A Particle Gibbs Sampling Approach to Topology Inference in Gene Regulatory Networks

Conference Paper · May 2020

DOI: 10.1109/ICASSP40776.2020.9053525

Yousef El-Laham
Stony Brook University
22 PUBLICATIONS **148** CITATIONS
SEE PROFILE

Mónica F. Bugallo
Stony Brook University
144 PUBLICATIONS **2,174** CITATIONS
SEE PROFILE

A PARTICLE GIBBS SAMPLING APPROACH TO TOPOLOGY INFERENCE IN GENE REGULATORY NETWORKS

Marija Iloska *Yousef El-Laham* *Mónica F. Bugallo*

Department of Electrical and Computer Engineering
Stony Brook University, Stony Brook, NY 11794-2350
{marija.iloska, yousef.ellaham, monica.bugallo}@stonybrook.edu

ABSTRACT

In this paper, we propose a novel Bayesian approach for estimating a gene network's topology using particle Gibbs sampling. The conditional posterior distributions of the unknowns in a state-space model describing the time evolution of gene expressions are derived and employed for exact Bayesian posterior inference. Specifically, the proposed scheme provides the joint posterior distribution of the unknown gene expressions, the adjacency matrix describing the topology of the network, and the coefficient matrix describing the strength of the gene interactions. We validate the proposed method with numerical simulations on synthetic data experiments.

***Index Terms*—** Gibbs sampling, gene networks, network topology, particle filtering, state-space models

1. INTRODUCTION

Gene regulatory networks are networks comprised of interacting genes that work to perform a particular cell function [1]. From a mathematical perspective, these networks are represented by a directed graph, whose nodes correspond to individual genes. The edges between two nodes in the graph indicate an interaction between genes [2]. Given noisy gene expression data, one important task is the estimation of the topology of a gene regulatory network [3, 4], especially in cases where the number of genes in the network is large. This estimation problem has broader impacts to applications beyond those involving gene networks [5, 6].

One popular approach to tackling estimation problems related to gene networks is to utilize a state-space formulation. In gene networks, states in a state-space model (SSM) correspond to time-indexed gene expressions and the observed data correspond to noise corrupted versions of those gene expressions. Some works have been proposed for estimating the true values of the gene expressions even without knowledge of the topology of the network. In the case that the SSM is linear and Gaussian, the Kalman filter [7] provides the optimal solution for the estimation problem. For nonlinear SSMs, numerical strategies based on Kalman filtering extensions or particle filtering (PF) [8–10] are usually preferred. In [11], a multiple particle filtering (MPF) methodology that uses Rao-Blackwellization to integrate out the coefficient matrix describing the topology of the network is proposed. Methods based on MPF that jointly estimate the gene expressions and the network topology have also been proposed [12], however because the weights of the particles in the approach are approximated, no convergence guarantees exist for this method.

In this work, we propose a novel Bayesian estimation method for gene regulatory networks. The proposed scheme encompasses a particle Gibbs sampler, where our contribution is in the derivation of the conditional distributions for the SSM describing the interactions of genes. Since our method is based on Gibbs sampling, it enjoys the same convergence guarantees as demonstrated in [13]. Simulation results on synthetic data show promise in our proposed method as compared another approach based on the least absolute shrinkage and selection operator (LASSO) [2].

This work was carried out thanks to the support of the National Science Foundation (NSF) under Award CCF-1617986. The authors thank the Research Computing and Cyberinfrastructure and the Institute for Advanced Computational Science at Stony Brook University for access to the high-performance SeaWulf computing system sponsored by NSF (OAC-1531492).

2. PROBLEM FORMULATION

Consider a gene regulatory network comprised of d_x genes. Let $\mathbf{x}_t = [x_{1,t}, \dots, x_{d_x,t}]^\top \in \mathbb{R}^{d_x \times 1}$ denote the vector of gene expressions, where $x_{r,t}$ denotes the expression of gene r at time instant t. The evolution of gene expressions can be modeled by the state transition equation

$$\mathbf{x}_t = \mathbf{C}\mathbf{g}(\mathbf{x}_{t-1}) + \mathbf{u}_t, \tag{1}$$

