. Here, it is particularly useful because it allows us to build upon a previously established inference technique, BD-MCMC, for approximating the partial posterior distribution at each time step.

SMC works in the following way. At time $t-1$, we assume we have N particles (samples of $A_{1:t-1}$ and $K_{1:t-1}$) that are approximately sampled from the posterior distribution $P(A_{1:t-1}, K_{1:t-1} \mid X_{1:t-1})$. We then update these particles at time t so that we are left with N particles approximately sampled from $P(A_{1:t}, K_{1:t} \mid X_{1:t})$. In this way, after we've passed through every time step, we will be left with N samples from the full posterior distribution, $P(A_{1:T}, K_{1:T} \mid X_{1:T})$. The SMC algorithm that we derived is outlined in Algorithm 2.

For the proposal distribution in step 6, we use:

$$Q_t = P(A_t, K_t \mid A_{1:t-1}^{(i)}, K_{1:t-1}^{(i)}, X_{1:t}) = P(A_t, K_t \mid K_{t-1}^{(i)}, A_{t-1}^{(i)}, X_t^{(i)})$$

Algorithm 1 Birth-Death MCMC

inputs: Data X and prior on adjacency matrices $P(A)$
Initialize A as an upper triangular matrix with all ones
Sample precision matrix K
Define birth rates $\beta_{(i,j)}(K) = 1$ for all $(i,j) \in \bar{E} := \{(i,j) : i \leq j, A_{(i,j)} = 0\}$
repeat
  Calculate the total birth rate $\beta(K) = |\bar{E}|$
  Calculate the death rates,

$$\delta_{(i,j)}(K) = \frac{p(X \mid K_{-(i,j)}, A_{-(i,j)})p(A_{-(i,j)})}{p(X \mid K, A)p(A)}, \quad \text{for all } (i,j) \in \bar{E}$$

  Calculate the total death rate, $\delta(K) = \sum_{(i,j)\in E} \delta_{(i,j)}(K)$
  Calculate the waiting time, $\lambda(K) = \beta(K) + \delta(K)$
  Simulate the type of jump (birth or death), with respective probabilities:

$$p(\text{birth of } (i,j)) = \frac{\beta_{(i,j)}(K)}{\lambda(K)}, \quad \text{for all } (i,j) \in \bar{E}$$

$$p(\text{death of } (i,j)) = \frac{\delta_{(i,j)}(K)}{\lambda(K)}, \quad \text{for all } (i,j) \in E$$

  if birth of (i,j) **then**
    Sample a new precision matrix K given $A_{+(i,j)}$
  else if death of (i,j) **then**
    Sample a new precision matrix K given $A_{-(i,j)}$
  end if
until convergence
output: Samples from the full posterior distribution, $P(A, K \mid X)$

Algorithm 2 SMC for Dynamic Bayesian GGM

inputs:
(i) A sequence of observations $X_t : t = 1, ..., T$
(ii) Parameters b and D for the G-Wishart prior over precision matrices
(iii) Parameters α^0, β^0, α^1, β^1 for the Beta priors over the Bernoulli parameters p_{ij}^0 and p_{ij}^1
(iv) Number of particles, N
Run BD-MCMC using data X_1 and uniform graph prior $P(A_1)$ as inputs; outputs samples from the partial posterior distribution $P(A_1, K_1 \mid X_1)$
Generate N particles sampled from $P(A_1, K_1 \mid X_1)$
for $t = 2, ..., T$ **do**
  for $i = 1, ..., N$ **do**
    Sample A_t and K_t from the proposal distribution,

$$Q_t(A_t, K_t \mid A_{1:t-1}^{(i)}, K_{1:t-1}^{(i)}, X_{1:t})$$

    using BD-MCMC with data X_t and graph prior $P(A_t) = P(A_t|A_{t-1})$ as inputs.
    Extend particle i with A_t and K_t
    Evaluate the importance weight,

$$w_t^{(i)} \propto \frac{P(X_t \mid A_t^{(i)}, K_t^{(i)})P(A_t^{(i)}, K_t^{(i)} \mid A_{1:t-1}^{(i)}, K_{1:t-1}^{(i)}, X_{1:t-1})}{Q_t(A_t, K_t \mid A_{1:t-1}^{(i)}, K_{1:t-1}^{(i)}, X_{1:t})}$$

  end for
  Resample the particles according to a categorical distribution made up of the normalized weights $w_1, ..., w_N$
end for
output: N samples from the full posterior distribution, $P(A_{1:T}, K_{1:T} \mid X_{1:T})$

To sample from Q_t, we carry out BD-MCMC with data X_t and choose graph prior $P(A_t) = P(A_t|A_{t-1})$ (given in Section 2.1).

