BIOINFORMATICS ORIGINAL PAPER

Vol. 29 no. 19 2013, pages 2410–2418
doi:10.1093/bioinformatics/btt433

Gene expression

Advance Access publication August 11, 2013

ROBNCA: robust network component analysis for recovering transcription factor activities

Amina Noor[1], Aitzaz Ahmad[2], Erchin Serpedin[1,*], Mohamed Nounou[3] and Hazem Nounou[4]

[1]Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843, USA, [2]Corporate Research and Development, Qualcomm Technologies Inc., San Diego, CA 92121, USA, [3]Department of Chemical Engineering and [4]Department of Electrical Engineering, Texas A&M University at Qatar, Doha Qatar

Associate Editor: Ziv Bar-Joseph

ABSTRACT

Motivation: Network component analysis (NCA) is an efficient method of reconstructing the transcription factor activity (TFA), which makes use of the gene expression data and prior information available about transcription factor (TF)–gene regulations. Most of the contemporary algorithms either exhibit the drawback of inconsistency and poor reliability, or suffer from prohibitive computational complexity. In addition, the existing algorithms do not possess the ability to counteract the presence of outliers in the microarray data. Hence, robust and computationally efficient algorithms are needed to enable practical applications.

Results: We propose ROBust Network Component Analysis (ROBNCA), a novel iterative algorithm that explicitly models the possible outliers in the microarray data. An attractive feature of the ROBNCA algorithm is the derivation of a closed form solution for estimating the connectivity matrix, which was not available in prior contributions. The ROBNCA algorithm is compared with FastNCA and the non-iterative NCA (NI-NCA). ROBNCA estimates the TF activity profiles as well as the TF–gene control strength matrix with a much higher degree of accuracy than FastNCA and NI-NCA, irrespective of varying noise, correlation and/or amount of outliers in case of synthetic data. The ROBNCA algorithm is also tested on *Saccharomyces cerevisiae* data and *Escherichia coli* data, and it is observed to outperform the existing algorithms. The run time of the ROBNCA algorithm is comparable with that of FastNCA, and is hundreds of times faster than NI-NCA.

Availability: The ROBNCA software is available at http://people.tamu.edu/~amina/ROBNCA

Contact: serpedin@ece.tamu.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on March 26, 2013; revised on June 28, 2013; accepted on July 24, 2013

1 INTRODUCTION

Recent advances in technology have enabled monitoring of cellular activities using more sophisticated techniques, and have provided a deluge of biological data. Using these data to unravel the underlying phenomena that regulate various activities in a living organism offers the potential to reap numerous benefits. One of the key biological processes is transcriptional regulation, which controls the gene expression and amount of RNA produced. This process is regulated by transcription factors (TFs), which are specialized proteins causing the genes to express by binding onto the gene promoters. A thorough understanding of this complex transcriptional regulation and TF–gene interaction will potentially aid in predicting the biological processes and designing control strategies to cure and/or avoid the diseased conditions (Lähdesmäki *et al.*, 2008). Microarray technologies are able to measure the level of gene expressions and quantify them in the form of gene expression data. Such data are widely used in the inference of gene–gene interactions. Transcription factor activity (TFA), which is defined as the concentration of its subpopulation with DNA binding ability, controls the transcriptional regulation (Jajamovich *et al.*, 2011). The correlation between TFAs and TF expression level is modified at the post-transcriptional and post-translational stage. It is, therefore, much harder to measure TFA profiles experimentally, and scientists have resorted to computational methods for their estimation (Yang *et al.*, 2005).

Several statistical techniques including principal component analysis (PCA) (Jolliffe, 1986) and independent component analysis (ICA) (Comon, 1992) have been used to deduce useful information from sets of biological data. However, the successful application of these algorithms hinges on the assumptions of orthogonality and independence between the signals, which do not hold for biological signals in practice (Chang *et al.*, 2008). In fact, some prior information is usually available for many systems, and it should be incorporated in the system model, e.g. ChIP–chip data indicates which TFs and genes are known to interact. The gene regulatory network can be modelled linearly as follows (Liao *et al.*, 2003)

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{S} + \boldsymbol{\Gamma}\,, \tag{1}$$

where $\boldsymbol{Y}$ is the $N \times K$ gene expression data matrix, $\boldsymbol{A}$ is the $N \times M$ control strength or connectivity matrix and $\boldsymbol{S}$ is the $M \times K$ matrix denoting the TFAs. The uncertainties in the observation data are assumed to be Gaussian (Chang *et al.*, 2008; Jacklin *et al.*, 2012), and are represented by the entries of the noise matrix $\boldsymbol{\Gamma}$. Genes and TFs are known to interact in a dynamic and non-linear manner; however, a log-linear relationship provides a good approximation. Because a particular TF regulates only a few other genes, the connectivity matrix $\boldsymbol{A}$ is expected to be sparse. The problem then boils down to estimating $\boldsymbol{S}$ and

*To whom correspondence should be addressed.

$\boldsymbol{A}$, where $\boldsymbol{Y}$ is available and some a-priori information about the matrix $\boldsymbol{A}$ is known.

Network component analysis (NCA), proposed by Liao *et al.* (2003), provides a more accurate model for TF–gene regulation and makes use of the related prior information available. It was shown that provided certain conditions are met, the NCA algorithm produces a unique solution of the aforementioned estimation problem in the absence of noise. The *NCA criteria* require that: (i) the matrix $\boldsymbol{A}$ is full column-rank; (ii) if a row is removed from $\boldsymbol{S}$ as well as the output elements connected to it, the updated control strength matrix should still be of full column-rank; (iii) the TFA matrix $\boldsymbol{S}$ should have a full row-rank. These criteria guarantee that the solution obtained is unique up to a scale ambiguity (Jacklin *et al.*, 2012; Liao *et al.*, 2003). When the NCA criteria are satisfied, the optimization problem reduces to:

$$\min_{\mathbf{A},\mathbf{S}} \|\mathbf{Y}-\mathbf{AS}\|_F^2 \qquad \text{s.t. } \mathbf{A}(I)=0\ , \tag{2}$$

where $\|.\|_F$ denotes the Frobenius norm and I is the set of all indices where the entries of matrix $\boldsymbol{A}$ are known to be zero. The algorithm in (Galbraith *et al.*, 2006) allows the recovery of source signals when the microarray data consist of fewer data points and (Tran *et al.*, 2005) formulates the incorporation of regulatory knockout constraints as well.

