EURASIP Journal on Bioinformatics and Systems Biology
a SpringerOpen Journal

RESEARCH Open Access

Regularized EM algorithm for sparse parameter estimation in nonlinear dynamic systems with application to gene regulatory network inference

Bin Jia[1] and Xiaodong Wang[2*]

Abstract

Parameter estimation in dynamic systems finds applications in various disciplines, including system biology. The well-known expectation-maximization (EM) algorithm is a popular method and has been widely used to solve system identification and parameter estimation problems. However, the conventional EM algorithm cannot exploit the sparsity. On the other hand, in gene regulatory network inference problems, the parameters to be estimated often exhibit sparse structure. In this paper, a regularized expectation-maximization (rEM) algorithm for sparse parameter estimation in nonlinear dynamic systems is proposed that is based on the maximum *a posteriori* (MAP) estimation and can incorporate the sparse prior. The expectation step involves the forward Gaussian approximation filtering and the backward Gaussian approximation smoothing. The maximization step employs a re-weighted iterative thresholding method. The proposed algorithm is then applied to gene regulatory network inference. Results based on both synthetic and real data show the effectiveness of the proposed algorithm.

Keywords: Nonlinear dynamic system; Parameter estimation; Sparsity; Expectation-maximization; Forward-backward recursion; Gaussian approximation; Gene regulatory network

1 Introduction

The dynamic system is a widely used modeling tool that finds applications in many engineering disciplines. Techniques for state estimation in dynamic systems have been well established. Recently, the problem of *sparse* state estimate has received significant interest. For example, various approaches to *static* sparse state estimation have been developed in [1-4], where the problem is essentially an underdetermined inverse problem, i.e., the number of measurements is small compared to the number of states. Extensions of these methods for *dynamic* sparse state estimation have been addressed in [5-7].

The expectation-maximization (EM) algorithm has also been applied to solve the sparse state estimate problem in dynamic systems [8-12]. In particular, in [8-10], the EM algorithm is employed to update the parameters of the Bernoulli-Gaussian prior and the measurement noise. These parameters are then used in the generalized approximate message passing algorithm [8-10]. In [12,13], the EM algorithm is used to iteratively estimate the parameters that describe the prior distribution and noise variances. Moreover, in [14], the EM algorithm is used for blind identification, where the sparse state is explored. Note that in the above works, only *linear* dynamic systems are considered.

In this paper, we focus on the *sparse parameter* estimation problem instead of the sparse state estimation problem. We consider a general *nonlinear* dynamic system, where both the state equation and the measurement equation are parameterized by some unknown parameters which are assumed to be sparse. One particular application is the inference of gene regulatory networks. The gene regulatory network can be modeled by the state-space model [15], in which the gene regulations are represented by the unknown parameters. The gene regulatory network

*Correspondence: wangx@ee.columbia.edu
[2]Department of Electrical Engineering, Columbia University, New York, NY 10027, USA
Full list of author information is available at the end of the article

is known to be sparse due to the fact that a gene directly regulates or is regularized by a small number of genes [16-19]. The EM algorithm has been applied to parameter estimation in dynamic systems [20]. However, the EM algorithm cannot exploit the sparsity of the parameters. Here, we propose a regularized expectation-maximization (rEM) algorithm for sparse parameter estimation in nonlinear dynamic systems. Specifically, the sparsity of the parameters is imposed by a Laplace prior and we consider the approximate maximum *a posteriori* (MAP) estimate of the parameters. It should be emphasized that the proposed method is an approximate MAP-EM algorithm based on various Gaussian assumptions and quadrature procedures for approximating Gaussian integrals. Note that the MAP-EM algorithm may get stuck at local minima or saddle points. Similar to the conventional EM algorithm, the rEM algorithm also consists of an expectation step and a maximization step. The expectation step involves the forward Gaussian approximation filtering and the backward Gaussian approximation smoothing. The maximization step involves solving an ℓ_1 minimization problem for which a re-weighted iterative thresholding algorithm is employed. To illustrate the proposed sparse parameter estimation method in dynamic systems, we consider the gene-regulatory network inference based on gene expression data.

The unscented Kalman filter has been used in the inference of gene regulatory network [15,21,22]. However, the methods proposed in [15,21,22] are fundamentally different with the method proposed in this paper. Firstly, the unscented Kalman filter is only used once in [15,21,22], while it is used in each iteration of the rEM algorithm in this paper. Secondly, not only the unscented Kalman filter but also the unscented Kalman smoother is used in our proposed rEM algorithm. In essence, only the observation before time k is used to the estimation at time k in the unscented Kalman filter. However, in our rEM algorithm, all observation data is used to the estimation at time k (by the unscented Kalman smoother). The fundamental difference between the proposed work and that of [9] is that the proposed work is for the sparse parameter estimation problem of the dynamic system, while that of [9] is only for the sparse parameter estimation of the static problem. In addition, a general nonlinear dynamic system is involved in our work and only linear system is involved in the work of [9]. The main difference between the proposed work and that of [23] is that the sparsity constraint is enforced. The main contribution of this paper is to use the sparsity-enforced EM algorithm to solve the sparse parameter estimation problem. In addition, the reweighted iterative threshold algorithm is proposed to solve the ℓ_1 optimization algorithm. To the best knowledge of the authors, the proposed rEM with the reweighted iterative threshold optimization algorithm is innovative. Furthermore, we have systematically investigated the performance of the proposed algorithm and compared the results with other conventional algorithms.

The remainder of this paper is organized as follows. In Section 2, the problem of the sparse parameter estimation in dynamic systems is introduced and the regularized EM algorithm is formulated. In Section 3, the E-step of rEM that involves forward-backward recursions and Gaussian approximations is discussed. Section 4 discusses the ℓ_1 optimization problem involved in the maximization step. Application of the proposed rEM algorithm to gene regulatory network inference is discussed in Section 5. Concluding remarks are given in Section 6.

2 Problem statement and the MAP-EM algorithm

We consider a general discrete-time nonlinear dynamic system with unknown parameters, given by the following state and measurement equations:

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) + \boldsymbol{u}_k, \tag{1}$$

$$\text{and } \boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta}) + \boldsymbol{v}_k, \tag{2}$$

where $\boldsymbol{x}_k$ and $\boldsymbol{y}_k$ are the state vector and the observation vector at time k, respectively; $\boldsymbol{\theta}$ is the unknown parameter vector; $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$ are two nonlinear functions; $\boldsymbol{u}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{U}_k)$ is the process noise, and $\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k)$ is the measurement noise. It is assumed that both $\{\boldsymbol{u}_k\}$ and $\{\boldsymbol{v}_k\}$ are independent noise processes and they are mutually independent. Note that the nonlinear functions $\boldsymbol{f}$ and $\boldsymbol{h}$ are assumed to be differentiable.

Define the notation $\boldsymbol{Y}^k \triangleq [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_k]$. The problem considered in this paper is to estimate the unknown system parameter vector $\boldsymbol{\theta}$ from the length-K measurement data $\boldsymbol{Y}^K$. We assume that $\boldsymbol{\theta}$ is *sparse*. In particular, it has a Laplacian prior distribution which is commonly used as a sparse prior,

$$p(\boldsymbol{\theta}) = \prod_{i=1}^{m} \frac{\lambda_i}{2} e^{-\lambda_i |\theta_i|}. \tag{3}$$

In the EM algorithm and the MAP-EM algorithm [23], given an estimate $\boldsymbol{\theta}'$, a new estimate $\boldsymbol{\theta}''$ is given by

$$\boldsymbol{\theta}'' = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}'), \tag{4}$$

and

$$\boldsymbol{\theta}'' = \arg\max_{\boldsymbol{\theta}} \left[Q(\boldsymbol{\theta}, \boldsymbol{\theta}') + \log p(\boldsymbol{\theta}) \right], \tag{5}$$

respectively.

Note that the regularized EM can be viewed as a special MAP-EM. To differentiate the sparsity-enforced EM algorithm from the general MAP-EM algorithm, rEM is used. In this paper, the following assumptions are made. (1) The probability density function of the state is assumed to be Gaussian. The Bayesian filter is optimal; however,

exact finite-dimensional solutions do not exist. Hence, numerical approximation has to be made. The Gaussian approximation is frequently assumed due to the relatively low complexity and high accuracy [24-26]. (2) The integrals are approximated by various quadrature methods. Many numerical rules, such as Gauss-Hermite quadrature [25], unscented transformation [27], cubature rule [24], and the sparse grid quadrature [26], as well as the Monte Carlo method [28], can be used to approximate the integral. However, the quadrature rule is the best when computational complexity and accuracy are both considered [29].

We next consider the expression of the Q-function in (5). Due to the Markovian structure of the state-space model (1) to (2), we have

$$p(\boldsymbol{X}^K, \boldsymbol{Y}^K|\boldsymbol{\theta}) = p(\boldsymbol{x}_1|\boldsymbol{\theta}) \prod_{k=2}^{K} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) \prod_{k=1}^{K} p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{\theta}). \tag{6}$$

Therefore,

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}') &= \int \log p(\boldsymbol{X}^K, \boldsymbol{Y}^K|\boldsymbol{\theta}) p(\boldsymbol{X}^K|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{X}^K \\ &= \int \log p(\boldsymbol{x}_1|\boldsymbol{\theta}) p(\boldsymbol{x}_1|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{x}_1 \\ &\quad + \sum_{k=2}^{K} \int \underbrace{\log p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{\theta})}_{-\frac{1}{2}(\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}))^T \boldsymbol{U}_k^{-1} (\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})) - c_k} p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{x}_{k-1} \mathrm{d}\boldsymbol{x}_k \\ &\quad + \sum_{k=1}^{K} \int \underbrace{\log p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{\theta})}_{-\frac{1}{2}(\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta}))^T \boldsymbol{R}_k^{-1} (\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta})) - d_k} p(\boldsymbol{x}_k|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{x}_k, \end{aligned} \tag{7}$$

where $c_k \triangleq \frac{1}{2}[\log|\boldsymbol{U}_k| + \dim(\boldsymbol{x}_k)\log(2\pi)]$ and $d_k \triangleq \frac{1}{2}[\log|\boldsymbol{R}_k| + \dim(\boldsymbol{y}_k)\log(2\pi)]$. We assume that the initial state $\boldsymbol{x}_1$ is independent of the parameter $\boldsymbol{\theta}$. Hence, with the prior given in (3), the optimization in (5) can be rewritten as

$$\begin{aligned} \boldsymbol{\theta}'' &= \arg\max_{\boldsymbol{\theta}} \left[Q(\boldsymbol{\theta}, \boldsymbol{\theta}') + \log p(\boldsymbol{\theta})\right] \\ &= \arg\min_{\boldsymbol{\theta}} \sum_{k=2}^{K} \left\{ 2c_k + \int \left[(\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})^T \boldsymbol{U}_k^{-1} (\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta})\right] \right. \\ &\qquad \left. \times p\ (\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{x}_{k-1} \mathrm{d}\boldsymbol{x}_k \right\} \\ &\quad + \sum_{k=1}^{K} \left\{ 2d_k + \int \left[(\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta})^T \boldsymbol{R}_k^{-1} (\boldsymbol{y}_k - \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\theta})\right] p(\boldsymbol{x}_k|\boldsymbol{Y}^K, \boldsymbol{\theta}') \mathrm{d}\boldsymbol{x}_k \right\} \\ &\quad + 2\|\boldsymbol{\lambda} \circ \boldsymbol{\theta}\|_1, \end{aligned} \tag{8}$$

where $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \cdots, \lambda_m]^T$, and '$\circ$' denotes the pointwise multiplication.