2.2.2 Collapsed Gibbs Sampling

Gibbs sampling is a type of MCMC that uses the conditional probability of a single variable conditioned on all other variables as the sampling distribution for that variable. In collapsed Gibbs sampling, some variables are integrated out so that they don't need to be sampled. Here, we integrate out $K_1, ..., K_T$ and sample all other hidden variables. To do this, we iterate through all edges at each time step and sample new values for A^t_{ij} from

$$\begin{aligned}
P(A^t_{ij} \mid \text{rest}\backslash K_{1:T}) &= P(A^t_{ij} \mid A_{-t}, A^t_{-ij}, X_{1:T}) \\
&= P(A^t_{ij} \mid A^{t-1}_{ij} A^t_{-ij}, A^{t+1}_{ij}, X_t) \\
&= \frac{1}{Z} P(A^t_{ij} \mid A^{t-1}_{ij}) P(A^{t+1}_{ij} \mid A^t_{ij}) P(X_t \mid A_t) \\
&= \frac{1}{Z} P(A^t_{ij} \mid A^{t-1}_{ij}) P(A^{t+1}_{ij} \mid A^t_{ij}) \int_{K_t} P(X_t, K_t \mid A_t) \, \mathrm{d}K_t \\
&= \frac{1}{Z} P(A^t_{ij} \mid A^{t-1}_{ij}) P(A^{t+1}_{ij} \mid A^t_{ij}) \int_{K_t} P(X_t \mid K_t) P(K_t \mid A_t) \, \mathrm{d}K_t \\
&= \frac{1}{Z} P(A^t_{ij} \mid A^{t-1}_{ij}) P(A^{t+1}_{ij} \mid A^t_{ij}) (2\pi)^{(\frac{-np}{2})} \frac{I_{G[A_t]}(b+n, D+S)}{I_{G[A_t]}(b, D)}
\end{aligned}$$

In the last line above, $I_{G[A_t]}$ is the normalization constant for a G-Wishart distribution with the given parameters that are consistent with graph A_t. $I_{G[A_t]}(b, D)$ is the normalization constant for the prior distribution over K_t and $I_{G[A_t]}(b+n, D+S)$ is the normalization constant for the posterior distribution over $K_t \mid X_t$, where n is the number of samples of X_t and $S = X_t X_t^T$ is the sample covariance matrix of X_t. Although I_G cannot be computed exactly, various approximations have been derived. Here we use a Monte Carlo simulation technique presented in [1] to approximate I_G. For completeness, the Gibbs sampling algorithm that we use is outlined in Algorithm 3.

Algorithm 3 Collapsed Gibbs Sampling for Dynamic Bayesian GGM

input:
(i) A sequence of observations $X_t : t = 1, ..., T$
(ii) Parameters b and D for the G-Wishart prior over precision matrices
(iii) Parameters α^0, β^0, α^1, β^1 for the Beta priors over the Bernoulli parameters p^0_{ij} and p^1_{ij}
Initialize values of and $A_{1:T}$
while not converged **do**
  for $t = 1, ..., T$ **do**
    for $\{i, j\} : i < j$ **do**
      Sample a new value for A^t_{ij} from $P(A^t_{ij} \mid \text{rest}\backslash K_{1:T})$
    end for
  end for
end while
output: Samples from the full posterior distribution, $P(A_{1:T} \mid X_{1:T})$

3 Experiments

We carry out experiments on three data sets: synthetic data, stock price data taken over a sequence of weeks, and gene expression data taken from a lineage of stem cells.