where $\mathbf{C} \in \mathbb{R}^{d_x \times d_x}$ is a coefficient matrix which describes the interaction between genes, $\mathbf{g}(\mathbf{x}_{t-1}) = [g(x_{1,t-1}), \dots, g(x_{d_x,t-1})]^\top \in \mathbb{R}^{d_x \times 1}$ is a vector whose elements are given by a nonlinear transformation $g(\cdot)$ of each gene at the previous time instant, and $\mathbf{u}_t \in \mathbb{R}^{d_x \times 1}$ is the process noise. The matrix $\mathbf{C}$ accounts for the influence imposed on each gene from all the other genes. More specifically, if $c_{j,k}$ is the element in the jth row and kth column of $\mathbf{C}$, then $c_{j,k}$ entails the amount of influence that the kth gene has on the jth gene. At each time instant, we observe a vector $\mathbf{y}_t \in \mathbb{R}^{d_x \times 1}$ that is related to the true gene expression vector $\mathbf{x}_t$ by the following observation equation:

$$\mathbf{y}_t = \mathbf{x}_t + \mathbf{v}_t, \tag{2}$$

where $\mathbf{v}_t \in \mathbb{R}^{d_x \times 1}$ is additive observation noise. Together, equation (1) and (2) define a SSM for the evolution of gene expressions in time. Our goal in this work is to determine the topology of the gene regulatory network, described by the adjacency matrix $\mathbf{A}$, whose element in the jth row and kth column is defined as

$$a_{j,k} = \begin{cases} 0, & \text{if } c_{j,k} = 0, \\ 1, & \text{if } c_{j,k} \neq 0 \end{cases}, \tag{3}$$

for $j = 1, \dots, d_x$ and $k = 1, \dots, d_x$.

3. PARTICLE GIBBS SAMPLING

Here, we describe the particle Gibbs sampling (PGS) approach for general SSMs. Consider the matrix of states $\mathbf{x}_{0:T} = [\mathbf{x}_0, \dots, \mathbf{x}_T] \in$

Algorithm 1 Particle Gibbs Sampling (PGS)

```
1: Initialization: Set $\mathbf{\Theta}^{(0)}$.
2: for $i = 1, \ldots, I$ do
3:     Sample $\mathbf{x}_{0:T}$ using a PF method
           $\mathbf{x}_{0:T}^{(i)} \sim \hat{p}_M(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta}^{(i-1)})$.
4:     Sample $\mathbf{\Theta}$ from the exact conditional
           $\mathbf{\Theta}^{(i)} \sim p(\mathbf{\Theta}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)})$.
5: end for
6: Output: Return $\mathcal{X} = \{\mathbf{x}_{0:T}^{(i)}, \mathbf{\Theta}^{(i)}\}_{i=I_0+1}^{I}$.
```

$\mathbb{R}^{d_x \times T+1}$ and the matrix of observations $\mathbf{y}_{1:T} = [\mathbf{y}_1, \ldots, \mathbf{y}_T] \in \mathbb{R}^{d_y \times T}$. Let $\mathbf{\Theta}$ denote all unknown parameters contained in the SSM of interest. To infer the joint posterior of $\mathbf{x}_{0:T}$ and $\mathbf{\Theta}$ given $\mathbf{y}_{1:T}$, one popular approach in the literature is Gibbs sampling [14]. Gibbs sampling obtains samples from the joint posterior $p(\mathbf{x}_{0:T}, \mathbf{\Theta}|\mathbf{y}_{1:T})$ by iteratively sampling from the full set of conditional distributions:

$$\mathbf{x}_{0:T}^{(i)} \sim p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta}^{(i-1)}),$$
$$\mathbf{\Theta}^{(i)} \sim p(\mathbf{\Theta}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}),$$

for $i = 1, \ldots, I$, where $\mathbf{\Theta}^{(0)}$ denotes an initial value for the model parameters and i is the iteration index of the Gibbs sampler. At the end of the algorithm, we obtain a Markov chain $\mathcal{X} = \{\mathbf{x}_{0:T}^{(i)}, \mathbf{\Theta}^{(i)}\}_{i=I_0+1}^{I}$, where each element of $\mathcal{X}$ is taken as a sample from the joint posterior $p(\mathbf{x}_{0:T}, \mathbf{\Theta}|\mathbf{y}_{1:T})$.[1]