Note that in many applications, the unknown parameters $\boldsymbol{\theta}$ are only related to the state equation, but not to the measurement equation. Therefore, the second term in (8) can be removed. In the next section, we discuss the procedures for computing the densities $p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{Y}^K, \boldsymbol{\theta}')$ and $p(\boldsymbol{x}_k|\boldsymbol{Y}^K, \boldsymbol{\theta}')$, the integrals, and the minimization in (8).

3 The E-step: computing the *Q*-function

We first discuss the calculation of the probability density functions of the states $p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{Y}^K, \boldsymbol{\theta}')$ and $p(\boldsymbol{x}_k|\boldsymbol{Y}^K, \boldsymbol{\theta}')$ in (8), which involves a forward recursion of a point-based Gaussian approximation filter to compute $p(\boldsymbol{x}_k|\boldsymbol{Y}^k, \boldsymbol{\theta}')$ and $p(\boldsymbol{x}_{k+1}|\boldsymbol{Y}^k, \boldsymbol{\theta}')$, $k = 1, 2, ..., K$, and a backward recursion of a point-based Gaussian approximation smoother to compute $p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1}|\boldsymbol{Y}^K, \boldsymbol{\theta}')$ and $p(\boldsymbol{x}_k|\boldsymbol{Y}^K, \boldsymbol{\theta}')$, $k = K, K-1, ..., 1$. For notational simplicity, in the remainder of this section, we drop the parameter $\boldsymbol{\theta}'$.

3.1 Forward recursion

The forward recursion is composed of two steps: prediction and filtering. Specifically, given the prior probability density function (PDF) $p(\boldsymbol{x}_{k-1}|\boldsymbol{Y}^{k-1})$ at time $k-1$, we need to compute the predicted conditional PDF $p(\boldsymbol{x}_k|\boldsymbol{Y}^{k-1})$; then, given the measurement $\boldsymbol{y}_k$ at time k, we update the filtered PDF $p(\boldsymbol{x}_k|\boldsymbol{Y}^k)$. These PDF recursions are in general computationally intractable unless the system is linear and Gaussian. The Gaussian approximation filters are based on the following two assumptions: (1) Given $\boldsymbol{Y}^{k-1}$, $\boldsymbol{x}_{k-1}$ has a Gaussian distribution, i.e., $\boldsymbol{x}_{k-1}|\boldsymbol{Y}^{k-1} \sim \mathcal{N}(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1})$; and (2) $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ are jointly Gaussian, given $\boldsymbol{Y}^{k-1}$.

It then follows that the predictive PDF is Gaussian, i.e., $\boldsymbol{x}_k|\boldsymbol{Y}^{k-1} \sim \mathcal{N}(\hat{\boldsymbol{x}}_{k|k-1}, \boldsymbol{P}_{k|k-1})$, with [24,26,27]

$$\hat{\boldsymbol{x}}_{k|k-1} \triangleq \mathbb{E}\{\boldsymbol{x}_k|\boldsymbol{Y}^{k-1}\} = \mathbb{E}_{\boldsymbol{x}_{k-1}|\boldsymbol{Y}^{k-1}} \left\{\boldsymbol{f}(\boldsymbol{x}_{k-1})\right\}, \tag{9}$$

$$\boldsymbol{P}_{k|k-1} \triangleq \mathrm{Cov}\{\boldsymbol{x}_k|\boldsymbol{Y}^{k-1}\} = \mathbb{E}_{\boldsymbol{x}_{k-1}|\boldsymbol{Y}^{k-1}} \left\{(\boldsymbol{f}(\boldsymbol{x}_{k-1}) - \hat{\boldsymbol{x}}_{k|k-1})(\boldsymbol{f}(\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k|k-1})^T\right\} + \boldsymbol{U}_{k-1}, \tag{10}$$

where $\mathbb{E}_{\boldsymbol{x}_{k-1}|\boldsymbol{Y}^{k-1}} \left\{\boldsymbol{g}(\boldsymbol{x}_{k-1})\right\} = \int \boldsymbol{g}(\boldsymbol{x})\phi(\boldsymbol{x}; \hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1})\mathrm{d}\boldsymbol{x}$, and $\phi\left(\boldsymbol{x}; \hat{\boldsymbol{x}}, \boldsymbol{P}\right)$ denotes the multivariate Gaussian PDF with mean $\hat{\boldsymbol{x}}$ and covariance $\boldsymbol{P}$.

Moreover, the filtered PDF is also Gaussian, i.e., $\boldsymbol{x}_k|\boldsymbol{Y}^k \sim \mathcal{N}(\hat{\boldsymbol{x}}_{k|k}, \boldsymbol{P}_{k|k})$ [24,26,27], where

$$\hat{\boldsymbol{x}}_{k|k} \triangleq \mathbb{E}\{\boldsymbol{x}_k|\boldsymbol{Y}^k\} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{L}_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_{k|k-1}), \tag{11}$$

$$\text{and} \quad \boldsymbol{P}_{k|k} \triangleq \mathrm{Cov}\{\boldsymbol{x}_k|\boldsymbol{Y}^k\} = \boldsymbol{P}_{k|k-1} - \boldsymbol{L}_k \boldsymbol{P}_k^{xy}, \tag{12}$$

with

$$\hat{\boldsymbol{y}}_{k|k-1} = \mathbb{E}_{\boldsymbol{x}_k|\boldsymbol{Y}^{k-1}} \{\boldsymbol{h}(\boldsymbol{x}_k)\}, \tag{13}$$

$$\boldsymbol{L}_k = \boldsymbol{P}_k^{xy}(\boldsymbol{R}_k + \boldsymbol{P}_k^{yy})^{-1}, \tag{14}$$

$$\boldsymbol{P}_k^{xy} = \mathbb{E}_{\boldsymbol{x}_k|\boldsymbol{Y}^{k-1}}\left\{(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k-1})(\boldsymbol{h}(\boldsymbol{x}_k) - \hat{\boldsymbol{y}}_{k|k-1})^T\right\}, \tag{15}$$

$$\boldsymbol{P}_k^{yy} = \mathbb{E}_{\boldsymbol{x}_k|\boldsymbol{Y}^{k-1}}\left\{(\boldsymbol{h}(\boldsymbol{x}_k) - \hat{\boldsymbol{y}}_{k|k-1})(\boldsymbol{h}(\boldsymbol{x}_k) - \hat{\boldsymbol{y}}_{k|k-1})^T\right\}. \tag{16}$$

3.2 Backward recursion

In the backward recursion, we compute the smoothed PDFs $p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{Y}^K)$ and $p(\boldsymbol{x}_k|\boldsymbol{Y}^K)$. Here, the approximate assumption made is that conditioned on $\boldsymbol{y}^k$, $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$ are jointly Gaussian [30], i.e.,

$$\begin{bmatrix}\boldsymbol{x}_k \\ \boldsymbol{x}_{k+1}\end{bmatrix} \mid \boldsymbol{Y}^k \sim \mathcal{N}\left(\begin{bmatrix}\hat{\boldsymbol{x}}_{k|k} \\ \hat{\boldsymbol{x}}_{k+1|k}\end{bmatrix}, \times \begin{bmatrix}\boldsymbol{P}_{k|k} & \boldsymbol{C}_k \\ \boldsymbol{C}_k^T & \boldsymbol{P}_{k+1|k}\end{bmatrix}\right), \tag{17}$$

with $\boldsymbol{C}_k \triangleq \mathrm{Cov}\{\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{Y}^k\}$

$$= \mathbb{E}_{\boldsymbol{x}_k|\boldsymbol{Y}_k}\left\{(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k})(\boldsymbol{f}(\boldsymbol{x}_k) - \hat{\boldsymbol{x}}_{k+1|k})^T\right\}. \tag{18}$$

Due to the Markov property of the state-space model, we have $p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{Y}^K) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{Y}^k)$. Therefore, we can write [30]

$$\begin{aligned} p(\boldsymbol{x}_k, \boldsymbol{x}_{k+1}|\boldsymbol{Y}^K) &= p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{Y}^K)p(\boldsymbol{x}_{k+1}|\boldsymbol{Y}^K) \\ &= p(\boldsymbol{x}_k|\boldsymbol{x}_{k+1}, \boldsymbol{Y}^k)p(\boldsymbol{x}_{k+1}|\boldsymbol{Y}^K). \end{aligned} \tag{19}$$

Now, assume that

$$\boldsymbol{x}_{k+1}|\boldsymbol{Y}^K \sim \mathcal{N}(\tilde{\boldsymbol{x}}_{k+1}, \tilde{\boldsymbol{P}}_{k+1}), \text{ with } \tilde{\boldsymbol{x}}_K = \hat{\boldsymbol{x}}_{K|K}, \ \tilde{\boldsymbol{P}}_K = \boldsymbol{P}_{K|K}. \tag{20}$$