The NCA problem in (2) was first solved by using alternate least squares (ALS) for both $\boldsymbol{A}$ and $\boldsymbol{S}$ (Liao *et al.*, 2003). However, because the ALS solution requires solving a high dimensional matrix optimization problem at each iteration, it entails prohibitive computational complexity for large datasets, which often need to be handled in gene networks. FastNCA provides a closed form solution for $\boldsymbol{A}$, which uses singular value decomposition (SVD) (Chang *et al.*, 2008), and is several tens of times faster than the ALS algorithm. The authors in (Jacklin *et al.*, 2012) propose a non-iterative version of NCA, herein referred to as NI-NCA, which offers greater consistency in terms of TFA estimation at the cost of much higher computational complexity than FastNCA. However, because the decomposition techniques used to derive these algorithms are susceptible even to the presence of small amount of outliers (Mateos and Giannakis, 2012), their performance is expected to deteriorate significantly when data points are corrupted by outliers. It is commonly known that the microarray data are noisy and are corrupted with outliers because of erroneous measurements and/or abnormal response of genes, and robust algorithms are required for gene network inference (Finegold and Drton, 2011). Therefore, it is imperative to develop an NCA algorithm that has an inherent ability to mitigate the effect of outliers, and also entails low computational costs and provides good consistency and accuracy. It is precisely this avenue which is the focus of our current work. The main contributions of this article can be summarized as follows:

(1) A novel algorithm, ROBust Network Component Analysis (ROBNCA), is proposed which has the inherent ability to counteract the presence of outliers in the data $\boldsymbol{Y}$ by explicitly modelling the outliers as an additional sparse matrix. The iterative algorithm estimates each of the parameters efficiently at each iteration, and delivers superior consistency and greater accuracy for TFA estimation.

(2) A particularly attractive feature of the ROBNCA algorithm is the derivation of a closed form solution for the estimation of the connectivity matrix $\boldsymbol{A}$, a major source of high computational complexity in contemporary algorithms. To further lower the computational burden, a still faster closed form solution is derived that requires matrix inversion of much smaller size. The resulting algorithm is comparable with FastNCA in terms of computational complexity, and is hundreds of times faster than NI-NCA.

(3) The performance of ROBNCA is tested on Haemoglobin test data from (Jacklin *et al.*, 2012) for both low and highly correlated source signals.

ROBNCA is seen to outperform the state-of-the-art algorithms for estimating both $\boldsymbol{A}$ and $\boldsymbol{S}$ in terms of mean square error (MSE). In addition, ROBNCA is applied to yeast cell cycle data (Lee *et al.*, 2002) and *Escherichia coli* data (Kao *et al.*, 2004), and by plotting the standard deviation of estimates, it is observed that ROBNCA offers better consistency than FastNCA and NI-NCA.

2 METHODS

2.1 NCA with outliers

Most of the contemporary algorithms have studied the gene network construction problem using NCA with Gaussian noise models. However, inaccuracies in measurement procedures and abnormal gene responses often render heavier tails to the gene expression data, and Gaussian noise models may no longer be a natural fit in these cases. The decomposition techniques used in the available algorithms are highly sensitive to the presence of outliers i.e. the samples that do not conform to the Gaussian noise model, and their estimation capabilities are extremely susceptible to outliers. As a consequence, the gene network inference becomes unreliable for practical purposes. Therefore, we focus on deriving computationally efficient NCA algorithms that are robust to the presence of outliers.

Towards that end, we take the approach of explicitly modelling the outliers as an additional matrix that corrupts the data points. From (1), it follows that the complete system model that accounts for the presence of outliers as well as noise can be expressed as

$$\boldsymbol{Y}=\boldsymbol{AS}+\boldsymbol{O}+\boldsymbol{\Gamma}, \tag{3}$$

where the matrix $\boldsymbol{O}$ denotes the outliers. The outlier matrix $\boldsymbol{O}$ is a column sparse matrix, as there are typically a few outliers. The joint optimization problem for the estimation of the three parameters, which also allows for controlling outlier sparsity, can be formulated as

$$\begin{aligned}\left\{\widehat{\boldsymbol{A}},\widehat{\boldsymbol{S}},\widehat{\boldsymbol{O}}\right\} &= \arg\min_{\boldsymbol{A},\boldsymbol{S},\boldsymbol{O}} \|\boldsymbol{Y}-\boldsymbol{AS}-\boldsymbol{O}\|_F^2+\lambda_0\|\boldsymbol{O}\|_0 \\ &\text{such that} \quad \boldsymbol{A}(I)=0,\end{aligned} \tag{4}$$

where the non-convex l_0 norm $\|\boldsymbol{O}\|_0$ denotes the number of non-zero columns in $\boldsymbol{O}$, and the extent of sparsity in the columns of $\boldsymbol{O}$ is controlled by the tuning parameter λ_0. The optimization problem in (4) is reminiscent of compressive sampling techniques based on the l_0 norm, and are known to be NP-hard (Tropp, 2006). Therefore, some relaxation is needed to solve the joint optimization problem without incurring exponentially increasing computational complexity. A viable alternative is the column-wise l_2 sum i.e. $\|\boldsymbol{O}\|_{2,c}=\sum_{k=1}^{K}\|\boldsymbol{o}_k\|_2$, which is the closest convex

approximation of $\|\boldsymbol{O}\|_0$ (Tropp, 2006). With this relaxation, the resulting joint optimization problem can be expressed as

$$\left\{\widehat{\boldsymbol{A}},\widehat{\boldsymbol{S}},\widehat{\boldsymbol{O}}\right\} = \arg\min_{\boldsymbol{A},\boldsymbol{S},\boldsymbol{O}} \|\boldsymbol{Y}-\boldsymbol{A}\boldsymbol{S}-\boldsymbol{O}\|_F^2 + \lambda_2\|\boldsymbol{O}\|_{2,c} \quad \text{such that} \quad \boldsymbol{A}(I)=0. \tag{5}$$

Our goal is to estimate the three parameters $\boldsymbol{A}$, $\boldsymbol{S}$ and $\boldsymbol{O}$ by solving the optimization problem (5). However, it can be noticed that the optimization problem is not jointly convex with respect to (w.r.t) $\{\boldsymbol{A},\boldsymbol{S},\boldsymbol{O}\}$. Therefore, we resort to an iterative algorithm that alternately optimizes (3) w.r.t one parameter at a time.

2.2 The ROBNCA algorithm

The update of each of the parameters, $\boldsymbol{S}(j)$, $\boldsymbol{A}(j)$ and $\boldsymbol{O}(j)$, at an iteration j is discussed as follows.

2.2.1 Update of the TFA matrix At iteration j, the value of the parameter $\boldsymbol{S}(j)$ is updated by minimizing the objective function (3) w.r.t $\boldsymbol{S}$, while fixing the parameters $\boldsymbol{A}$ and $\boldsymbol{O}$ to their respective values at iteration $(j-1)$. By defining the matrix $\boldsymbol{X}(j)=\boldsymbol{Y}-\boldsymbol{O}(j-1)$, the optimization problem can be written as

$$\boldsymbol{S}(j) = \arg\min_{\boldsymbol{S}} \|\boldsymbol{X}(j)-\boldsymbol{A}(j-1)\boldsymbol{S}\|_F^2. \tag{6}$$

Because the connectivity matrix $\boldsymbol{A}(j-1)$ has full column rank (by virtue of NCA criterion 1), the matrix $\boldsymbol{A}^T(j-1)\boldsymbol{A}(j-1)$ is invertible. Therefore, an estimate of the TFA matrix $\boldsymbol{S}$ at the j^{th} iteration can be readily obtained as

$$\boldsymbol{S}(j) = \left(\boldsymbol{A}^T(j-1)\boldsymbol{A}(j-1)\right)^{-1}\boldsymbol{A}^T(j-1)\boldsymbol{X}(j). \tag{7}$$

The estimate $\boldsymbol{S}(j)$, so obtained, is used in the upcoming steps to determine $\boldsymbol{A}$ and $\boldsymbol{O}$.