Unfortunately, for most SSMs, it is impossible to directly sample from $p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta})$. To overcome this issue, a PGS method has been proposed [13]. In PGS, PF is used to obtain samples from $p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta})$. Particle filters are Monte Carlo schemes employed for state estimation in SSMs [15, 16]. PF methods use sequential importance sampling to construct an approximation to $p(\mathbf{x}_{0,T}|\mathbf{y}_{1:T}, \mathbf{\Theta})$ that is a discrete random measure composed of M particle streams and weights

$$\begin{aligned} p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta}) &\approx \hat{p}_M(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta}), \\ &= \sum_{m=1}^{M} w_T^{(m)} \delta(\mathbf{x}_{0:T} - \mathbf{x}_{0:T}^{(m)}), \end{aligned} \tag{4}$$

where $w_T^{(m)}$ denotes the normalized importance weight of the mth particle stream $\mathbf{x}_{0:T}^{(m)}$ and $\delta(\mathbf{x}_{0:T} - \mathbf{x}_{0:T}^{(m)})$ denotes a Dirac-delta function centered at $\mathbf{x}_{0:T}^{(m)}$. A sample is obtained from this approximation using multinomial resampling (see [13] for more information regarding sampling from (4) and the conditions for convergence of the PGS approach). Under the assumption that $p(\mathbf{\Theta}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T})$ can be obtained in closed form, the PGS method simply cycles between running a particle filter to obtain a sample from $p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{\Theta})$ and sampling model parameters $\mathbf{\Theta}$ from $p(\mathbf{\Theta}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T})$. We summarize the approach in Algorithm 1.

[1]The parameter I_0 is the burn-in period of the Gibbs sampler, which is necessary to discard samples obtained in the early iterations of the algorithm before the Markov chain had converged to the stationary distribution.

4. PROPOSED PGS FOR GENE TOPOLOGY ESTIMATION

We propose a novel method for gene network topology estimation which employs PGS on the SSM described by (1) and (2). We consider the process and observation noises to be Gaussian, i.e. $\mathbf{u}_t \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 \mathbb{I}_{d_x})$ and $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \sigma_v^2 \mathbb{I}_{d_x})$, where $\mathbb{I}_{d_x}$ denotes the $d_x \times d_x$ identity matrix. We assume that the nonlinear function $g(x_{r,t})$ is a sigmoid function given by

$$g(x_{r,t}) = \frac{1}{1 + e^{-x_{r,t}}}. \tag{5}$$

In this form, the unknowns in the problem are the gene expressions $\mathbf{x}_{0:T}$ and the coefficient matrix $\mathbf{C}$. A PGS method designed for inferring the posterior $p(\mathbf{x}_{0:T}, \mathbf{C}|\mathbf{y}_{1:T})$ would sample as follows:

$$\mathbf{x}_{0:T}^{(i)} \sim \hat{p}(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{C}^{(i-1)}),$$
$$\mathbf{C}^{(i)} \sim p(\mathbf{C}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}),$$

where $\hat{p}(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{C}^{(i-1)})$ is a particle-based approximation of $p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \mathbf{C}^{(i-1)})$ obtained from PF. Unfortunately, a PGS method that samples from $p(\mathbf{x}_{0:T}, \mathbf{C}|\mathbf{y}_{1:T})$ does not provide us with an elegant way to estimate the topology of the network. The task is even more challenging when we have a larger network at hand; more than one set of parameters can produce the same observations, resulting in an identifiability issue. Moreover, since the topologies of gene regulatory networks are known to be sparse, a mechanism for inducing sparsity into the coefficient matrix $\mathbf{C}$ is needed. To tackle this issue, we first propose to reformulate the problem statement.

Let $\mathbf{x}_{1:t,j} \in \mathbb{R}^{t \times 1}$ denote the evolution of gene j from time instant 1 up to time instant t, $\mathbf{G}_{t-1} \in \mathbb{R}^{t \times d_x}$ be a matrix whose elements are the nonlinear transformations of all genes up from time instant 0 up to time instant $t-1$, and $\mathbf{c}_j$ be the jth row of the coefficient matrix $\mathbf{C}$. In other words,

$$\begin{aligned} \mathbf{x}_{1:t,j} &= [x_{1,j},\ x_{2,j},\ \ldots,\ x_{t,j}]^\top, \\ \mathbf{G}_{t-1} &= [\mathbf{g}(\mathbf{x}_0),\ \mathbf{g}(\mathbf{x}_1),\ \ldots,\ \mathbf{g}(\mathbf{x}_{t-1})]^\top, \\ \mathbf{C} &= [\mathbf{c}_1,\ \mathbf{c}_2,\ \ldots,\ \mathbf{c}_{d_x}]^\top, \end{aligned}$$