It then follows from (17) and (19) that [30]

$$\begin{bmatrix}\boldsymbol{x}_k \\ \boldsymbol{x}_{k+1}\end{bmatrix}|\boldsymbol{Y}^K \sim \mathcal{N}\left(\begin{bmatrix}\tilde{\boldsymbol{x}}_k \\ \tilde{\boldsymbol{x}}_{k+1}\end{bmatrix}, \begin{bmatrix}\tilde{\boldsymbol{P}}_k & \boldsymbol{D}_k\tilde{\boldsymbol{P}}_{k+1} \\ \tilde{\boldsymbol{P}}_{k+1}\boldsymbol{D}_k^T & \tilde{\boldsymbol{P}}_{k+1}\end{bmatrix}\right), \tag{21}$$

where

$$\tilde{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_{k|k} + \boldsymbol{D}_k(\tilde{\boldsymbol{x}}_{k+1} - \hat{\boldsymbol{x}}_{k+1|k}), \tag{22}$$

$$\tilde{\boldsymbol{P}}_k = \boldsymbol{P}_{k|k} + \boldsymbol{D}_k(\tilde{\boldsymbol{P}}_{k+1} - \boldsymbol{P}_{k+1|k})\boldsymbol{D}_k^T, \tag{23}$$

$$\boldsymbol{D}_k = \boldsymbol{C}_k\boldsymbol{P}_{k+1|k}^{-1}. \tag{24}$$

3.3 Approximating the integrals

The integrals associated with the expectations in the forward-backward recursions for computing the approximate state PDFs, i.e., (9), (10), (13), (15), (16), and (18), as well as the integrals involved in computing the function $Q(\boldsymbol{\theta}, \boldsymbol{\theta}')$ in (8), are integrals of Gaussian type that can be efficiently approximated by various quadrature methods. Specifically, if a set of weighted points $\{(\boldsymbol{\gamma}_i, w_i), i = 1, \ldots, N\}$ can be used to approximate the integral

$$\mathbb{E}_{\mathcal{N}(\boldsymbol{0},\boldsymbol{I})}\{\boldsymbol{g}(\boldsymbol{x})\} = \int \boldsymbol{g}(\boldsymbol{x})\phi(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I})\,\mathrm{d}\boldsymbol{x} \approx \sum_{i=1}^{N} w_i\boldsymbol{g}(\boldsymbol{\gamma}_i), \tag{25}$$

then the general Gaussian-type integral can be approximated by

$$\mathbb{E}_{\mathcal{N}(\hat{\boldsymbol{x}},\boldsymbol{P})}\{\boldsymbol{g}(\boldsymbol{x})\} = \int \boldsymbol{g}(\boldsymbol{x})\phi(\boldsymbol{x}; \hat{\boldsymbol{x}}, \boldsymbol{P})\,\mathrm{d}\boldsymbol{x} \approx \sum_{i=1}^{N} w_i\boldsymbol{g}(\boldsymbol{S}\boldsymbol{\gamma}_i + \hat{\boldsymbol{x}}), \tag{26}$$

where $\boldsymbol{P} = \boldsymbol{S}\boldsymbol{S}^T$ and $\boldsymbol{S}$ can be obtained by Cholesky decomposition or singular value decomposition (SVD).

By using different point sets, different Gaussian approximation filters and smoothers can be obtained, such as the Gauss-Hermite quadrature (GHQ) [25], the unscented transform (UT) [27], the spherical-radial cubature rule (CR) [24], the sparse grid quadrature rule (SGQ) [26], and the quasi Monte Carlo method (QMC) [28]. Both the UT and the CR are the third-degree numerical rules which means the integration can be exactly calculated when $\boldsymbol{g}(\boldsymbol{x})$ is a polynomial with the degree up to three. In addition, the form of the CR is identical to the UT with a specific parameter. The main advantage of the UT and the CR is that the number of points required by the rule increases linearly with the dimension. However, one problem of the UT and the CR is that the high-order information of the nonlinear function is difficult to capture so that the accuracy may be low when $\boldsymbol{g}(\boldsymbol{x})$ is a highly nonlinear function. The GHQ rule, in contrast, can capture arbitrary degree information of $\boldsymbol{g}(\boldsymbol{x})$ by using more points. It has been proven that GHQ can provide more accurate results than the UT or the CR [25,26]. Similarly, the QMC method can also obtain more accurate results than the UT. However, both the GHQ rule and the QMC method require a large number of points for the high-dimensional problem. Specifically, the number of points required by the GHQ rule increases exponentially with the dimension. To achieve a similar performance of the GHQ with a small number of points, the SGQ is proposed [26], where the number of points increases only polynomially with the dimension.

For the numerical results in this paper, the UT is used in the Gaussian approximation filter and smoother, where we have $N = 2n + 1$, with n being the dimension of the state vector $\boldsymbol{x}_k$. The quadrature points and the corresponding weights are given, respectively, by

$$\boldsymbol{\gamma}_i = \begin{cases} \mathbf{0}, & i = 1, \\ \sqrt{(n+\kappa)}\boldsymbol{e}_{i-1}, & i = 2, \cdots, n+1, \\ -\sqrt{(n+\kappa)}\boldsymbol{e}_{i-n-1}, & i = n+2, \cdots, 2n+1, \end{cases} \tag{27}$$

and

$$w_i = \begin{cases} \dfrac{\kappa}{n+\kappa}, & i = 1, \\ \dfrac{1}{2(n+\kappa)}, & i = 2, \cdots, 2n+1, \end{cases} \tag{28}$$

where κ is a tunable parameter, and $\boldsymbol{e}_i$ is the ith n dimensional unit vector. Note that $\kappa = 0$ is used as the default value in this paper, as in the cubature Kalman filter [24]. In addition, $\kappa = -3$ can also be used as in the unscented Kalman filter [27].

4 The M-step: solving the ℓ_1 optimization problem

Solving the ℓ_1 optimization problems in (8) is not trivial since $|\theta_i|$ is nondifferentiable at $\theta_i = 0$. The ℓ_1 optimization is a useful tool to obtain sparse solutions. Methods for solving linear inverse problems with sparse constraints are reviewed in [1]. Some more recent developments include the projected scaled subgradient [31] method, the gradient support pursuit method [32], and the greedy sparse-simplex method [33]. In this paper, for the maximization step in the proposed rEM algorithm, due to the simplicity of implementation, we will employ a modified version of the iterative thresholding algorithm.

4.1 Iterative thresholding algorithm

Denote $\tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ as the two summation terms in (8). We consider the optimization problem in (8)

$$\arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}') + 2\|\boldsymbol{\lambda} \circ \boldsymbol{\theta}\|_1. \tag{29}$$

The solution to (29) can be iteratively obtained by solving a sequence of optimization problems [34]. As in the Newton's method, the Taylor series expansion of the $\tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}')$ around the solution $\boldsymbol{\theta}^t$ at the tth iteration is given by

$$\tilde{Q}(\boldsymbol{\theta}^t + \Delta\boldsymbol{\theta}, \boldsymbol{\theta}') \cong \tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}') + \Delta\boldsymbol{\theta}^T \nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}') + \frac{\alpha_t}{2}\|\Delta\boldsymbol{\theta}\|_2^2, \tag{30}$$

where $\nabla\tilde{Q}$ is the gradient of the negative Q-function and α_t is such that $\alpha_t \boldsymbol{I}$ mimics the Hessian $\nabla^2\tilde{Q}$. Then, $\boldsymbol{\theta}^{t+1}$ is given by

$$\boldsymbol{\theta}^{t+1} = \arg\min_{\boldsymbol{z}} (\boldsymbol{z} - \boldsymbol{\theta}^t)^T \nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}') + \frac{\alpha_t}{2}\|\boldsymbol{z} - \boldsymbol{\theta}^t\|_2^2 + 2\|\boldsymbol{\lambda} \circ \boldsymbol{z}\|_1, \tag{31}$$

where $\boldsymbol{z}$ denotes the variable to be optimized in the objective function.

The equivalent form of (31) is given by

$$\boldsymbol{\theta}^{t+1} = \arg\min_{\boldsymbol{z}} \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{u}^t\|_2^2 + \frac{2}{\alpha_t}\|\boldsymbol{\lambda} \circ \boldsymbol{z}\|_1, \tag{32}$$

$$\text{with} \quad \boldsymbol{u}^t = \boldsymbol{\theta}^t - \frac{1}{\alpha_t}\nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}'), \tag{33}$$

$$\alpha_t \approx \frac{(\boldsymbol{s}^t)^T \boldsymbol{r}^t}{\|\boldsymbol{s}^t\|^2}, \tag{34}$$

$$\boldsymbol{s}^t = \boldsymbol{\theta}^t - \boldsymbol{\theta}^{t-1}, \tag{35}$$

$$\boldsymbol{r}^t = \nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}') - \nabla\tilde{Q}(\boldsymbol{\theta}^{t-1}, \boldsymbol{\theta}'). \tag{36}$$

Note that Equation 34 is derived as follows. Because we require that $\alpha_t \boldsymbol{I}$ mimics the Hessian $\nabla^2\tilde{Q}$, i.e., $\alpha_t \boldsymbol{s}^t \approx \boldsymbol{r}^t$, solving α_t in the least-squares sense, we have

$$\alpha_t \approx \arg\min_{\alpha} \|\alpha\boldsymbol{s}^t - \boldsymbol{r}^t\|_2^2 = \frac{(\boldsymbol{s}^t)^T \boldsymbol{r}^t}{\boldsymbol{s}^t)^T \boldsymbol{s}^t}. \tag{37}$$

The solution to (32) is given by $\boldsymbol{\theta}^{t+1} = \eta^S(\boldsymbol{u}^t, \frac{2\boldsymbol{\lambda}}{\alpha_t})$, where

$$\eta^S(\boldsymbol{u}^t, \boldsymbol{a}) = \text{sign}(\boldsymbol{u}^t)\max\left\{|\boldsymbol{u}^t| - \boldsymbol{a}, \mathbf{0}\right\} \tag{38}$$

is the soft thresholding function with $\text{sign}(\boldsymbol{u}^t)$ and $\max\left\{|\boldsymbol{u}^t| - \boldsymbol{a}, \mathbf{0}\right\}$ being component-wise operators.

Finally, the iterative procedure for solving (29) is given by

$$\boldsymbol{\theta}^{t+1} = \text{sign}\left(\boldsymbol{\theta}^t - \frac{1}{\alpha_t}\nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}')\right) \max\left\{\left|\boldsymbol{\theta}^t - \frac{1}{\alpha_t}\nabla\tilde{Q}(\boldsymbol{\theta}^t, \boldsymbol{\theta}')\right| - \frac{2\boldsymbol{\lambda}}{\alpha_t}, \mathbf{0}\right\}. \tag{39}$$

And the iteration stops when the following condition is met:

$$\frac{|J(\boldsymbol{\theta}^{t+1}) - J(\boldsymbol{\theta}^t)|}{|J(\boldsymbol{\theta}^t)|} \leq \epsilon, \tag{40}$$

where ϵ is a given small number.

4.2 Adaptive selection of $\boldsymbol{\lambda}$

So far, the parameters λ_i in the Laplace prior are fixed. Here, we propose to adaptively tune them based on the output of the iterative thresholding algorithm. The algorithm consists of solving a sequence of weighted ℓ_1-minimization problems. λ_i used for the next iteration are computed from the value of the current solution. A good choice of λ_i is to make them counteract the influence of the magnitude of the ℓ_1 penalty function [35]. Following this idea, we propose an iterative re-weighted thresholding algorithm. At the beginning of the maximization step, we set $\lambda_i = 1, \forall i$. Then, we run the iterative thresholding algorithm to obtain $\boldsymbol{\theta}$. Next, we update λ_i as $\lambda_i = \frac{1}{|\theta_i| + \epsilon}, \forall i$, where ϵ is a small positive number, and run the iterative thresholding algorithm again using the new $\boldsymbol{\lambda}$. The above process is repeated until it converges at the point where

the maximization step completes. Note that for the iterative re-weighted thresholding algorithm, the assumption that θ has a Laplacian prior no longer holds.