2.2.2 Update of the connectivity matrix The next step in the iterative algorithm is to solve the optimization problem (3) w.r.t the matrix $\boldsymbol{A}$, while fixing the values of the parameters $\boldsymbol{S}$ and $\boldsymbol{O}$ to $\boldsymbol{S}(j)$ and $\boldsymbol{O}(j-1)$, respectively. The resulting optimization problem can be written as

$$\boldsymbol{A}(j) = \arg\min_{\boldsymbol{A}} \|\boldsymbol{X}(j)-\boldsymbol{A}\boldsymbol{S}(j)\|_F^2 \quad \text{such that} \quad \boldsymbol{A}(I)=0 \tag{8}$$

Remark 1. *The optimization problem (8) was also considered in the original work on NCA by Liao et al. (2003). However, a closed form solution was not provided and the proposed algorithm relied on costly optimization techniques to update the matrix $\boldsymbol{A}$. Because this minimization needs to be performed at each iteration until convergence, the ALS algorithm is known to be extremely slow for large networks, and computational resources required may be prohibitive (Jacklin et al., 2012). Hence, it is imperative that a closed form solution is obtained for the optimization problem in (8), so that the algorithm is faster and efficient.*

Without loss of generality, we can consider the transposed system

$$\widetilde{\boldsymbol{X}} = \widetilde{\boldsymbol{S}}\widetilde{\boldsymbol{A}} + \widetilde{\boldsymbol{\Gamma}}. \tag{9}$$

where $\widetilde{\boldsymbol{X}}$, $\widetilde{\boldsymbol{S}}$, $\widetilde{\boldsymbol{A}}$ and $\widetilde{\boldsymbol{\Gamma}}$ denote the transpose of the original matrices, respectively. The resulting equivalent optimization problem can now be stated as

$$\widetilde{\boldsymbol{A}}(j) = \arg\min_{\widetilde{\boldsymbol{A}}} \|\widetilde{\boldsymbol{X}}(j)-\widetilde{\boldsymbol{S}}(j)\widetilde{\boldsymbol{A}}\|_F^2 \quad \text{such that} \quad \widetilde{\boldsymbol{A}}(\tilde{I})=0, \tag{10}$$

where $\tilde{I}$ is the set of all indices where the entries of the matrix $\widetilde{\boldsymbol{A}}$ are known to be zero. The following theorem presents a closed form solution of the optimization problem (10), herein referred to as ROBNCA 1.

Theorem 1. *The solution of (10) at the jth iteration is given by*

$$\widetilde{\boldsymbol{a}}_n(j) = \boldsymbol{Q}^{-1}(j)\left[\widetilde{\boldsymbol{w}}_n(j) - \boldsymbol{C}_n^T\boldsymbol{\Psi}^{-1}(j)\boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\widetilde{\boldsymbol{w}}_n(j)\right], \tag{11}$$

where $\boldsymbol{\Psi}(j)=\boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\boldsymbol{C}_n^T$, $\widetilde{\boldsymbol{w}}_n(j)=\widetilde{\boldsymbol{S}}^T(j)\widetilde{\boldsymbol{x}}_n(j)$, the symmetric matrix $\boldsymbol{Q}(j)=\widetilde{\boldsymbol{S}}^T(j)\widetilde{\boldsymbol{S}}(j)$ and $\widetilde{\boldsymbol{a}}_n$ and $\widetilde{\boldsymbol{x}}_n$ represent the n^{th} columns of matrices $\widetilde{\boldsymbol{A}}$ and $\widetilde{\boldsymbol{X}}$, respectively. The $L_n\times M$ matrix $\boldsymbol{C}_n$ is a matrix of zeroes except $\boldsymbol{C}_n(\tilde{I}_n)=1$, where $\tilde{I}_n$ is the set of indices where the entries of $\widetilde{\boldsymbol{a}}_n$ are zero, and L_n denotes the number of zero entries in $\widetilde{\boldsymbol{a}}_n$.

Proof. The n^th column of (9) can be written as

$$\widetilde{\boldsymbol{x}}_n = \widetilde{\boldsymbol{S}}\widetilde{\boldsymbol{a}}_n + \widetilde{\gamma}_n. \tag{12}$$

The objective function in (10) can be equivalently expressed as

$$\|\widetilde{\boldsymbol{X}}(j)-\widetilde{\boldsymbol{S}}(j)\widetilde{\boldsymbol{A}}\|_F^2 = \sum_{n=1}^{N}\|\widetilde{\boldsymbol{x}}_n(j)-\widetilde{\boldsymbol{S}}(j)\widetilde{\boldsymbol{a}}_n\|^2. \tag{13}$$

The constraint $\widetilde{\boldsymbol{A}}(\tilde{I})=0$ can be written as a set of n constraints $\boldsymbol{C}_n\widetilde{\boldsymbol{a}}_n=\boldsymbol{0}$ for $n=1,\ldots,N$. The $L_n\times M$ matrix $\boldsymbol{C}_n$ is constructed such that it consists of all zeroes except $\boldsymbol{C}_n(\tilde{I}_n)=1$. For instance, if $M=6$, and $\widetilde{\boldsymbol{a}}_n=[a_{n_1},a_{n_2},0,a_{n_4},0,a_{n_6}]^T$, the 2×6 matrix $\boldsymbol{C}_n$ consists of all zeroes except $\boldsymbol{C}_n(1,3)=\boldsymbol{C}_n(2,5)=1$. It can be easily verified that the matrix $\boldsymbol{C}_n$ so constructed has full row rank.

The optimization problem in (10) can now be written as

$$\widetilde{\boldsymbol{A}}(j) = \arg\min_{\widetilde{\boldsymbol{A}}} \sum_{n=1}^{N}\|\widetilde{\boldsymbol{x}}_n(j)-\widetilde{\boldsymbol{S}}(j)\widetilde{\boldsymbol{a}}_n\|^2 \quad \text{such that} \quad \boldsymbol{C}_n\widetilde{\boldsymbol{a}}_n=\boldsymbol{0},\quad \forall n=1,\ldots,N. \tag{14}$$

The optimization problem is, therefore, separable in terms of columns of $\widetilde{\boldsymbol{A}}$, and can be equivalently solved by considering one column at a time. This also reduces the computational complexity of estimating the connectivity matrix $\widetilde{\boldsymbol{A}}$. Henceforth, we will use convex optimization techniques to derive a closed form solution of the separable optimization problem. For the n^th column, we have

$$\widetilde{\boldsymbol{a}}_n(j) = \arg\min_{\widetilde{\boldsymbol{a}}_n} \frac{1}{2}\widetilde{\boldsymbol{a}}_n^T\boldsymbol{Q}(j)\widetilde{\boldsymbol{a}}_n - \widetilde{\boldsymbol{w}}_n^T(j)\widetilde{\boldsymbol{a}}_n \quad \text{such that} \quad \boldsymbol{C}_n\widetilde{\boldsymbol{a}}_n=\boldsymbol{0}, \tag{15}$$

where the objective function is re-scaled and terms independent of $\widetilde{\boldsymbol{a}}_n$ are neglected. The Lagrangian dual function can be expressed as

$$\mathcal{L}(\widetilde{\boldsymbol{a}}_n,\mu) = \frac{1}{2}\widetilde{\boldsymbol{a}}_n^T\boldsymbol{Q}(j)\widetilde{\boldsymbol{a}}_n - \widetilde{\boldsymbol{w}}_n^T(j)\widetilde{\boldsymbol{a}}_n + \mu^T\boldsymbol{C}_n\widetilde{\boldsymbol{a}}_n.$$