3.1 Synthetic Data

We generate a synthetic sequence of 10 networks, and the corresponding observations, in the following way. An initial graph is generated randomly, and the next four graphs are made identical to

the first. To generate the sixth graph from the fifth graph, a randomly selected subset of the "on" edges are turned off and a randomly selected subset of the "off" edges are turned on (approximately one quarter the edges of each type are switched). The remaining graphs are made identical to the sixth. For each graph, we then deterministically choose a consistent precision matrix K_t and then generate the observations $X_t \sim \text{Normal}(0, K_t^{-1})$.

We show results of SMC inference for our synthetically generated time-evolving network in Figure 3. This figure shows the number of incorrectly inferred edges vs. number of particles for a 12-node graph varying over 10 time-steps. We show this curve for our DB-GGM, for a static Bayesian GGM performing inference independently at each time-step ("expanded"), and for a static Bayesian GGM performing inference on the union of the data in all time-steps ("collapsed"). We find that our DB-GGM performs better than the static Bayesian GGM in both cases, and for each setting of the number of particles.

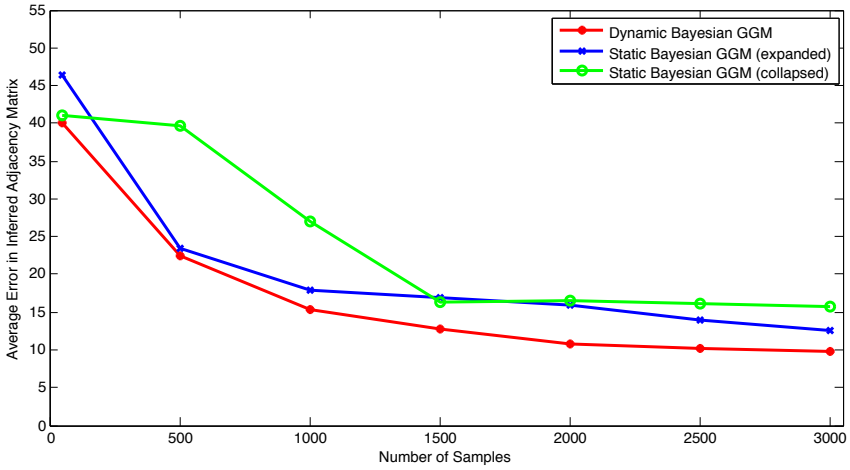

Figure 3: Average number of incorrectly inferred edges (out of 60 total edges) vs. number of particles for the SMC inference algorithm. Each plotted point is an average over five runs.

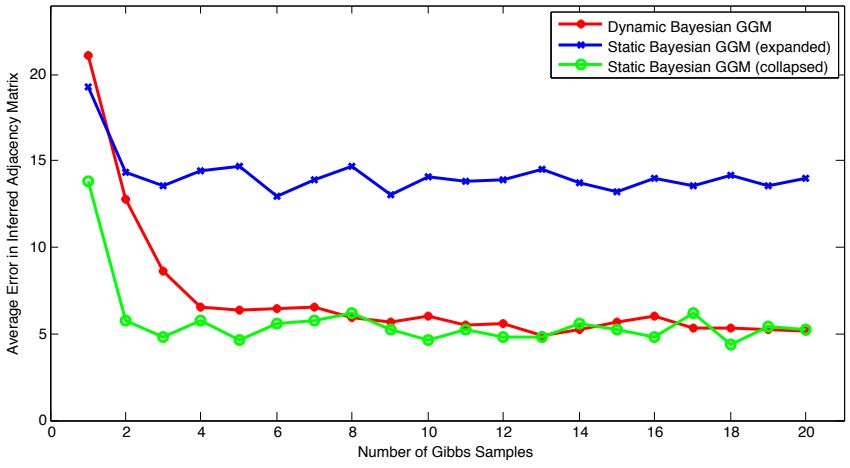

Figure 4: Average number of incorrectly inferred edges (out of 60 total edges) vs. number of Gibbs iterations for the Gibbs inference algorithm. Each plotted point is an average over five runs.