5 Application to gene regulatory network inference

The gene regulatory network can be described by a graph in which genes are viewed as nodes and edges depict causal relations between genes. By analyzing collected gene expression levels over a period of time, one can find some regulatory relations between different genes. Under the discrete-time state-space modeling, for a gene regulatory network with n genes, the state vector $\boldsymbol{x}_k = [x_{1,k}, \ldots, x_{n,k}]^T$, where $x_{i,k}$, denotes the gene expression level of the ith gene at time k.

In this case, the nonlinear function $\boldsymbol{f}(\boldsymbol{x})$ in the general dynamic Equation (1) is given by [15]

$$\boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) = \boldsymbol{A}\boldsymbol{g}(\boldsymbol{x}_{k-1}), \tag{41}$$

with

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix} g(x_1) \\ \vdots \\ g(x_n) \end{bmatrix}, \tag{42}$$

and

$$g_i(x) = \frac{1}{1+e^{-x}}, \qquad \forall i = 1, \cdots, n. \tag{43}$$

In (41), $\boldsymbol{A}$ is an $n \times n$ regulatory coefficient matrix with the element a_{ij} denoting the regulation coefficient from gene j to gene i. A positive coefficient a_{ij} indicates that gene j activates gene i, and a negative θ_{ij} indicates that gene j represses gene i. The parameter to be estimated is $\boldsymbol{\theta} = \boldsymbol{A}$ which is sparse.

For the measurement model, we have

$$\boldsymbol{y}_k = \boldsymbol{x}_k + \boldsymbol{v}_k. \tag{44}$$

5.1 Inference of gene regulatory network with four genes

In the simulations, we consider a network with four genes. The true gene regulatory coefficients matrix is given by

$$\boldsymbol{A} = \begin{bmatrix} 3 & 0 & 0 & -4.5 \\ -2.9 & 0 & 5 & 0 \\ -6 & 4 & 0 & 0 \\ 0 & -5 & 2 & 0 \end{bmatrix}. \tag{45}$$

To compare the EM algorithm with the proposed rEM algorithm, the simulation was conduced ten times. In each time, the initial value of $\boldsymbol{A}(\boldsymbol{\theta})$ is randomly generated from a Gaussian distribution with mean 0 and variance 2. The EM, rEM, and rEM$_w$, as well as the basis pursuit de-noising dynamic filtering (BPDN-DF) method and the ℓ_1 optimization method, are tested. Here, rEM$_w$ denotes the version of the rEM algorithm with the iterative re-weighted thresholding discussed in Section 4.2.

As a performance metrics, the receiver operating characteristic (ROC) curve is frequently used. However, for this specific example, with the increasing of the false-positive rate, the true-positive rate given by rEM and EM is always high (close to 1) which makes the distinguishment of the performance of rEM algorithm and EM algorithm difficult. Hence, the root mean-squared error (RMSE) and the sparsity factor (SF) are used in this section. The RMSE is defined by

$$\mathrm{RMSE} = \sqrt{\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\mathrm{A_{ij}} - \bar{\mathrm{A}}_{\mathrm{ij}})^2}, \tag{46}$$

where $\bar{\boldsymbol{A}}$ denotes the estimated $\boldsymbol{A}$. The SF is given by

$$\mathrm{SF} = \frac{\phi_0}{\phi}, \tag{47}$$

where ϕ_0 and ϕ are the number of zero values of the estimated parameter and the number of zero values of true parameter, respectively. It can be seen that the estimation is over sparse if the sparsity factor is greater than 1.

In addition, to test the effectiveness of the proposed method at finding the support of the unknown parameters, the number of matched elements is used and can be obtained by the following procedures: (1) Compute the support of $\boldsymbol{A}$ using the true parameters (denoted by $\boldsymbol{A}_s$) and the support of $\boldsymbol{A}$ using the estimated parameters (denoted by $\bar{\boldsymbol{A}}_s$). Note that we assign $[\boldsymbol{A}_s]_{ij} = 1$ if $A_{ij} \neq 0$ and $[\boldsymbol{A}_s]_{ij} = 0$ if $A_{ij} = 0$. Similarly, we assign $[\bar{\boldsymbol{A}}_s]_{ij} = 1$ if $\bar{A}_{ij} \neq 0$ and $[\bar{\boldsymbol{A}}_s]_{ij} = 0$ if $\bar{A}_{ij} = 0$. (2) Compute the number of zero elements of $\boldsymbol{A}_s - \bar{\boldsymbol{A}}_s$ as the matched elements. It is easy to see that it is effective at finding the support of the unknown parameters when the number of matched elements is large.

5.1.1 The effect of different λ

The performance of rEM using different λ $(10, 5, 1, 0.5, 0.1)$ is compared with the EM algorithm and the rEM$_w$. The RMSE and SF are shown in Figures 1 and 2, respectively. The RMSE does not increase monotonously with the decreasing of parameter λ. It can be seen that the rEM with $\lambda = 5$ has better performance than that using other λ. In addition, the rEM with all λ except $\lambda = 10$ outperforms the EM algorithm. It provides smaller RMSE and sparser result. The rEM$_w$ provides the smallest RMSE and sparsest parameter estimation. The number of matched elements of test algorithms with different λ is given in Figure 3. It can also be seen that rEM$_w$ provides more matched elements than the EM algorithm.

5.1.2 The effect of noise

Two different cases are tested. In the first case, the covariance of the process noise and measurement noise are chosen to be 0.01. In the second case, they are chosen to be 0.1. The performance of two test cases is shown in

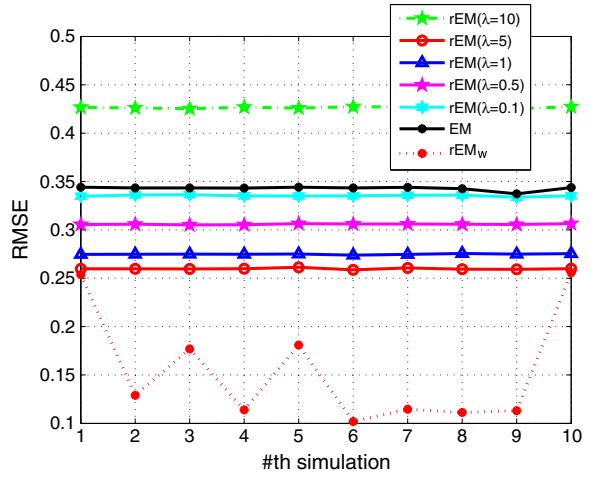

Figure 1 RMSE of rEM with different λ and rEM$_w$.

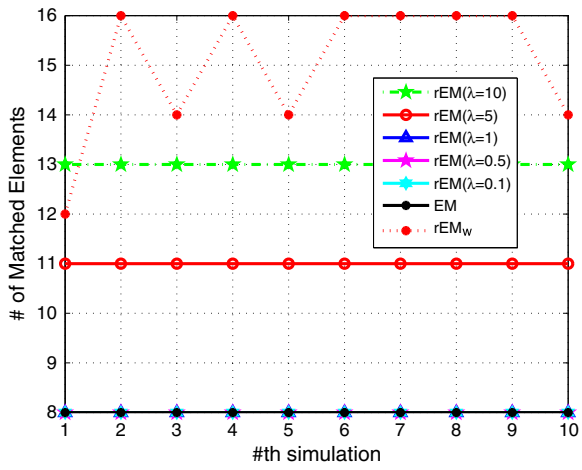

Figure 3 The number of matched elements of rEM with different λ and rEM$_w$.

Figures 4, 5, 6. It can be seen that the RMSE of rEM$_w$ with $\boldsymbol{U}, \boldsymbol{R} = 0.01\boldsymbol{I}$ is smaller than that with $\boldsymbol{U}, \boldsymbol{R} = 0.1\boldsymbol{I}$. In addition, the rEM$_w$ with $\boldsymbol{U}, \boldsymbol{R} = 0.01$ provides a larger number of matched elements than that with $\boldsymbol{U}, \boldsymbol{R} = 0.1$ as shown in Figure 6. Hence, the estimation accuracy is better when the process noise and measure noise are small.

5.1.3 *The effect of the number of observations*

In order to test the effect of the number of observations, the rEM$_w$ algorithm with 10 and 20 observations are tested. The simulation results are shown in Figures 7, 8, 9. It can be seen that rEM$_w$ with more observations gives less RMSE. In addition, as shown in Figure 9, rEM$_w$ with more observations gives slightly better result for finding the support of the unknown parameters.

5.1.4 *The effect of κ*

In order to test the performance of κ, the rEM$_w$ algorithm with different κ (0,-1,-3) is tested. The performance results are shown in Figures 10, 11, 12. Note that the cubature rule corresponds to $\kappa = 0$, and the unscented transformation corresponds to $\kappa = -1$. Roughly speaking, the performance of rEM$_w$ with different κ is close. Specifically, it can be seen that the RMSE of rEM$_w$ using $\kappa = -1$ and rEM$_w$ using $\kappa = -3$ is less than that of rEM$_w$ using $\kappa = 0$. The sparsity factor of rEM$_w$ using $\kappa = -1$ is more close to 1 than that of rEM$_w$ using $\kappa = -3$ and $\kappa = 0$. Moreover, the number of matched elements of rEM$_w$ using $\kappa = -1$ is more than that of rEM$_w$ using $\kappa = -3$ and $\kappa = 0$. Hence, the performance of rEM using $\kappa = -1$ is the best in this case.

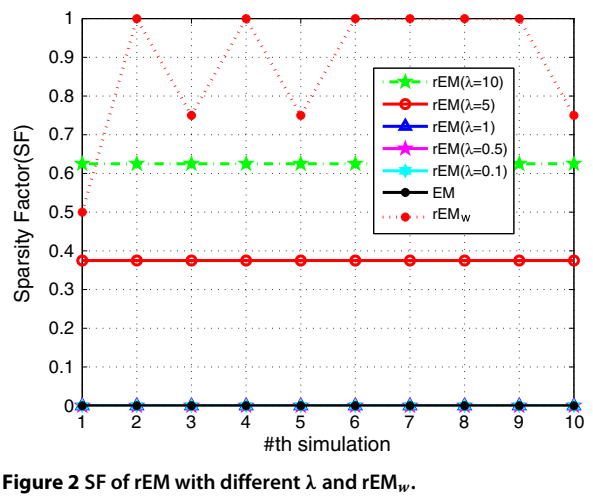

Figure 2 SF of rEM with different λ and rEM$_w$.

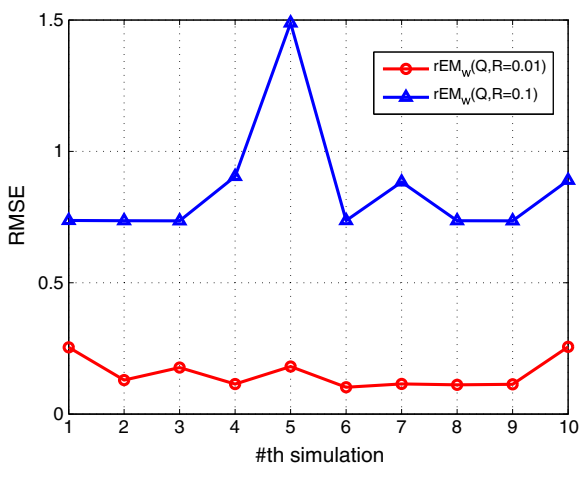

Figure 4 RMSE of rEM$_w$ with different noise levels.