The Karush–Kuhn–Tucker (KKT) conditions can be written as (Boyd and Vandenberghe, 2004)

$$\boldsymbol{Q}(j)\widetilde{\boldsymbol{a}}_n - \widetilde{\boldsymbol{w}}_n(j) + \boldsymbol{C}_n^T\mu = \boldsymbol{0} \tag{16}$$

$$\boldsymbol{C}_n\widetilde{\boldsymbol{a}}_n = \boldsymbol{0}. \tag{17}$$

Lemma 1. *The KKT conditions are necessary and sufficient for the optimization problem (15).*

Proof. Since the optimization problem (15) contains linear equality constraints, the KKT conditions are necessary for optimality (Boyd and Vandenberghe, 2004). Let any $\widetilde{\boldsymbol{a}}_n^*$ be a local minimum. Then, since the KKT conditions are necessary, there exists a Lagrange multiplier μ^* such that $(\widetilde{\boldsymbol{a}}_n^*,\mu^*)$ is the solution to the system of equations in (16) and (17). Now since the objective function is convex, it follows that $\widetilde{\boldsymbol{a}}_n^*$ is also a global minimum (Boyd and Vandenberghe, 2004). This implies that the KKT conditions are also sufficient for optimality.

Hence, a solution to (15) can be obtained by solving the KKT system of equations. Using (16), it follows that

$$\widetilde{\boldsymbol{a}}_n = \boldsymbol{Q}^{-1}(j)\left(\widetilde{\boldsymbol{w}}(j) - \boldsymbol{C}_n^T\mu\right), \tag{18}$$

where the matrix $\boldsymbol{Q}(j)$ is indeed invertible by virtue of the linear independence of the rows of $\boldsymbol{S}$ (NCA criterion 3). Substituting (18) in (17), we have

$$\boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\boldsymbol{C}_n^T\mu = \boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\widetilde{\boldsymbol{w}}(j).$$

Since the matrix $\boldsymbol{C}_n$ has full row rank, the matrix $\boldsymbol{\Psi}(j) \triangleq \boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\boldsymbol{C}_n^T$ is invertible. The Lagrange multiplier can, therefore, be expressed as

$$\mu = \boldsymbol{\Psi}^{-1}(j)\boldsymbol{C}_n\boldsymbol{Q}^{-1}(j)\widetilde{\boldsymbol{w}}(j). \tag{19}$$

Upon substituting (19) in (18), the solution $\widetilde{\boldsymbol{a}}_n$ in THEOREM 1 readily follows.

Therefore, using THEOREM 1, an estimate of $\widetilde{\boldsymbol{A}}(j)$ can be efficiently obtained and this approach results in substantial reduction in computation complexity compared with the ALS algorithm.

REMARK 2. *While the aforementioned closed form solution provides a significant advantage in terms of computational complexity over the ALS algorithm, we note that the solution requires inverting the matrix* $\boldsymbol{Q}$. *For large networks, this can potentially be a large matrix, whose inverse incurs computational load, and may lead to inaccuracies as well. In the following discussion, we derive a still faster algorithm, ROBNCA 2, that takes advantage of the special structure of the column vector* $\widetilde{\boldsymbol{a}}_n$ *and provides added savings over the closed form solution derived in THEOREM 1.*

We begin by noting that the rows of $\widetilde{\boldsymbol{X}}$ and $\widetilde{\boldsymbol{A}}$ can always be reordered in (9). Hence, without loss of generality, the vector $\widetilde{\boldsymbol{a}}_n$ can be partitioned as

$$\widetilde{\boldsymbol{a}}_n = \begin{bmatrix} \overline{\boldsymbol{a}}_n \\ \boldsymbol{0}_{L_n\times 1} \end{bmatrix}, \tag{20}$$

where $\overline{\boldsymbol{a}}_n \in \mathcal{R}^{(M-L_n)\times 1}$ is a vector consisting of the non-zero entries in $\widetilde{\boldsymbol{a}}_n$. Construct an $L_n \times M$ matrix $\boldsymbol{U}_n$ such that

$$\boldsymbol{U}_n = \begin{bmatrix} \boldsymbol{0}_{L_n\times(M-L_n)} & \boldsymbol{I}_{L_n} \end{bmatrix}. \tag{21}$$

With the above definition, the optimization problem (15) can be equivalently represented as

$$\begin{aligned} \widetilde{\boldsymbol{a}}_n(j) = \arg\min_{\widetilde{\boldsymbol{a}}_n} \frac{1}{2}\widetilde{\boldsymbol{a}}_n^T\boldsymbol{Q}(j)\widetilde{\boldsymbol{a}}_n - \widetilde{\boldsymbol{w}}_n^T(j)\widetilde{\boldsymbol{a}}_n \\ \text{such that} \quad \boldsymbol{U}_n\widetilde{\boldsymbol{a}}_n = \boldsymbol{0}. \end{aligned} \tag{22}$$

Define a substitution

$$\widetilde{\boldsymbol{a}}_n = \boldsymbol{V}_n\overline{\boldsymbol{a}}_n, \tag{23}$$

where the $M \times L_n$ matrix $\boldsymbol{V}_n$ is constructed such that it lies in the null space of the matrix $\boldsymbol{U}_n$, i.e. $\boldsymbol{U}_n\boldsymbol{V}_n = \boldsymbol{0}$. The matrix $\boldsymbol{V}_n$ is, therefore, given by

$$\boldsymbol{V}_n = \begin{bmatrix} \boldsymbol{I}_{(M-L_n)} \\ \boldsymbol{0}_{L_n\times(M-L_n)} \end{bmatrix}. \tag{24}$$

By substituting $\widetilde{\boldsymbol{a}}_n$ from (23) into (22), and noting that the constraint is always satisfied due to the construction of $\boldsymbol{V}_n$, we have an unconstrained optimization problem in the variable $\overline{\boldsymbol{a}}_n$ given by

$$\overline{\boldsymbol{a}}_n(j) = \arg\min_{\overline{\boldsymbol{a}}_n} \frac{1}{2}\overline{\boldsymbol{a}}_n^T\boldsymbol{V}_N^T\boldsymbol{Q}(j)\boldsymbol{V}_N\overline{\boldsymbol{a}}_n - \widetilde{\boldsymbol{w}}_n^T(j)\boldsymbol{V}_N\overline{\boldsymbol{a}}_n. \tag{25}$$

The solution of the aforementioned unconstrained quadratic optimization problem can be easily obtained as

$$\overline{\boldsymbol{a}}_n(j) = \left(\boldsymbol{V}_n^T\boldsymbol{Q}(j)\boldsymbol{V}_n\right)^{-1}\boldsymbol{V}_n^T\widetilde{\boldsymbol{w}}_n(j), \tag{26}$$

where the matrix $\boldsymbol{V}_n^T\boldsymbol{Q}(j)\boldsymbol{V}_n$ is invertible, as $\boldsymbol{V}_n$ has full column rank.