In Figure 4, we show results of our collapsed Gibbs inference algorithm on the synthetic data. This figure plots the number of incorrectly inferred edges vs. number of Gibbs iterations. The same three curves as in the previous figure are also shown here (for the DB-GGM, expanded static GGM, and collapsed static GGM). Although the expanded static GGM performs significantly worse than the other two, we find that both the DB-GGM and collapsed static GGM converge to similar errors. We

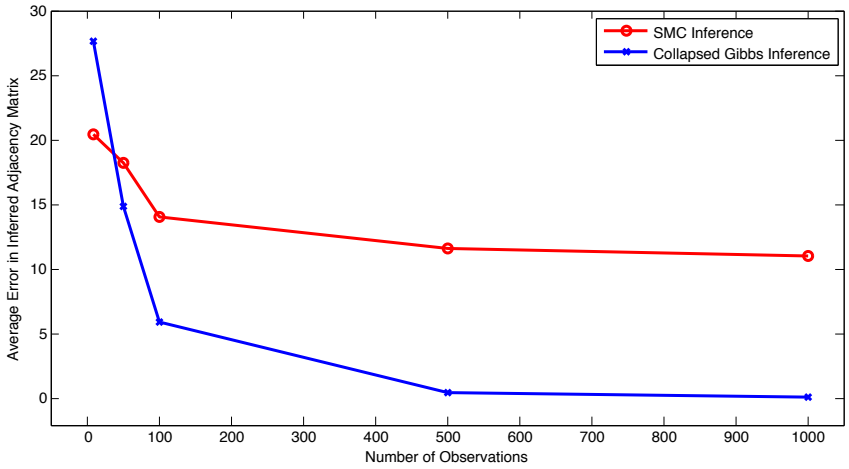

Figure 5: Average number of incorrectly inferred edges (out of 60 total edges) vs. number of observations for the SMC inference algorithm and Gibbs inference algorithm. SMC is run with 1000 particles and Gibbs is run for 5 iterations.

believe this is likely due to the fact that our synthetic network remains consistent over multiple time steps and only changes between time-steps 5 and 6.

We also investigated how the number of incorrectly inferred edges change as the number of obervations per time step is increased for both the SMC and Gibbs inference algorithms. The results of this experiment are shown in Figure 5. Both algorithms have very poor performance when given a very small amount of data (10 observations), although SMC yields more accurate results than Gibbs. When given a large amount of data (at least 100 observations), performance improves significantly for both algorithms but Gibbs has significantly lower error rates than SMC.

3.2 Genetic Networks

In this experiment, we apply our method to a dataset containing expression levels for 25 genes in a lineage of 9 stem cell types. The algorithm infers one network for each cell type, but here we summarize the results by showing a single network consisting of the edges that appeared in at least two different cell types (Figure 6). Although we do not analyze these results any further here, we hope the inferred network reflects underlying regulatory relationships between these 25 genes.

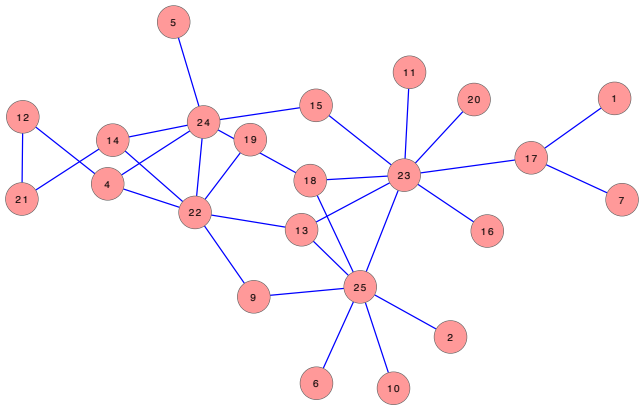

Figure 6: Genetic network constructed by selecting edges that are consistent across the inferred networks for the 9 different cell types.