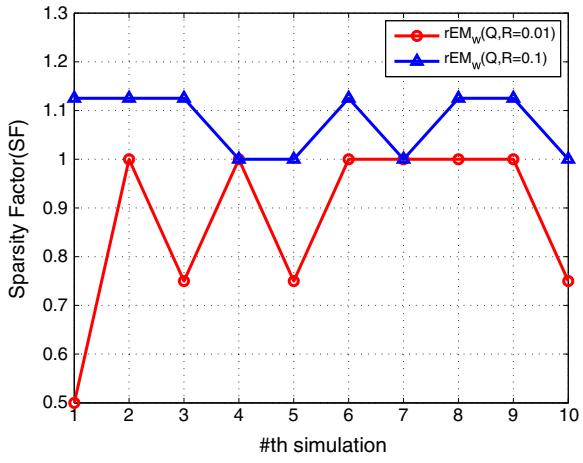

Figure 5 SF of rEM$_w$ with different noise levels.

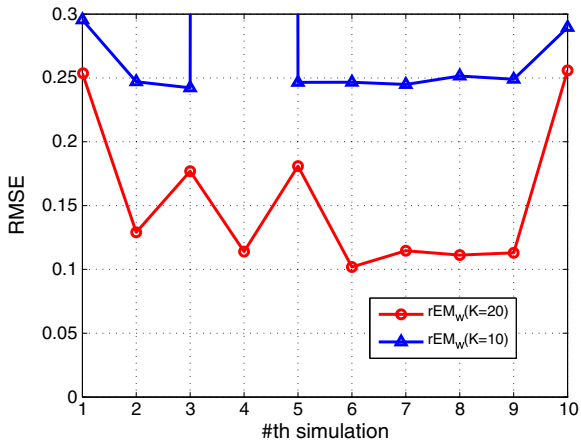

Figure 7 RMSE of rEM$_w$ with different lengths of observations.

5.1.5 *Effect of sparsity level*

The performance comparison of the rEM$_w$ and the conventional EM with different sparsity levels of $\boldsymbol{A}$ is shown in Figures 13, 14, 15. In this subsection, another $\boldsymbol{A}$ which is denser than the previously used $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = \begin{bmatrix} 3 & -1 & 0 & -4.5 \\ -2.9 & 0 & 5 & 1 \\ -6 & 4 & 0 & -1 \\ 1 & -5 & 2 & 0 \end{bmatrix}. \tag{48}$$

Note that '(Denser)' is used to denote the result using $\boldsymbol{A}$ shown in Equation 48. It can be seen that the RMSE of rEM$_w$(Denser) is comparable to that of the EM(Denser). However, the sparsity factor of rEM$_w$(Denser) is closer to 1 than that of the EM(Denser) which means that the rEM$_w$(Denser) is better. In addition, the number of matched elements of the rEM$_w$(Denser) is large than that of the EM(Denser), which means that the rEM$_w$(Denser) is better than the EM(Denser) in finding the support of the unknown parameters. The performance of the rEM$_w$(Denser), however, is worse than that of the rEM$_w$ in terms of the improvement of the RMSE. Hence, the rEM algorithm may have close performance with the EM algorithm when the sparsity is not obvious.

5.1.6 *Comparison with ℓ_1 optimization*

We compare the proposed rEM algorithm and the ℓ_1 optimization-based method, as well as the conventional EM algorithm. The ℓ_1 optimization is a popular approach to obtain the sparse solution. For the problem under consideration, it obtains an estimate of $\boldsymbol{\theta}$ by solving the following optimization problem:

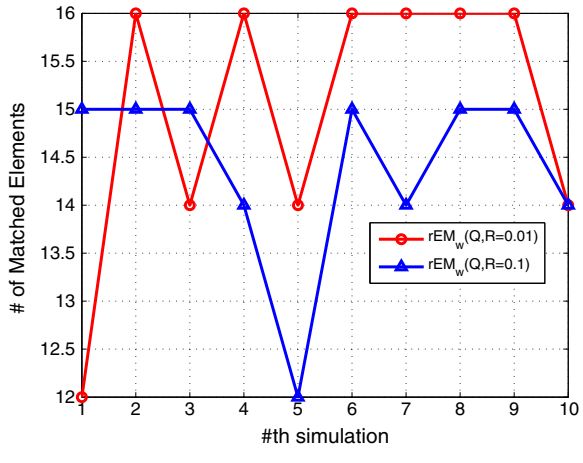

Figure 6 The number of matched elements of rEM$_w$ with different noise levels.

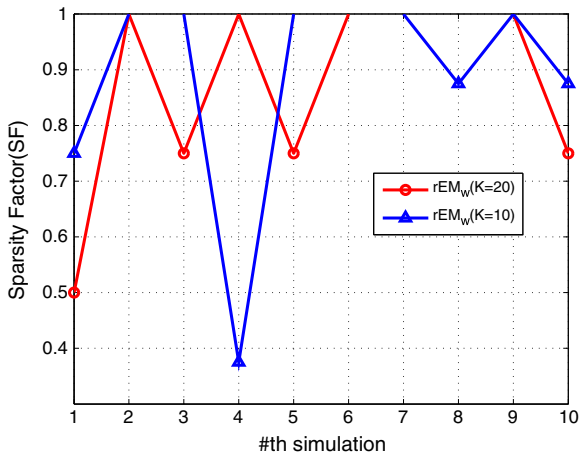

Figure 8 SF of rEM$_w$ with different lengths of observations.

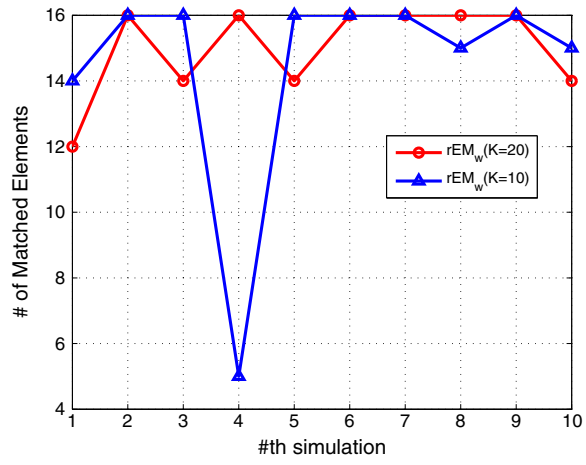

Figure 9 The number of matched elements of rEM$_w$ with different lengths of observations.

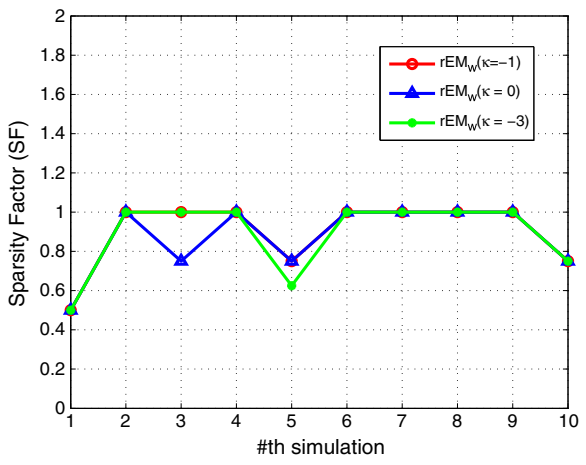

Figure 11 SR of rEM$_w$ with different κ.

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{k=2}^{K} [\boldsymbol{y}_k - \boldsymbol{A}(\boldsymbol{\theta})\boldsymbol{g}(\hat{\boldsymbol{x}}_{k-1})]^T [\boldsymbol{y}_k - \boldsymbol{A}(\boldsymbol{\theta})\boldsymbol{g}(\hat{\boldsymbol{x}}_{k-1})] + \lambda \|\boldsymbol{\theta}\|_1, \tag{49}$$

where $\hat{\boldsymbol{x}}_1 = \boldsymbol{x}_1$ and $\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{g}(\hat{\boldsymbol{x}}_k)$.

We also compare the ℓ_1 optimization method with the proposed rEM$_w$ algorithm, and the results are shown in Figures 16, 17, 18. Seven different λ (5, 2, 1, 0.5, 0.1, 0.05, and 0.01) are used in the ℓ_1 optimization method. The RMSE does not decrease monotonously with the decreasing of the parameter λ. Among all tested values, the ℓ_1 optimization method with $\lambda = 0.1$ gives the smallest RMSE. However, the sparsity factor of the ℓ_1 optimization with $\lambda = 0.1$ is far from the ideal value 1. The ℓ_1 optimization with $\lambda = 5$ gives the best support detection as shown in Figure 18. The re-weighted ℓ_1 optimization algorithm is also used in the simulation. However, all ℓ_1 optimization-based methods cannot achieve better performance than that of using the rEM$_w$.

5.1.7 Comparison with BPDN-DF

To solve the problem using BPDN-DF, the model in (41) and (44) are modified as

$$\tilde{\boldsymbol{x}}_k = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{k-1}) \begin{bmatrix} \boldsymbol{A}(\boldsymbol{\theta}_{k-1})\boldsymbol{g}(\boldsymbol{x}_{k-1}) \\ \boldsymbol{\theta}_{k-1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{v}_k \\ \boldsymbol{0} \end{bmatrix} \tag{50}$$

and

$$\boldsymbol{h}(\tilde{\boldsymbol{x}}_k) = \tilde{\boldsymbol{H}}_k + \boldsymbol{n}_k = \begin{bmatrix} \boldsymbol{I}_4 & \boldsymbol{0}_{16} \end{bmatrix} \tilde{\boldsymbol{x}}_k + \boldsymbol{n}_k, \tag{51}$$

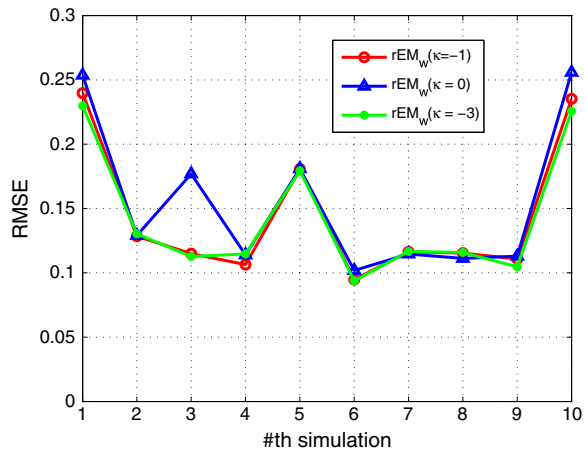

Figure 10 RMSE of rEM$_w$ with different κ.