The symmetric invertible matrix $\boldsymbol{Q}(j)$ can be partitioned as

$$\boldsymbol{Q}(j) = \begin{bmatrix} \boldsymbol{Q}_{11}(j) & \boldsymbol{Q}_{12}(j) \\ \boldsymbol{Q}_{21}(j) & \boldsymbol{Q}_{22}(j) \end{bmatrix},$$

where the invertible matrix $\boldsymbol{Q}_{11}(j)$ is the upper $(M-L_n)\times(M-L_n)$ submatrix of $\boldsymbol{Q}(j)$. From the structure of $\boldsymbol{V}_n$, the matrix $\boldsymbol{V}_n^T\boldsymbol{Q}(j)\boldsymbol{V}_n$ can be reduced as

$$\begin{aligned} &\boldsymbol{V}_n^T\boldsymbol{Q}(j)\boldsymbol{V}_n \\ &= \begin{bmatrix} \boldsymbol{I}_{(M-L_n)} & \boldsymbol{0}_{(M-L_n)\times L_n} \end{bmatrix}\begin{bmatrix} \boldsymbol{Q}_{11}(j) & \boldsymbol{Q}_{12}(j) \\ \boldsymbol{Q}_{21}(j) & \boldsymbol{Q}_{22}(j) \end{bmatrix}\begin{bmatrix} \boldsymbol{I}_{(M-L_n)} \\ \boldsymbol{0}_{L_n\times(M-L_n)} \end{bmatrix} \\ &= \boldsymbol{Q}_{11}(j). \end{aligned} \tag{27}$$

Similarly, by partitioning $\widetilde{\boldsymbol{w}}_n(j)$ as

$$\widetilde{\boldsymbol{w}}_n(j) = \begin{bmatrix} \overline{\boldsymbol{w}}_n(j) \\ \widehat{\boldsymbol{w}}_n(j) \end{bmatrix},$$

it follows that

$$\boldsymbol{V}_n^T\widetilde{\boldsymbol{w}}_n(j) = \overline{\boldsymbol{w}}_n(j), \tag{28}$$

where $\overline{\boldsymbol{w}}_n(j)$ is the upper $(M-L_n)\times 1$ vector of $\widetilde{\boldsymbol{w}}_n(j)$. Collecting all the terms, the solution $\overline{\boldsymbol{a}}_n$ can now be compactly represented as

$$\overline{\boldsymbol{a}}_n(j) = \boldsymbol{Q}_{11}^{-1}(j)\overline{\boldsymbol{w}}_n(j). \tag{29}$$

Once all columns $\widetilde{\boldsymbol{a}}_n(j)$ are determined, the connectivity matrix $\boldsymbol{A}(j)$ can be easily updated.

REMARK 3. *By comparing the closed form solution derived in (11) with (29), it is clear that the latter only requires inverting a submatrix* $\boldsymbol{Q}_{11}(j)$ *of* $\boldsymbol{Q}(j)$. *Since the connectivity matrix is usually sparse and the number of non-zero entries* $(M-L_n)$ *in the* n^{th} *column is usually very small, inverting the* $(M-L_n)\times(M-L_n)$ *matrix* $\boldsymbol{Q}_{11}(j)$ *results in a considerable reduction in computational complexity and ensures a much faster implementation of the iterative algorithm.*

The respective computational times incurred in calculating (11) and (29) will be quantified in Section 3 to emphasize the usefulness of deriving (29).

2.2.3 Update of the outlier matrix The last step in the iterative algorithm pertains to the estimation of the outlier matrix $\boldsymbol{O}$ by using the values $\boldsymbol{S}(j)$ and $\boldsymbol{A}(j)$ obtained in the preceding steps. It is straightforward to notice that the optimization problem (3) w.r.t $\boldsymbol{O}$ decouples across the columns and results in K subproblems, each of which being expressed as follows:

$$\boldsymbol{o}_k(j) = \arg\min_{\boldsymbol{o}_k} \|\boldsymbol{b}_k(j) - \boldsymbol{o}_k\|_2^2 + \lambda_2\|\boldsymbol{o}_k\|_2, \tag{30}$$

where $\boldsymbol{b}_k(j) = \boldsymbol{y}_k - \boldsymbol{A}(j)\boldsymbol{s}_k(j)$. The solution to (30) is given by (Kekatos and Giannakis, 2011)

$$\boldsymbol{o}_k(j) = \frac{\boldsymbol{b}_k(j)\left(\|\boldsymbol{b}_k(j)\|_2 - \frac{\lambda_2}{2}\right)_+}{\|\boldsymbol{b}_k(j)\|_2}, \quad k = 1, \ldots, K \tag{31}$$

where $(g)_+ \triangleq \max(0, g)$. The solution (31) is intuitively satisfying, as it sets the outlier $\boldsymbol{o}_k(j)$ to zero whenever $\|\boldsymbol{b}_k(j)\|_2$ fails to exceed the threshold $\lambda_2/2$, where λ_2 is the sparsity-controlling parameter. Several approaches have been identified in the literature for selecting λ_2, which depend on any a-priori information available about the extent of sparsity (Giannakis *et al.*, 2011). If the concentration of outliers is unknown, a typical rule of thumb is to take $\lambda_2 = 0.7$ where this value has been determined to provide 95% asymptotic efficiency of the estimator (Kekatos and Giannakis, 2011). If a rough estimate of the concentration of outliers is available, (30) can be solved for a grid of values and selecting the λ_2 giving the expected number of outliers, which can be performed efficiently using the Group-LARS algorithm (Yuan and Lin, 2005). It is noted that the performance of the algorithm is insensitive to minor variations in the value of the parameter. Since the subproblems at each iteration have unique minimizers, and the non-differentiable regularization affects only the outlier matrix $\boldsymbol{O}$, the convergence of the ROBNCA algorithm is established using the results in (Tseng, 2001).

PROPOSITION 2. *As $j \to \infty$, the iterates generated by the ROBNCA algorithm converge to a stationary point of (3).*

It is important to point out that ROBNCA is significantly different from NI-NCA algorithm. NI-NCA, as the name suggests, is a non-iterative algorithm that uses a subspace-based method for the estimation of the connectivity matrix $\boldsymbol{A}$ using eigen-decomposition and relies on solving a constrained quadratic optimization problem, which has high computational cost. On the other hand, in ROBNCA, we propose two closed form solutions for the estimation of the connectivity matrix $\boldsymbol{A}$, which result in considerable reduction in computational complexity.

3 RESULTS AND DISCUSSION

This section investigates the observed performance of ROBNCA, in comparison with the state-of-the-art algorithms including FastNCA, NI-NCA and ALS in terms of MSE using both synthetic and real data. The efficiency and consistency of ROBNCA in estimating the TFAs under various scenarios is also illustrated. The datasets for all of the experiments as well as the MATLAB implementation of FastNCA and NI-NCA are downloaded from http://www.seas.ucla.edu/liaoj/download.htm and http://www.ece.ucdavis.edu/jacklin/NCA, respectively.