We propose to augment a variable to the PGS method by decomposing the coefficient matrix as

$$\mathbf{C} = \mathbf{A} \circ \tilde{\mathbf{C}},$$

where $\tilde{\mathbf{C}} \in \mathbb{R}^{d_x \times d_x}$ is the coefficient matrix under the assumption that the network is fully connected, and $\circ$ denotes the Hadamard product. We can now write the evolution of gene j over time as

$$\mathbf{x}_{1:t,j} = \mathbf{G}_{t-1}(\mathbf{a}_j \circ \tilde{\mathbf{c}}_j) + \mathbf{u}_j, \tag{6}$$

where $\mathbf{u}_j \in \mathbb{R}^{t \times 1}$ is a vector of process noises, $\tilde{\mathbf{c}}_j = [\tilde{c}_{j,1}, \ldots, \tilde{c}_{j,d_x}]$ corresponds to the jth row of $\tilde{\mathbf{C}}$ and $\mathbf{a}_j$ corresponds to the jth row of $\mathbf{A}$. Our goal now, is to design a PGS method to iteratively sample $\mathbf{x}_{0:T}$ and the unknowns $\mathbf{\Theta} = \{\tilde{\mathbf{C}}, \mathbf{A}\}$. One implementation of such a scheme is to sample as follows:

$$\begin{aligned} \mathbf{x}_{0:T}^{(i)} &\sim \hat{p}(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \tilde{\mathbf{C}}^{(i-1)}, \mathbf{A}^{(i-1)}), \\ \tilde{\mathbf{C}}^{(i)} &\sim p(\tilde{\mathbf{C}}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}, \mathbf{A}^{(i-1)}), \\ \mathbf{A}^{(i)} &\sim p(\mathbf{A}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}, \tilde{\mathbf{C}}^{(i)}). \end{aligned} \tag{7}$$

In the remaining subsections, we elaborate in detail how PGS is implemented for the considered model.

Algorithm 2 PGS for Gene Regulatory Networks

```
1:  Initialization
2:  Set $\mathbf{C}^{(0)} \sim p(\mathbf{C})$,
3:  $\mathbf{a}_{j,k}^{(0)} = 1$, for $j = 1, \ldots d_x$, $k = 1, \ldots, d_x$
4:  $\mathbf{C}^{(0)} = \mathbf{A}^{(0)} \circ \tilde{\mathbf{C}}^{(0)}$
5:  for $i = 1 : I$ do
6:      Sample $\mathbf{x}_{0:T}$ using a PF method
            $\mathbf{x}_{0:T}^{(i)} \sim \hat{p}_M(\mathbf{x}_{0:T} | \mathbf{y}_{1:T}, \tilde{\mathbf{C}}^{(i-1)}, \mathbf{A}^{(i-1)})$.
7:      for $j = 1, \ldots, d_x$ do
8:          Compute $\boldsymbol{\mu}_{post}$ and $\boldsymbol{\Sigma}_{post}$ as in (14) and sample
                $\ddot{\mathbf{c}}_{0j}^{(i)} \sim p(\ddot{\mathbf{c}}_{0j})$
                $\ddot{\mathbf{c}}_{1j}^{(i)} \sim p(\ddot{\mathbf{c}}_{1j} | \mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}, \mathbf{a}_j^{(i-1)})$
                $\tilde{\mathbf{c}}_j^{(i)} = \mathbf{H}_{0j}^{(i)\top} \ddot{\mathbf{c}}_{0j}^{(i)} + \mathbf{H}_{1j}^{(i)\top} \ddot{\mathbf{c}}_{1j}^{(i)}$
9:      end for
10:     for $j = 1, ..., d_x$ do
11:         for $k = 1, ..., d_x$ do
12:             Compute $\alpha_{j,k}$ according to (18)
13:             Generate a uniform random number $u \sim \mathcal{U}(0, 1)$. If
                $u < \alpha_{j,k}$, then set $a_{j,k} = 1$. Otherwise, set $a_{j,k} = 0$.
14:         end for
15:     end for
16: end for
```

4.1. Sampling $\mathbf{x}_{0:T}$

Samples of the gene expressions are obtained using a PF method as describe in Section 3. In this work, since the considered SSM is only nonlinear in the state-transitions and the considered noises are additive and Gaussian, a PF with the optimal proposal distribution can be utilized [17].