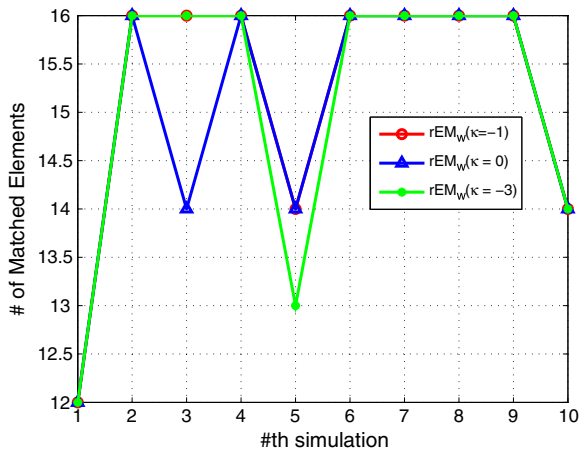

Figure 12 The number of matched elements of rEM$_w$ with different κ.

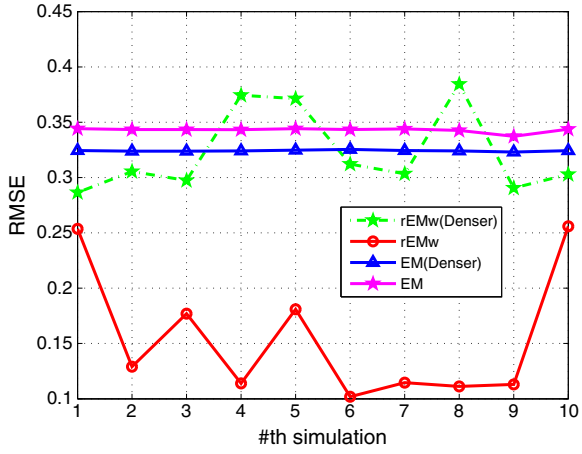

Figure 13 RMSE of the EM and rEM$_w$ for normal and denser A.

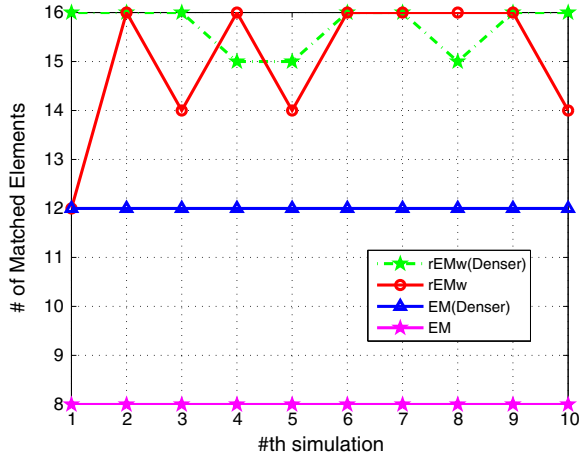

Figure 15 The number of matched elements of the EM and rEM$_w$ for normal and denser A.

respectively. Note that $\tilde{\boldsymbol{x}}_k = \left[\boldsymbol{x}_k^T, \boldsymbol{\theta}_k^T\right]^T$. Then, $\hat{\tilde{\boldsymbol{x}}}_k$ is given by [36]

$$\hat{\tilde{\boldsymbol{x}}}_k = \arg\min_{\tilde{\boldsymbol{x}}} \left[\|\boldsymbol{y}_k - \tilde{H}_k\tilde{\boldsymbol{x}}\|_2^2 + \boldsymbol{\lambda}\|\tilde{\boldsymbol{x}}\|_1 + \|\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{k-1})\|_2^2\right], \tag{52}$$

where $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_{20}]$ with $\lambda_i = 0$, $i = 1, 2, 3, 4$ since our objective is to explore the sparsity of the parameter $\boldsymbol{\theta}$. The exact same initial values used in testing EM and rEM are used to test the performance of the BPDN-DF. The simulation results are shown in Figures 19, 20, 21. It can be seen that although the sparsity factor of BPDN-DF is comparable with that of the rEM$_w$, the RMSE of the BPDN-DF is much larger than that of the rEM$_w$. In addition, as shown in Figure 21, the rEM$_w$ is better than the BPDN-DF in finding the support of the unknown parameters. The possible reason is that the BPDN-DF does not consider the noise in the dynamic system, and the measurement matrix $\tilde{H}_k$ is an ill-conditioned matrix. In the simulation, $\lambda_j = 0.1, j = 5, \cdots, 20$. Based on our tests by using other values of λ, there is no obvious improvement.

5.2 Inference of gene regulatory network with eight genes

In this section, we test the proposed algorithm using a larger gene regulatory network which includes eight genes; the performances of the EM, the rEM, the rEM$_w$, the ℓ_1 optimization method, and BPDN-DF are given. Forty data points are collected to infer the structure of the network. The system noise and measurement noise are assumed to be Gaussian-distributed with means $\mathbf{0}$ and covariances $\boldsymbol{U}_k = 0.01\boldsymbol{I}_8$ and $\boldsymbol{R}_k = 0.01\boldsymbol{I}_8$, respectively.

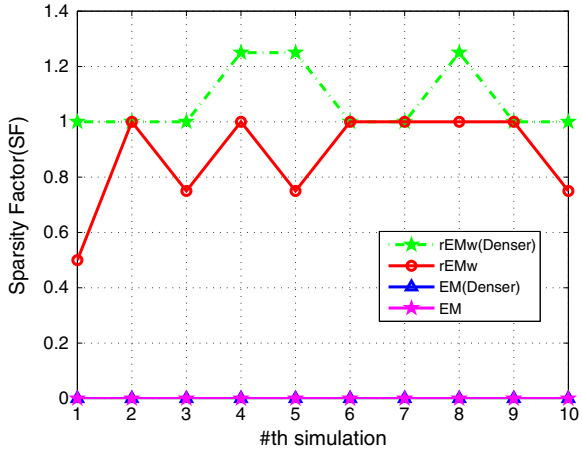

Figure 14 SF of the EM and rEM$_w$ for normal and denser A.

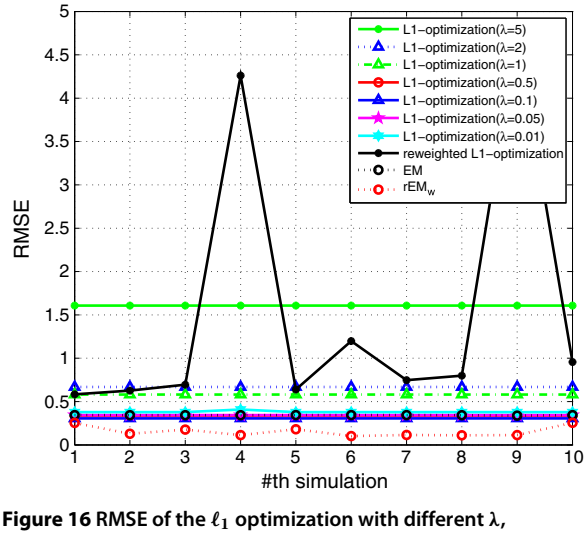

Figure 16 RMSE of the ℓ_1 optimization with different λ, reweighted ℓ_1 optimization, EM, and rEM$_w$.

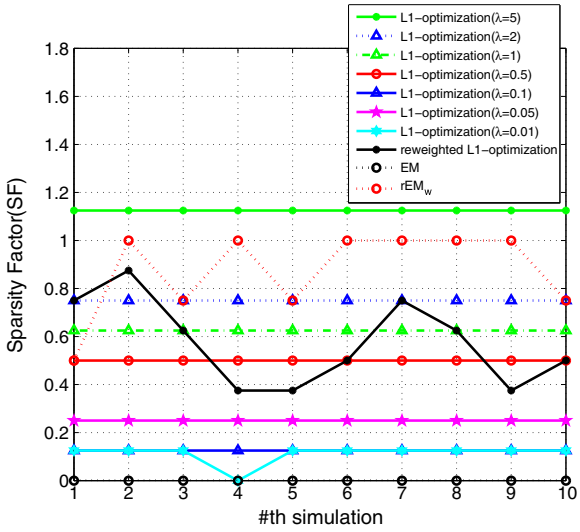

Figure 17 SF of the ℓ_1 optimization with different λ, reweighted ℓ_1 optimization, EM, and rEM$_w$.

The connection coefficient matrix is given by

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 2.4 & 3.2 \\ 0 & 0 & 0 & 4.1 & 0 & -2.4 & 0 & 4.1 \\ -5.0 & 2.1 & -1.5 & 0 & 4.5 & 0 & 2.1 & 0 \\ 0 & 1.3 & 2.5 & -3.7 & 1.8 & 0 & 0 & -3.1 \\ 0 & 0 & 0 & -2.6 & -3.2 & 0 & -1 & 4 \\ -1.5 & -1.8 & 0 & 3.4 & 1.4 & 1.1 & 0 & 1.7 \\ -1.8 & 0 & 0 & -3 & 1.1 & 2.4 & 0 & 0 \\ -1.3 & 0 & -1 & 0 & 2.1 & 0 & 0 & 2.2 \end{pmatrix}. \quad (53)$$

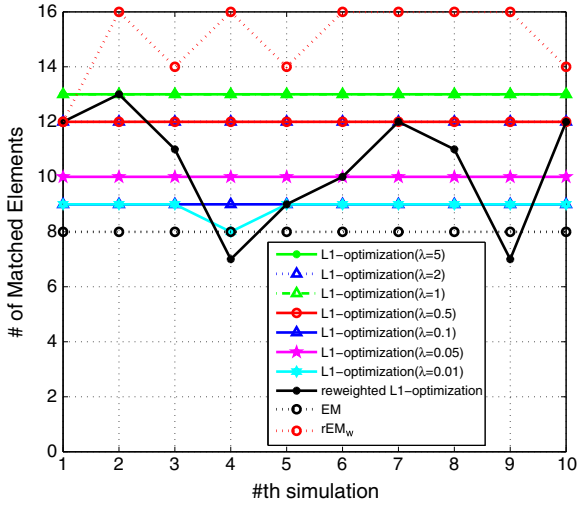

Figure 18 The number of matched elements of the ℓ_1 optimization with different λ, reweighted ℓ_1 optimization, EM, and rEM$_w$.

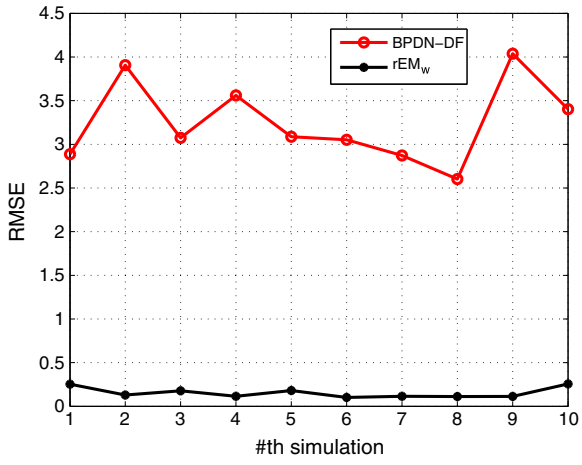

Figure 19 RMSE of BPDN-DF and rEM$_w$.