3.1 Synthetic and haemoglobin test data

First, to evaluate the performance of various algorithms, test data from (Liao *et al.*, 2003) is used. The spectroscopy data consist of $M = 7$ haemoglobin solutions formed by mixing up $N = 3$ pure haemoglobin components. The connectivity matrix in this case represents the concentration and presence or absence of each component in the mixture. In addition, the structure of this matrix is validated to comply with the NCA criteria. The absorbance spectra are taken for wavelengths in the range of 380–700 nm, with 1 nm increments to get $K = 7$ observation points, which is defined as $\boldsymbol{Abs} = \boldsymbol{C}\epsilon$ (Liao *et al.*, 2003), where the rows of $\boldsymbol{Abs}$ give the absorbance spectra for the range of wavelengths, $\boldsymbol{C}$ denotes the connectivity matrix and ϵ gives the spectra of the pure components. The importance of using these data is that this experiment mimics the gene regulatory network very closely and contains all of its key properties. The knowledge of the pure spectra helps us to effectively evaluate the performance of various NCA algorithms. In addition, using the data from (Liao *et al.*, 2003) and (Jacklin *et al.*, 2012) ensures a fair comparison.

The proposed algorithm is tested against varying noise for two important scenarios: (i) when the source signals are correlated, and (ii) the observed data are corrupted with outliers. Using the same connectivity matrix, source signals were generated which had low, moderate and high correlation (Jacklin *et al.*, 2012). The outliers are artificially added to the data by modelling them as a Bernoulli process. The success probability indicates the concentration of outliers present and is assumed to be the same for all the genes. Since only a few points are expected to be corrupted in the real data, the outliers are assumed to be sparse and therefore the success probability of presence of outliers is kept small. The performance of ROBNCA, FastNCA and NI-NCA is evaluated in the aforementioned scenarios. ROBNCA algorithm is implemented in MATLAB. Since the observed data matrix $\boldsymbol{Y}$ is expected to contain outlying points, the algorithms are assessed by computing the MSE incurred in estimating the matrices $\boldsymbol{A}$ and $\boldsymbol{S}$, instead of fitting error for $\boldsymbol{Y}$. The comparison with ALS is omitted here because it takes much longer to run as will be shown in the next subsection.

3.1.1 Impact of correlation The algorithms are first tested for low and highly correlated source signals by varying the signal-to-noise ratio (SNR). The noise is modelled as Gaussian in all the experiments. The results are averaged over 100 iterations and are depicted in Figure 1. It is observed that the presence of a small amount of outliers makes the estimation using FastNCA unreliable and inconsistent for both low and highly correlated signals. On the other hand, NI-NCA is able to estimate $\boldsymbol{S}$ better than FastNCA, and the estimation of $\boldsymbol{A}$ is quite accurate and consistent as well. It can be observed that the overall estimation performance for $\boldsymbol{A}$ is much better and more consistent than that of $\boldsymbol{S}$. The reason for this could be attributed to the availability of some prior information for the former. Since ROBNCA takes into account the presence of outliers in the observed data, it outperforms the other two algorithms for estimating both $\boldsymbol{A}$ and $\boldsymbol{S}$ and its consistent performance should be contrasted with the unboundedness and unpredictability exhibited by the other two algorithms. In general, the performance of all the algorithms improves with the increase in SNR and degrades with the increase in correlation of the source signals.

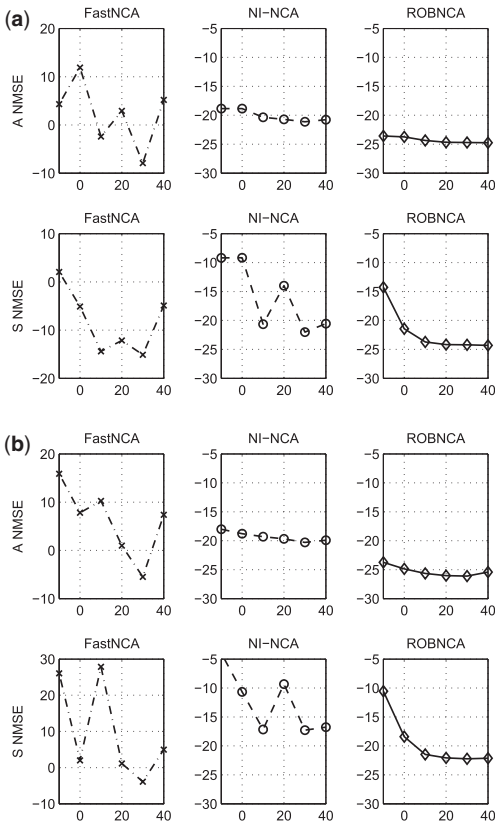

Fig. 1. Impact of correlation: Normalized mean square error (NMSE) (dB) for different algorithms and different datasets with (**a**) low correlated TFAs and (**b**) highly correlated TFAs against varying SNR (dB), with the level of outliers = 0.05

3.1.2 Impact of outliers As noted earlier, the presence of outliers can severely affect the performance of algorithms. It is therefore, important to investigate the impact of the presence of outlying points in the observation matrix $\boldsymbol{Y}$. Comparison performed for low and high concentration of outliers is depicted in Figure 2. It is observed from Figure 2a that in the case of low concentration of outliers, NI-NCA provides good accuracy for $\boldsymbol{A}$ and estimates it quite consistently. The estimation of $\boldsymbol{S}$ gives a small MSE as well and generally performs consistently. FastNCA, however, is not able to estimate both the matrices even for high SNRs. This indicates its high vulnerability to the presence of even a small number of outliers. In case of a higher concentration of outliers, the performance of NI-NCA degrades a little bit as depicted in Figure 2b. It is observed that ROBNCA is able to estimate the two matrices for both low and high outliers, and outperforms the other two algorithms.

The estimation of $\boldsymbol{O}$ matrix is shown in the Supplementary Material where Figure 1 depicts the outliers present in the synthetic data and their estimates using ROBNCA algorithm. It is noted that ROBNCA is able to identify the outliers very well. Figure 2 shows the recovered signal $\boldsymbol{AS}$ after subtracting the outlier matrix O from the data matrix $\boldsymbol{X}$. It can be observed that the recovered signal is a good match with the original signal.