4.2. Sampling $\tilde{\mathbf{C}}$

We sample the matrix $\tilde{\mathbf{C}}$ row by row by sampling $\tilde{\mathbf{c}}_j$ for $j = 1, \ldots, d_x$. When doing so, care must be given separately to the elements, as they are not all updated in the same manner. For a given element $\tilde{c}_{j,k}^{(i-1)}$, if the corresponding topology element $a_{j,k}^{(i-1)}$ is 0, then no information about $\tilde{c}_{j,k}$ is propagated to the next iteration i. In such a case, $\tilde{c}_{j,k}^{(i)}$ is obtained by sampling from a prior distribution. Otherwise, $\tilde{c}_{j,k}^{(i)}$ is updated by sampling from the conditional posterior distribution. For a given row $\tilde{\mathbf{c}}_j$, consider the vectors

$$\ddot{\mathbf{c}}_{0j} = \mathbf{H}_{0j} \tilde{\mathbf{c}}_j, \qquad \ddot{\mathbf{c}}_{1j} = \mathbf{H}_{1j} \tilde{\mathbf{c}}_j. \tag{8}$$

Here, $\mathbf{H}_{0j}$ is a transformation matrix that picks out the elements of $\tilde{\mathbf{c}}_j$ whose topology counterparts are equal to 0. Similarly, $\mathbf{H}_{1j}$ is a transformation matrix that picks out the elements of $\tilde{\mathbf{c}}_j$ whose corresponding topology element is 1. We assume that $p(\tilde{\mathbf{c}}_j) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\tilde{c}})$, for $j = 1, \ldots, d_x$. Since $\ddot{\mathbf{c}}_{0j}$ and $\ddot{\mathbf{c}}_{1j}$ are linear transformations of $\tilde{\mathbf{c}}_j$, their distributions are also Gaussian, i.e.

$$\ddot{\mathbf{c}}_{0j} \sim \mathcal{N}(\mathbf{0}, \mathbf{B}_{0j}), \qquad \ddot{\mathbf{c}}_{1j} \sim \mathcal{N}(\mathbf{0}, \mathbf{B}_{1j}) \tag{9}$$

with

$$\mathbf{B}_{0j} = \mathbf{H}_{0j} \boldsymbol{\Sigma}_{\tilde{c}} \mathbf{H}_{0j}{}^{\top}, \qquad \mathbf{B}_{1j} = \mathbf{H}_{1j} \boldsymbol{\Sigma}_{\tilde{c}} \mathbf{H}_{1j}{}^{\top}. \tag{10}$$

Furthermore, we define the vector $\ddot{\mathbf{g}}_{t-1} = \mathbf{H}_{1j} \mathbf{g}_{t-1}$ which carries all the information corresponding to the elements in $\ddot{\mathbf{c}}_{1j}$. Using Bayes theorem, we can now write

$$\begin{aligned} p(\ddot{\mathbf{c}}_{1j} | \mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}, \mathbf{a}_j^{(i-1)}) &\propto p(\mathbf{x}_{1:T,j} | \mathbf{G}_{T-1}, \mathbf{a}_j, \ddot{\mathbf{c}}_{1j}) p(\ddot{\mathbf{c}}_{1j}) \\ &\propto p(\ddot{\mathbf{c}}_{1j}) \prod_{t=1}^{T} \mathcal{N}(x_{t,j} | \ddot{\mathbf{g}}_{t-1}^{\top} \ddot{\mathbf{c}}_{1j}, \sigma_u^2). \end{aligned} \tag{11}$$

The product term in (11) can be expressed as

$$\begin{aligned} &\prod_{t=1}^{T} \mathcal{N}(x_{t,j} | \ddot{\mathbf{g}}_{t-1}^{\top} \ddot{c}_{1j}, \sigma_u^2) \\ &\propto \exp\left(-\frac{1}{2\sigma_u^2} \sum_{t=1}^{T} (-2\ddot{\mathbf{g}}_{t-1}^{\top} \ddot{\mathbf{c}}_{1j} x_{t,j} + \ddot{\mathbf{c}}_{1j}^{\top} \ddot{\mathbf{g}}_{t-1} \ddot{\mathbf{g}}_{t-1}^{\top} \ddot{\mathbf{c}}_j) \right), \end{aligned}$$

to attain $\mathcal{N}(\ddot{\mathbf{c}}_{1j} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with

$$\begin{aligned} \boldsymbol{\mu} &= \frac{1}{\sigma_u^2} \boldsymbol{\Sigma} \left(\sum_{t=1}^{T} \ddot{\mathbf{g}}_{t-1} x_{t,j} \right), \\ \boldsymbol{\Sigma} &= \sigma_u^2 \left(\sum_{t=1}^{T} \ddot{\mathbf{g}}_{t-1} \ddot{\mathbf{g}}_{t-1}^{\top} \right)^{-1}. \end{aligned} \tag{12}$$