For testing, each coefficient in $\hat{A}$ is initialized from a Gaussian distribution with mean 0 and variance 1. The system state is initialized using the first measurement.

The metric used to evaluate the inferred GRN is the ROC curve, in which the true-positive rate (TPR) and the false-positive rate (FPR) are involved. They are given by

$$\mathrm{TPR} = \frac{\mathrm{TP\#}}{\mathrm{TP\#} + \mathrm{FN\#}}, \quad (54)$$

$$\mathrm{FPR} = \frac{\mathrm{FP\#}}{\mathrm{FP\#} + \mathrm{TN\#}}, \quad (55)$$

where the number of true positives (TP#) denotes the number of links correctly predicted by the inference algorithm, the number of false positives (FP#) denotes the number of incorrectly predicted links. The number of true negatives (TN#) denotes the number of correctly predicted non-links, and the number of false negatives

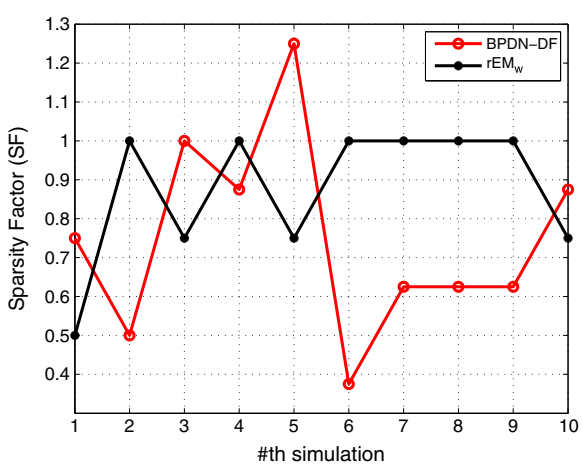

Figure 20 SF of BPDN-DF and rEM$_w$.

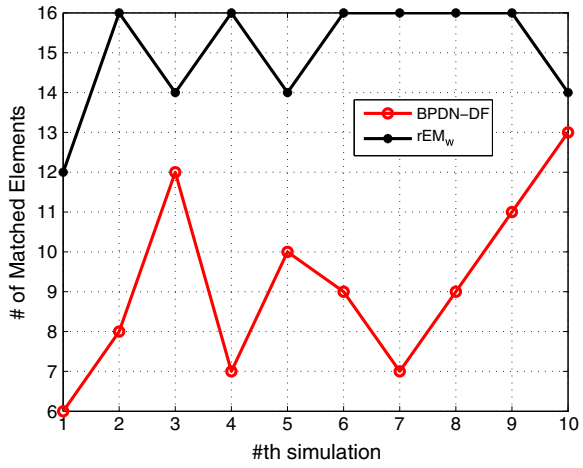

Figure 21 The number of matched elements of BPDN-DF and rEM$_w$.

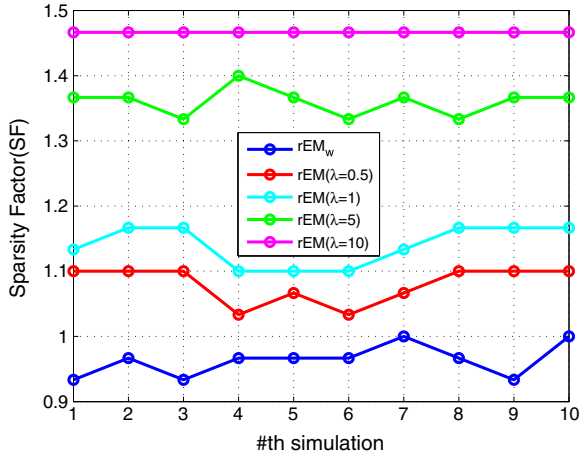

Figure 23 Sparsity factor of the rEM and the rEM$_w$.

(FN#) denotes the number of missed links by the inference algorithm [15].

The ROC curves of the EM, the rEM, and the rEM$_w$ are compared in Figure 22. The rEM with different λ is tested. In Figure 22, the curves of rEM with four typical values of λ are shown. There is no obvious improvement by using other λ. From the figure, it can be seen that the rEM$_w$ performs better than the rEM and the convectional EM algorithms.

In addition, the sparse solution is obtained by using rEM and rEM$_w$ while it cannot be obtained by using the EM algorithm. The sparsity factor of rEM and rEM$_w$ is shown in Figure 23; the sparsity of the solution given by rEM$_w$ is closer to the ground truth than that given by the EM algorithm.

In Figure 24, the ROC curves of the rEM$_w$, ℓ_1 optimization method, and BPDN-DF are compared. Similarly, the ℓ_1 optimization method with different λ is tested, and only four curves are shown in the figure. By using other values, there is no obvious improvement. The BPDN-DF with different λ has no obvious difference in the test. From Figure 24, it can be seen that the rEM$_w$ performs much better than the ℓ_1 optimization method and BPDN-DF algorithm. Hence, the sparsity factor of ℓ_1 optimization method and BPDN-DF is not shown.

5.3 Inference of gene regulatory network from malaria expression data

The dataset with the first six gene expression data of malaria is given in reference [37] and is used in this section. The initial covariance for the algorithm is $\boldsymbol{P}_0 = 0.5\boldsymbol{I}$. The process noise and measurement noise are

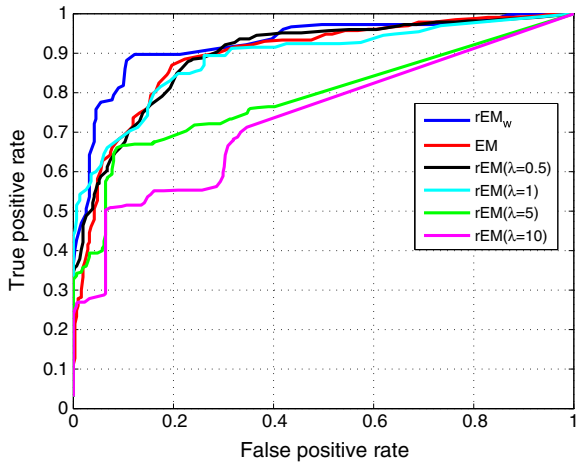

Figure 22 ROCs of the EM, the rEM, and the rEM$_w$.

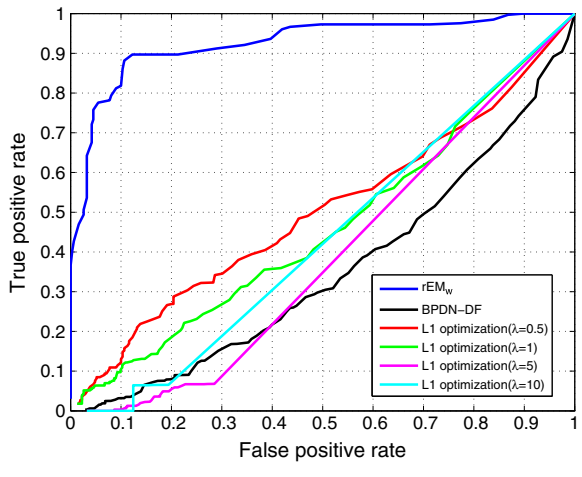

Figure 24 ROCs of the rEM$_w$, ℓ_1 optimization method, and BPDN-DF.

assumed to be Gaussian noise with zero mean and covariance $0.3^2\boldsymbol{I}$ and $0.4^2\boldsymbol{I}$, respectively. In the following, we show the inference results of the parameter and the state estimation provided by the unscented Kalman filter (UKF) based on the model using the inferred parameters.

The inferred $\boldsymbol{A}$ by the EM algorithm is

$$\bar{\boldsymbol{A}} = \begin{bmatrix} 2.2120 & -7.9443 & 2.3843 & 6.1800 & -3.5269 & 2.8300 \\ -0.6585 & -0.5319 & 0.5987 & 4.0023 & -2.8684 & 1.1167 \\ 1.9022 & -9.1935 & 3.0504 & 7.9274 & -5.0037 & 3.4825 \\ 1.8157 & -8.8003 & 3.4441 & 9.4813 & -7.1284 & 3.4345 \\ 1.8413 & -8.3515 & 2.2789 & 5.3726 & -1.6722 & 2.5999 \\ 2.1053 & -3.3850 & 3.4007 & 10.2753 & -12.3170 & 2.1100 \end{bmatrix}. \tag{56}$$

The inferred $\boldsymbol{A}$ by the rEM with $\lambda = 1$ is

$$\bar{\boldsymbol{A}} = \begin{bmatrix} 0.8448 & -6.3169 & 0.8943 & 3.9423 & 0 & 2.8387 \\ 0 & -0.2422 & 0 & 0.5051 & 0 & 1.3407 \\ 0.1424 & -7.1799 & 0.7461 & 5.2535 & 0.3607 & 2.9298 \\ 0.0048 & -8.1010 & 0.7851 & 5.3300 & 0 & 4.2157 \\ 0.4022 & -6.9358 & 2.0375 & 4.0332 & 0 & 2.7372 \\ 0 & -5.6613 & 0 & 4.3934 & -2.9426 & 6.3350 \end{bmatrix}. \tag{57}$$

The inferred $\boldsymbol{A}$ by the rEM$_w$ is

$$\bar{\boldsymbol{A}} = \begin{bmatrix} 0.3662 & -7.5033 & 0 & 9.6020 & 0 & 0 \\ -2.0531 & -1.1905 & 0 & 5.1439 & -0.0011 & 0 \\ 0 & -9.0526 & 0 & 11.6504 & 0 & 0 \\ 0 & -9.3419 & 0 & 14.4056 & -3.5361 & 1.0739 \\ 0.0034 & -8.5250 & 0 & 11.0732 & 0 & 0 \\ 0 & -3.8773 & 0.0025 & 13.1848 & -8.3610 & 1.4877 \end{bmatrix}. \tag{58}$$

The state estimation provided by the UKF based on the model using the inferred parameters of the EM, the rEM, the rEM$_w$, and the true gene expression is shown in Figure 25. The left top and right top panels are the expression of the first gene and the second gene, respectively. The left center and right center panels are the expression of the third gene and the fourth gene, respectively. The left bottom and right bottom panels are the expression of the fifth gene and the sixth gene, respectively. It can be seen that the estimate gene expression using the UKF and parameters given by the EM, the rEM, and the rEM$_w$ is close to the true gene expression data. In addition, The rEM$_w$ algorithm provide sparser