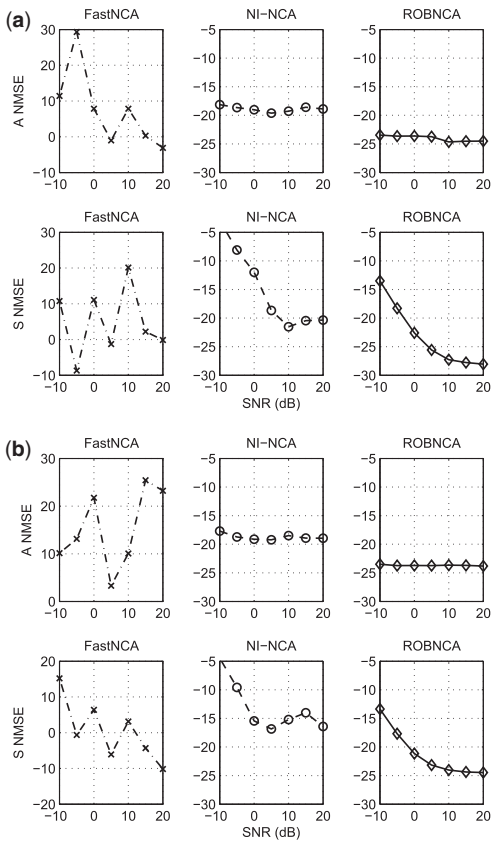

Fig. 2. Impact of outliers: Normalized mean square error (NMSE) (dB) for different algorithms and different datasets with (**a**) level of outliers = 0.01 and (**b**) level of outliers = 0.1 against varying SNR (dB) for a highly correlated dataset

These experiments indicate that ROBNCA solves the estimation problem with much more accuracy than NI-NCA and FastNCA. It is important to emphasize here that the MSE for NI-NCA is always higher than ROBNCA and its computational complexity is many times greater than the latter, which can prove to be a bottle-neck in case of large datasets.

3.2 Results for real data

We now turn our attention to the comparison of these algorithms on real data. Two datasets are considered for this purpose, which are the *Saccharomyces cerevisiae* cell cycle data (Lee *et al*., 2002) and *E.coli* data (Kao *et al*., 2004). The transcription factor activities are estimated for the TFs of interest in each experiment, and the results are compared for different algorithms. In addition, the variability of the estimates is evaluated using the subnetwork analysis (Yang *et al*., 2005) which will be explained in the following subsections.

3.2.1 S.cerevisiae cell cycle data The algorithms discussed in this article are applied to the yeast cell cycle data from (Lee *et al*., 2002) and (Spellman *et al*., 1998). To assess the performance and variability of the various NCA algorithms, *subnetwork analysis* is performed, which has also been used previously in (Chang *et al*., 2008), (Yang *et al*., 2005) and (Jacklin *et al*., 2012). The core idea behind this analysis is to divide the set of transcription factors into a number of smaller subsets, which are not mutually disjoint, where the intersection of these subsets contains the TFs of interest. The subnetworks were constructed to satisfy the gNCA criteria (Tran *et al*., 2005), which requires that the number of TFAs M should be less than the number of sample points K. These sub-networks are used to estimate the transcription factor activities independent of each other. These TFA estimates are then compared and a smaller disagreement between these estimates is a measure of consistency of the algorithm. This indicates that the results obtained are reliable despite of the presence or absence of certain genes or TFs from the experiment. The disagreement can be quantified as: $\text{disagreement}(i) = \frac{1}{K}\sum_i \left[\max_n s_{n,i}(k) - \min_n s_{n,i}(k)\right]$ where s indicates the rows of matrix $\boldsymbol{S}$, i is the TF index and n is the subnetwork index. The Yeast cell cycle dataset consists of results from three different synchronization experiments. The first experiment is the synchronization by elutriation, which is composed of one cell cycle from 0 to 390 min. The data consist of 14 points sampled at 30-min intervals. The second experiment performs the synchronization by $\alpha-$ factor arrest and contains two cell cycles from 0 to 119 min. A total of 18 samples are taken every 7 min. The synchronization in the third set is the result of cdc15 temperature sensitive mutant with samples taken every 20 min from 0 to 300 min. The data from the three experiments are concatenated to form one large dataset. The Yeast cell cycle study has 11 TFs of interest (Chang *et al*., 2008), which are Ace2, Fkh1, Fkh2, Mbp1, Mcm1, Ndd1, Skn7, Stb1, Swi4, Swi5 and Swi6. This section compares the performance of the NCA algorithms for these TFs and the related genes.

The subnetwork analysis is carried out by dividing the original network into four subnetworks each consisting of 40 TFs and the number of genes varies from 921 to 1247. The aforementioned 11 TFs are included in each of the subsets. The structure of $\boldsymbol{A}$ is

verified to satisfy the NCA criterion (2) for all of the subnetworks. The reconstruction of the 11 TFAs, which is the average of the four subnetworks, using ROBNCA, FastNCA and NI-NCA is depicted in Figure 3. The TFA estimation using ALS algorithm is skipped here because the algorithm takes very long to run for this large dataset. The results for the three experiments are shown separately in the three columns. The TFAs for these experiments are expected to have a periodic behavior with one, two and three cycles in elutriation, $\alpha-$ factor and cdc-15, respectively, which can easily be corroborated from the figure. The results from ROBNCA differ from FastNCA in some of the instances. On the other hand, NI-NCA provides very similar estimates to that of ROBNCA. It can be inferred that the results of these two algorithms are more reliable as compared with FastNCA because the former reveal the periodic behavior in almost all of the TFs.

We now look to investigate the consistency of the algorithms. The disagreement between the TFA estimates of the four subnetworks is calculated and the results are shown in Figure 4a. Out of the three algorithms considered, ROBNCA incurs the smallest disagreement. The performance of NI-NCA is somewhat comparable; however, FastNCA shows a high degree of inconsistency.

The simulations for standard deviation for TFAs are presented in the Supplementary Material for ROBNCA, NI-NCA and FastNCA. It is noted from Supplementary Figures S5–S7 in the Supplementary section that ROBNCA yields the lowest variation whereas FastNCA shows much higher variation in the TFA estimates than both the other algorithms. It can therefore be concluded that ROBNCA outperforms NI-NCA both in terms of estimating the TFAs as well as in terms of consistency for Yeast cell cycle data.

3.2.2 E.coli data The performance of NCA algorithms is now tested for *E.coli* data. This dataset contains the gene expression profiles obtained during transition of the sole carbon source from glucose to acetate (Kao *et al.*, 2004). Out of 296 genes found to be of relevance during the carbon source transition, 100 genes were separated so that the resulting network satisfies the NCA criteria. A total of 16 TFs were identified to be related to this experiment, which are ArcA, CRP, CysB, FadR, FruR, GatR, IcIR, LeuO, Lrp, Narl, PhoB, PurR, RpoE, RpoS, TrpR and TyrR. We perform subnetwork analysis to this data to estimate the transcription factor activities for the 16 TFs of interest. The downloaded network is divided into four subnetworks containing 81, 82, 85 and 88 genes, respectively. The number of TFs in each subnetwork is fixed to 20, where the aforementioned 16 TFs are included in all of them. The samples are taken at 5, 15, 30 and 60 min and then every hour until 6 h. Multiple samples are taken at these instances, which make a total of 25 time points. The advantage of using this data is that the ALS algorithm can be added to the performance evaluation because of its smaller subnetworks. ALS is known to have prohibitive computational complexity (Jacklin *et al.*, 2012) and is included here only for the comparison of estimation accuracy. The reconstruction of TFAs is performed using the four algorithms, and the average of the TFA estimates from four subnetworks is depicted in Figure 5. The results from ROBNCA, NI-NCA and ALS are in agreement for almost all of the TFAs. In addition, these estimates are also similar to those found in (Kao *et al.*, 2004) except for a few TFAs. The reason for this small dissimilarity could be that, in this article, the estimates are obtained using the subnetworks whereas (Kao *et al.*, 2004) use the complete network of 100 genes. For 5 out of the 16 TFs, namely GatR, Lrp, NarL, TrpR and TyrR, FastNCA is not able to recover the TFAs. Moreover, the TFAs predicted by ROBNCA are similar to those predicted by ALS, which is the original solution as shown in Figure 5. It can therefore be inferred that ROBNCA estimates the TFAs more accurately than FastNCA.