Finally, we combine our result in (12) with the prior $p(\ddot{\mathbf{c}}_{1j}) = \mathcal{N}(\mathbf{0}, \mathbf{B}_{1j})$ to obtain the posterior

$$p(\ddot{\mathbf{c}}_{1j} | \mathbf{y}_{1:T}, \mathbf{x}_{0:T}^{(i)}, \mathbf{a}_j^{(i-1)}) \propto \mathcal{N}(\boldsymbol{\mu}_{post}, \boldsymbol{\Sigma}_{post}), \tag{13}$$

with

$$\begin{aligned} \boldsymbol{\mu}_{post} &= \boldsymbol{\Sigma}_{post} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}, \\ \boldsymbol{\Sigma}_{post} &= (\boldsymbol{\Sigma}^{-1} + \mathbf{B}_{1j}^{-1})^{-1}. \end{aligned} \tag{14}$$

Ultimately, we sample

$$\begin{aligned} \ddot{\mathbf{c}}_{0j} &\sim \mathcal{N}(\mathbf{0}, \mathbf{B}_{0j}) \\ \ddot{\mathbf{c}}_{1j} &\sim \mathcal{N}(\boldsymbol{\mu}_{post}, \boldsymbol{\Sigma}_{post}) \end{aligned} \tag{15}$$

and obtain $\tilde{\mathbf{c}}_j$ as

$$\tilde{\mathbf{c}}_j = \mathbf{H}_{0j}^{\top} \ddot{\mathbf{c}}_{0j} + \mathbf{H}_{1j}^{\top} \ddot{\mathbf{c}}_{1j}, \quad j = 1, \ldots, d_x. \tag{16}$$

4.3. Sampling A

We propose to sample the topology element by element. Here we readily deduce the conditional posterior of $a_{j,k}$ as

$$\begin{aligned} &p(a_{j,k} | \mathbf{x}_{1:T}, \mathbf{a}_{-j}, \mathbf{y}_{1:T}, \tilde{\mathbf{c}}_j) \\ &\qquad \propto p(\mathbf{x}_{1:T,j} | \mathbf{G}_{T-1}, \mathbf{a}_j, \tilde{\mathbf{c}}_j) p(a_{j,k}) \end{aligned} \tag{17}$$

for $j = 1, ..., d_x$ and $k = 1, ..., d_x$. Each $a_{j,k}$ can take up a value of either 0 or 1, so we choose the prior of $a_{j,k}$ as $a_{j,k} \sim \text{Bernoulli}(\rho)$. We draw the samples by computing the probability $\alpha_{j,k}$ defined by

$$\alpha_{j,k} = \frac{\alpha_{j,k}^{+}}{\alpha_{j,k}^{+} + \alpha_{j,k}^{-}}, \tag{18}$$

where we have that

$$\begin{aligned} \alpha_{j,k}^{+} &= p(\mathbf{x}_{1:T} | \mathbf{y}_{1:T}, a_{j,k} = 1, \mathbf{a}_{j,-k}, \tilde{\mathbf{c}}_j) p(a_{j,k} = 1) \\ \alpha_{j,k}^{-} &= p(\mathbf{x}_{1:T} | \mathbf{y}_{1:T}, a_{j,k} = 0, \mathbf{a}_{j,-k}, \tilde{\mathbf{c}}_j) p(a_{j,k} = 0), \end{aligned} \tag{19}$$

	Novel				LASSO	
$\sigma_u^2 = 1$	$\sigma_c^2 = 1$	$\sigma_c^2 = 10$	$\sigma_c^2 = 20$	$\sigma_c^2 = 100$	(1)	(2)
Precision	0.8922	1.0000	0.7971	1.0000	0.5646	0.8366
Recall	1.0000	0.8750	1.0000	1.0000	1.0000	0.9888
F-score	0.9429	0.9333	0.8870	**1.0000**	0.7214	0.9055

Table 1: Results for $\sigma_v^2 = 10^{-4}$ and $d_x = 4$.

	Novel				LASSO	
$\sigma_u^2 = 1$	$\sigma_c^2 = 1$	$\sigma_c^2 = 10$	$\sigma_c^2 = 20$	$\sigma_c^2 = 100$	(1)	(2)
Precision	0.8750	0.9867	1.0000	1.0000	0.6926	0.7149
Recall	0.8750	1.0000	1.0000	1.0000	0.9812	0.8788
F-score	0.8750	0.9929	**1.0000**	**1.0000**	0.8116	0.7869

Table 2: Results for $\sigma_v^2 = 10^{-2}$ and $d_x = 4$.