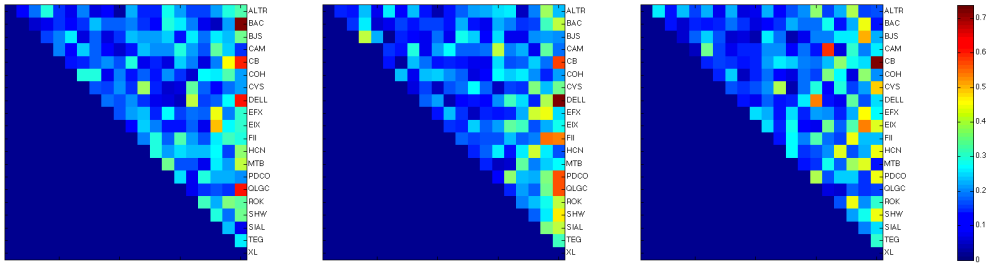

Figure 7: Heatmaps showing posterior probability of edges in stock network in sequence of 3 weeks.

3.3 Stock Market

Finally, we apply our SMC algorithm to a dataset containing the prices of 20 stocks over a period of 10 consecutive weeks. Figure 7 shows the posterior distribution over the network structure for each of the first three weeks. Since we have no ground truth of the underlying networks we estimate, the depth of our analysis is again limited. However, our inferred dynamic network could prove to be useful for financial analysts who wish to gain a better understanding of relationships between specific stock prices and how they vary over time.

4 Conclusion and Future Work

We have derived a time-varying, Bayesian model for inferring the structure of a sequence of Gaussian graphical models. Through experiments on synthetic data we have shown that our dynamic model outperforms static equivalents. In the future, it would also be valuable to compare the performance of our method with that of a discriminative technique for learning the structure of a dynamic network, such as [9]. We also applied our method to two real datasets for which we don't have a ground truth network on which to evaluate performance. To analyze these results further, we could try to understand whether the existence of edges coincides with underlying properties of the individual nodes (genes or stocks) known to specialists in the relevant field, and what reasons might exist for the dynamics to behave as demonstrated by the results.

In our experiments, due to run time limitations, we only applied our method to small networks with 12-25 nodes. In many real-world settings, however, there is a need to infer the structure of very large networks (thousands of nodes) using only a small number of observations. One of the main limitations of our method is that its computational complexity does not allow it to scale up to large networks, and therefore finding an inference technique that can scale up is a central goal for future work.

Our principal motivation for pursuing Gaussian graphical model structure learning in a Bayesian setting is that generative models allow for flexible specification of dependencies between variables, especially of the dynamics of time-varying graphs. In our DBN, we chose a very simple transition model in which each edge flips on or off with some probability p at every time step. We chose a Beta prior that encouraged the value of the Bernoulli parameters to be closer to 1 in order to incorporate our beliefs that the network structure does not change very much from one time point to the next. However, one could very easily construct a similar DBN with a different transition model in order to encode more complex dynamics. This would be especially useful when applying the method in a setting where the dynamics of the evolving network are well studied and there is ample room for prior knowledge to be incorporated into the model design.

References

[1] A. Atay-Kayis and H. Massam. The marginal likelihood for decomposable and non-decomposable graphical gaussian models. *Biometrika*, 92:317–335, 2005.

[2] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[3] P.J. Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[4] B. Jones, C. Carvalho, A. Dobra, C. Hans, C. Carter, and M. West. Experiments in stochastic computation for high-dimensional graphical models. *Statistical Science*, 20(4):388–400, 2005.

[5] N. Meinshausen and P. Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, 34(3):1436–1462, 2006.

[6] A. Mohammadi and E.C. Wit. Efficient birth-death mcmc inference for gaussian graphical models. *arXiv preprint arXiv:1210.5371*, 2012.

[7] A.P. Parikh, W. Wu, R.E. Curtis, and E.P. Xing. Treegl: reverse engineering tree-evolving gene networks underlying developing biological lineages. *Bioinformatics*, 27(13):i196–i204, 2011.

[8] A. Roverato. Hyper inverse wishart distribution for non-decomposable graphs and its application to bayesian inference for gaussian graphical models. *Scandinavian Journal of Statistics*, 29(3):391–411, 2002.

[9] L. Song, M. Kolar, and E.P. Xing. Keller: estimating time-varying interactions between genes. *Bioinformatics*, 25(12):i128–i136, 2009.

[10] H. Wang and S.Z. Li. Efficient gaussian graphical model determination under g-wishart prior distributions. *Electronic Journal of Statistics*, 6:168–198, 2012.