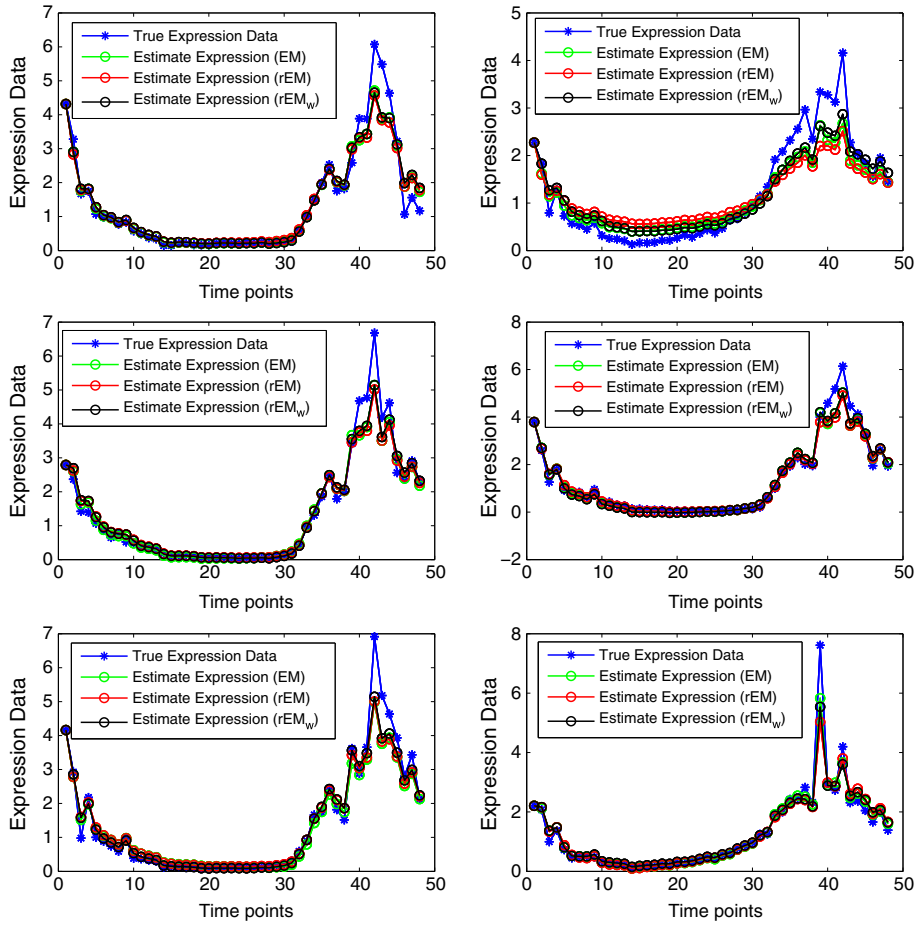

Figure 25 **True malaria gene expression and estimated gene expression by different algorithms.**

solution than the rEM algorithm. Both the rEM and the rEM_w algorithms give sparser solutions than the EM algorithm which validates the effectiveness of the proposed method.

6 Conclusions

In this paper, we have considered the problem of sparse parameter estimation in a general nonlinear dynamic system, and we have proposed an approximate MAP-EM solution, called the rEM algorithm. The expectation step involves the forward Gaussian approximation filtering and the backward Gaussian approximation smoothing. The maximization step employs a re-weighted iterative thresholding method. We have provided examples of the inference of gene regulatory network based on expression data. Comparisons with the traditional EM algorithm as well as with the existing approach to solving sparse problems such as the ℓ_1 optimization and the BPDN-DF show that the proposed rEM algorithm provides both more accurate estimation result and sparser solutions.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Intelligent Fusion Technology, Germantown, Inc., MD 20876, USA. [2]Department of Electrical Engineering, Columbia University, New York, NY 10027, USA.

Received: 26 December 2013 Accepted: 26 February 2014
Published: 3 April 2014

References

1. J Tropp, S Wright, Computational methods for sparse solution of linear inverse problems. Proc. IEEE. **98**(6), 948–958 (2010)
2. S Ji, Y Xue, L Carin, Bayesian compressive sensing. IEEE Trans. Signal Processing. **56**(6), 2346–2356 (2008)
3. EG Larsson, Y Selen, Linear regression with a sparse parameter vector. IEEE Trans. Signal Processing. **55**(2), 451–460 (2007)
4. M Figueiredo, R Nowak, S Wright, Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. IEEE J. Sel. Top. Sign. Process. **1**(4), 586–597 (2007)
5. D Zachariah, S Chatterjee, M Jansson, Dynamic iterative pursuit. IEEE Trans. Signal Processing. **60**(9), 4967–4972 (2012)
6. C Qiu, W Lu, N Vaswani, Real-time dynamic MR image reconstruction using Kalman filtered compressed sensing. Paper presented in the IEEE international conference on acoustics, speech and signal processing (ICASSP), Taipei, 19–24 April 2009, pp. 393–396
7. J Ziniel, P Schniter, Efficient high-dimensional inference in the multiple measurement vector problem. IEEE Trans. Signal Processing. **61**(2), 340–354 (2013)
8. J Vila, P Schniter, Expectation-maximization Bernoulli-Gaussian approximate message passing. Paper presented at the forty-fifth Asilomar conference on signals, systems and computers (ASILOMAR), Pacific Grove, CA USA, 6–9 Nov 2011, pp. 799–803
9. J Vila, P Schniter, Expectation-maximization Gaussian-mixture approximate message passing. Paper presented in the 46th annual conference on information sciences and systems (CISS), Princeton, NJ USA, 21–23 March 2012, pp. 1–6
10. U Kamilov, S Rangan, A Fletcher, M Unser, Estimation, Approximate Message Passing with Consistent Parameter and Applications to Sparse Learning. Paper presented at the 26th annual conference on neural information processing systems, Lake Tahoe, NV, USA, 3–8 Dec 2012
11. A Gurbuz, M Pilanci, O Arikan, Expectation maximization based matching pursuit. Paper presented at the in IEEE international conference on acoustics, speech and signal processing (ICASSP), Kyoto, 25–30 March 2012, pp. 3313–3316
12. A Charles, C Rozell, *Re-weighted l_1 Dynamic Filtering for Time-Varying Sparse Signal Estimation* (Cornell University, Ithaca, 2012). arXiv:1208.0325
13. J Ziniel, P Schniter, Efficient high-dimensional inference in the multiple measurement vector problem. IEEE Trans. Signal Processing. **61**(2), 340–354 (2013)
14. S Barembruch, E Moulines, A Scaglione, A sparse EM algorithm for blind and semi-blind identification of doubly selective OFDM channels. Paper presented at the IEEE eleventh international workshop on signal processing advances in wireless communications (SPAWC), Marrakech, 20–23 June 2010, pp. 1–5
15. A Noor, E Serpedin, M Nounou, H Nounou, Inferring gene regulatory networks via nonlinear state-space models and exploiting sparsity. IEEE/ACM Trans. Comput. Biol. Bioinform. **9**(4), 1203–1211 (2012)
16. TS Gardner, D Di Bernardo, D Lorenz, JJ Collins, Inferring genetic networks and identifying compound mode of action via expression profiling. Science **301**(5629), 102–105 (2003)
17. J Tegner, MS Yeung, J Hasty, JJ Collins, Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proc. Natl. Acad. Sci. **100**(10), 5944–5949 (2003)
18. X Cai, JA Bazerque, GB Giannakis, Inference of gene regulatory networks with sparse structural equation models exploiting genetic perturbations. PLoS Comput. Biol. **9**(5), e1003068 (2013)
19. D Thieffry, AM Huerta, E Pérez-Rueda, J Collado-Vides, From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in *Escherichia coli*. Bioessays. **20**(5), 433–440 (1998)
20. Z Wang, F Yang, D Ho, S Swift, A Tucker, X Liu, Stochastic dynamic modeling of short gene expression time-series data. IEEE Trans. Nanobioscience. **7**(1), 44–55 (2008)
21. A Noor, E Serpedin, M Nounou, H Nounou, Reverse engineering sparse gene regulatory networks using cubature Kalman filter and compressed sensing. Adv. Bioinformatics. **2013**, 205763 (2013)
22. L Wang, X Wang, AP Arkin, MS Samoilov, Inference of gene regulatory networks from genome-wide knockout fitness data. Bioinformatics. **29**(3), 338–346 (2013)
23. G McLachlan, T Krishnan, *The EM Algorithm and Extensions* (Wiley-Interscience, Hoboken, 2008)
24. I Arasaratnam, S Haykin, Cubature Kalman filters. IEEE Trans. Automat. Contr. **54**(6), 1254–1269 (2009)
25. K Ito, K Xiong, Gaussian filters for nonlinear filtering problems. IEEE Trans. Automat. Contr. **45**(5), 910–927 (2000)
26. B Jia, M Xin, Y Cheng, Sparse-grid quadrature nonlinear filtering. Automatica. **48**(2), 327–341 (2012)
27. SJ Julier, JK Uhlmann, Unscented filtering and nonlinear estimation. Proc. IEEE. **92**(3), 401–422 (2004)
28. D Guo, X Wang, Quasi-Monte Carlo filtering in nonlinear dynamic systems. IEEE Trans. Signal Processing. **54**(6), 2087–2098 (2006)
29. B Jia, M Xin, Y Cheng, High-degree cubature Kalman filter. Automatica. **49**(2), 510–518 (2013)
30. S Sarkka, Unscented Rauch–Tung–Striebel smoother. IEEE Trans. Automat. Cont. **53**(3), 845–849 (2008)
31. M Schmidt, Graphical model structure learning with L1-regularization. Ph.D. Dissertation, University of British Columbia, 2010
32. S Bahmani, B Raj, P Boufounos, Greedy sparsity-constrained optimization. J. Mac. Learn. Res. **14**, 807–841 (2013)
33. A Beck, YC Eldar, Sparsity constrained nonlinear optimization: optimality conditions and algorithms. SIAM J. Optim. **23**(3), 1480–1509 (2013)
34. S Wright, R Nowak, M Figueiredo, Sparse reconstruction by separable approximation. IEEE Trans. Signal Processing. **57**(7), 2479–2493 (2009)
35. EJ Candes, MB Wakin, S Boyd, Enhancing sparsity by reweighted l1 minimization. J. Fourier. Anal. Appl. **14**, 877–905 (2008)

36. A Charles, MS Asif, J Romberg, C Rozell, Sparsity penalties in dynamical system estimation. Paper presented at the 45th annual conference on information sciences and systems (CISS), Baltimore, 23–25 March 2011, pp. 1–6
37. Z Wang, X Liu, Y Liu, J Liang, V Vinciotti, An extended kalman filtering approach to modeling nonlinear dynamic gene regulatory networks via short gene expression time series. IEEE/ACM Trans. Comput. Biol. Bioinform. **6**(3), 410–419 (2009)

doi:10.1186/1687-4153-2014-5
Cite this article as: Jia and Wang: **Regularized EM algorithm for sparse parameter estimation in nonlinear dynamic systems with application to gene regulatory network inference.** *EURASIP Journal on Bioinformatics and Systems Biology* 2014 **2014**:5.