	Novel				LASSO	
$\sigma_u^2 = 1$	$\sigma_c^2 = 1$	$\sigma_c^2 = 10$	$\sigma_c^2 = 20$	$\sigma_c^2 = 100$	(1)	(2)
Precision	0.7735	0.9668	0.7587	0.7282	0.6819	0.9234
Recall	0.7677	0.5149	0.7308	0.7482	0.6936	0.3210
F-score	**0.7705**	0.6717	0.7443	0.7374	0.6876	0.4762

Table 3: Results for $\sigma_v^2 = 10^{-4}$ and $d_x = 8$.

	Novel				LASSO	
$\sigma_u^2 = 1$	$\sigma_c^2 = 1$	$\sigma_c^2 = 10$	$\sigma_c^2 = 20$	$\sigma_c^2 = 100$	(1)	(2)
Precision	0.8825	0.8681	0.8205	0.7605	0.6494	0.8366
Recall	0.5762	0.7010	0.7433	0.7900	0.5956	0.4251
F-score	0.6591	0.7754	**0.7799**	0.7791	0.6202	0.5656

Table 4: Results for $\sigma_v^2 = 10^{-2}$ and $d_x = 8$.

for $j = 1, ..., d_x$ and $k = 1, ..., d_x$. Here, $\alpha_{j,k}^{+}$ and $\alpha_{j,k}^{-}$ are the unnormalized probabilities that $a_{j,k} = 1$ and $a_{j,k} = 0$, respectively. We then set $a_{j,k} = 1$ with probability $\alpha_{j,k}$. The proposed implementation of the PGS method is summarized in Algorithm 2.

5. SIMULATIONS

The proposed method was demonstrated on synthetic data sets generated under a range of different conditions. In particular, we fixed the state noise of $\sigma_u^2 = 1$ and tested different observations noises $\sigma_v^2 \in \{10^{-4}, 10^{-2}\}$. We considered a network of size $d_x \in \{4, 8\}$. The number of Gibbs sampling iterations was fixed to $I = 5000$ and a burn-in of $I_0 = 2500$ was applied. PF was done using $M = 250$ particles. The prior covariance matrix of $\tilde{\mathbf{C}}$ was set to $\mathbf{\Sigma}_{\tilde{\mathbf{c}}} = \sigma_c^2 \mathbb{I}_{d_x}$, where we tested $\sigma_c^2 \in \{1, 10, 20, 100\}$. The prior probability of $a_{j,k} = 1$ was set to 0.3, i.e. $\rho = p(a_{j,k} = 1) = 0.3$ for $j = 1, \ldots, d_x, k = 1, \ldots, d_x$. At the end of the algorithm, we obtained an estimate of the adjacency matrix $\hat{\mathbf{A}}$ by independently taking the mode of the posterior samples for each element $a_{j,k}$. For the LASSO-based PF approach by [2], we used 10-fold cross-validation to determine the appropriate LASSO regression hyperparameter. We display results for two different settings of the method: (1) We select the hyperparameter that minimizes mean squared error; and (2) We select the hyperparameter that gives an error that is one standard deviation from the minimum mean squared error. For both methods, the number of correctly and incorrectly estimated values (true positives (TP), true negatives (TN), false positives (FP), false negatives (FN)) are collected and averaged over $R = 100$ runs. To evaluate the performance, the classical metrics of precision, recall, and F-score were used:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$
$$\text{F-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

The first set of data is generated from a network comprised of $d_x = 4$ genes, where the true coefficient matrix $\mathbf{C}$ is given by:

$$\mathbf{C} = \begin{bmatrix} 3 & 0 & 0 & -4.5 \\ -2.9 & 0 & 5 & 0 \\ -6 & 4 & 0 & 0 \\ 0 & -5 & 2 & 0 \end{bmatrix}.$$

The results on this data set, summarized in Tables 1 and 2 demonstrate excellent performance of our proposed method as compared to the LASSO approach. The results indicate some relationship between the precision, recall, and F-score with the prior variance σ_c^2, which can be further looked into. The second network is comprised of $d_x = 8$ genes and has a coefficient matrix that is given by

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0.6 & 0.7 & 0 & 1.9 & 2.9 \\ -0.1 & 0 & 0 & 3.5 & 0 & -2.1 & 0 & 3.4 \\ -4.4 & 0.9 & -1.7 & -0.3 & 3.4 & 0 & 1.7 & 0 \\ 0 & 0.5 & 2.8 & -3.7 & 0.9 & 0 & 0 & -3.1 \\ 0 & 0.2 & 0 & -2.6 & -3.2 & -0.1 & -0.5 & 4 \\ -0.5 & -1.8 & 0 & 3.4 & 1.4 & 1.1 & 0 & -1.7 \\ -0.8 & 0 & 0 & -3 & 1.1 & 0.4 & 0 & 0 \\ -0.3 & 0 & -1 & 0 & 0.1 & 0 & 0 & 2.2 \end{bmatrix}.$$

Results for the $d_x = 8$ network are shown in Tables 3 and 4. The estimation of a select number of coefficients of the 8 x 8 example is shown in fig 1. The challenge here rises not only from the increased dimension, but also from the the nonzero coefficients that are close to 0. Thus, the overall performance was not as optimistic as for the 4×4 matrix, yet still promising when compared to the LASSO method.

6. CONCLUSIONS

In this work, we proposed a novel methodology for inferring the topology of a gene regulatory network. We considered a Bayesian treatment of the problem, where the time evolution of gene expressions is modeled via a nonlinear state-space model (SSM). We derived a particle Gibbs sampling method to jointly estimate the gene expressions and the unknown parameters of the SSM, which included the topology of the gene network. Simulations on synthetic data demonstrated superiority of the proposed scheme over an approach based on least absolute shrinkage and selection operator regression.

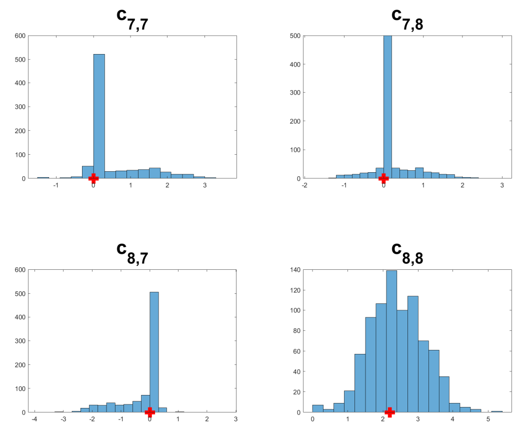

Fig. 1: Histograms for some of the entries of the coefficient matrix. The red marks denote the true values.

7. REFERENCES

[1] D. Anastassiou, "Genomic signal processing," *IEEE signal processing magazine*, vol. 18, no. 4, pp. 8–20, 2001.

[2] A. Noor, E. Serpedin, M. Nounou, and H. Nounou, "Inferring gene regulatory networks via nonlinear state-space models and exploiting sparsity," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 4, pp. 1203–1211, 2012.

[3] C. Chang, Z. Ding, Yeung S. Hung, and P. C. W. Fung, "Fast network component analysis (fastnca) for gene regulatory network reconstruction from microarray data," *Bioinformatics*, vol. 24, no. 11, pp. 1349–1358, 2008.

[4] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[5] A. Papana, C. Kyrtsou, D. Kugiumtzis, and C. Diks, "Financial networks based on Granger causality: A case study," *Physica A: Statistical Mechanics and its Applications*, vol. 482, pp. 65–73, 2017.

[6] D. Batavia and A. Tatu, "Estimating graph topology from sparse graph signals with an application to image denoising," in *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2017, pp. 1–6.

[7] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, D*, vol. 82, pp. 35–44, 1960.

[8] Z. Wang, X. Liu, Y. Liu, J. Liang, and V. Vinciotti, "An extended Kalman filtering approach to modeling nonlinear dynamic gene regulatory networks via short gene expression time series," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 6, no. 3, pp. 410–419, 2009.

[9] A. Noor, E. Serpedin, M. Nounou, and H. Nounou, "Reverse engineering sparse gene regulatory networks using cubature Kalman filter and compressed sensing," *Advances in bioinformatics*, vol. 2013, 2013.

[10] X. Shen and H. Vikalo, "Inferring parameters of gene regulatory networks via particle filtering," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, pp. 5, 2010.

[11] M. F. Bugallo, Ç. Taşdemir, and P. M. Djurić, "Estimation of gene expression by a bank of particle filters," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, 2015, pp. 494–498.

[12] Ç. Taşdemir, M. F. Bugallo, and P. M. Djurić, "A particle-based approach for topology estimation of gene networks," in *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2017, pp. 1–5.

[13] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[14] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian data analysis*, Chapman and Hall/CRC, 2013.

[15] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *IEEE signal processing magazine*, vol. 20, no. 5, pp. 19–38, 2003.

[16] A. Smith, *Sequential Monte Carlo methods in practice*, Springer Science & Business Media, 2013.

[17] C. Snyder, "Particle filters, the "optimal" proposal and high-dimensional systems," in *Proceedings of the ECMWF Seminar on Data Assimilation for atmosphere and ocean*, 2011, pp. 1–10.