The consistency of the algorithms is assessed for this experiment as well and the respective disagreement for each of the four algorithms is shown in Figure 4b. FastNCA is again seen to incur the maximum disagreement. NI-NCA and ALS perform better than FastNCA; however, ROBNCA gives the least disagreement for the four estimates of TFAs and performs the most consistently out of all the algorithms.

3.2.3 Computational complexity comparison An important feature of all gene network reconstruction algorithms is the computational complexity incurred in their implementation. The computational complexity of estimating $\boldsymbol{A}$ in (29) at a particular iteration is approximately $O(\sum_{n=1}^{N}(M-L_n)^3+(M-L_n)^2)$,

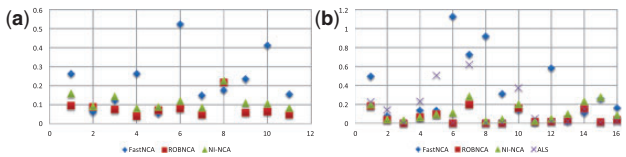

Fig. 4. Average disagreement for different algorithms across the subsets for TFAs. X-axis indicates the TFA index. Consistency comparison for (**a**) *S.cerevisiae* data and (**b**) *E.coli* data

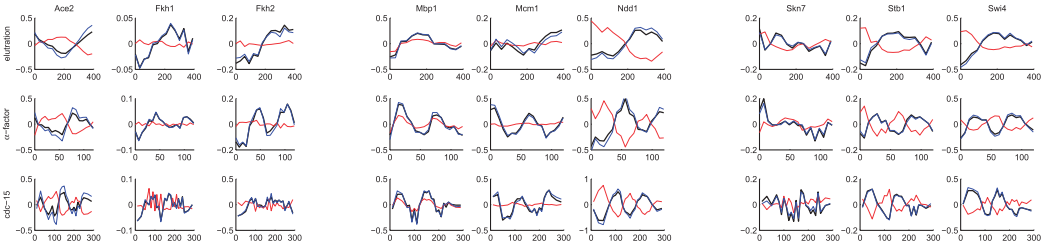

Fig. 3. TFAs reconstruction: Estimation of 11 TFAs (9 shown) of cell cycle-regulated yeast TFs. Average values of the TFs are shown for the four subnetworks. The results of ROBNCA (black), FastNCA (red) and NI-NCA (blue) are given

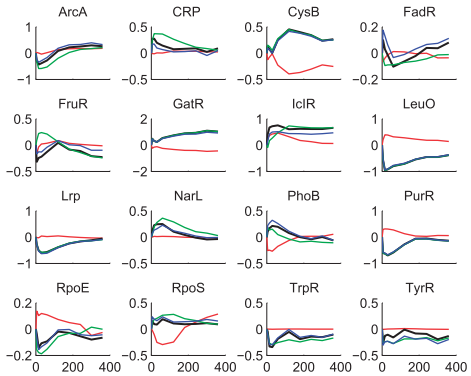

Fig. 5. TFAs reconstruction: Estimation of 16 TFAs of *E.coli*. Average values of TFs are shown. The results of ROBNCA (black), FastNCA (red), NI-NCA (blue) and ALS (green) are given

Table 1. Average computational time for various methods in seconds

Subset	***S.cerevisiae***				***E.coli***			
	1	2	3	4	1	2	3	4
FastNCA	0.2	0.2	0.24	0.2	0.014	0.007	0.007	0.008
ROBNCA 2	0.2	0.2	0.25	0.2	0.016	0.010	0.008	0.008
ROBNCA 1	1.0	0.8	0.99	0.8	0.020	0.018	0.016	0.023
NI-NCA LP	67	36	56.2	33	0.93	0.76	0.73	0.83
NI-NCA QP	71	30	125	97	0.59	0.13	0.13	0.13
ALS	Exceeds memory limit				5.3	6.0	7.1	3.5

where $(M - L_n)$ is the number of non-zero unknowns in the n^{th} column, which is usually small. We now compare the computational complexity of the four algorithms using the subnetwork data from Yeast and *E.coli*. Average runtime calculated in seconds is summarized for four subnetworks of each data in Table 1. These experiments were performed on a Windows 7 system with a 1.90 GHz Intel Core i7 processor on a Matlab 7.10.0. It is noted that the run time of ROBNCA is comparable with that of FastNCA and is hundreds of times faster than NI-NCA algorithms for both of its implementations, i.e. involving linear programming and quadratic programming. Moreover, the run time for ROBNCA is far superior to that of the ALS, a direct consequence of the closed form solution derived for estimating the connectivity matrix. It can also be observed that the faster closed form solution for estimating $\boldsymbol{A}$ (29) provides additional savings over its predecessor (11).

Therefore, it can be inferred from these experiments on synthetic and real datasets that ROBNCA renders superior performance than the contemporary algorithms not only on the yardsticks of accuracy and reliability, but also in terms of computational complexity. The high computational complexity of NI-NCA far outweighs the benefits it offers in terms of consistency. FastNCA has the smallest run time out of all the algorithms but has poor reliability and is the least robust to the presence of outliers in the data.

4 CONCLUSION

In this work, we present ROBNCA, an algorithm for robust network component analysis for estimating the TFAs. The ROBNCA algorithm accounts for the presence of outliers by modelling them as an additional sparse matrix. A closed form solution available at each step of the iterative ROBNCA algorithm ensures faster and reliable performance. The performance of the proposed ROBNCA algorithm is compared with NI-NCA and FastNCA for synthetic as well real datasets by varying SNR, degrees of correlation and outlier concentration. It is observed that while FastNCA is computationally simpler, yet the TFA recovery is inaccurate and unreliable, a direct consequence of the sensitivity of its decomposition approach to the presence of outliers. The NI-NCA algorithm offers performance somewhat comparable with the ROBNCA algorithm; however, the ROBNCA algorithm is much more computationally efficient and does not require solving costly optimization problems. Therefore, the cumulative benefits of robustness to the presence of outliers, higher consistency and accuracy compared with the existing state-of-the-art algorithms, and much lower computational complexity make ROBNCA well-suited to the analysis of gene regulatory networks, which invariably requires working with large datasets.

Funding: US National Science Foundation (NSF) grant [0915444] and QNRF-NPRP grant [09-874-3-235].

Conflict of Interest: none declared.

REFERENCES

Boyd,S. and Vandenberghe,L. (2004) *Convex Optimization*. Cambridge University Press, New York.

Chang,C. *et al.* (2008) Fast network component analysis (FastNCA) for gene regulatory network reconstruction from microarray data. *Bioinformatics*, **24**, 1349–1358.

Comon,P. (1992) Independent component analysis. *Higher-Order Statistics*, 29–38.

Finegold,M. and Drton,M. (2011) Robust graphical modeling of gene networks using classical and alternative t-distributions. *Ann. Appl.Stat.*, **5**, 1057–1080.

Galbraith,S.J. *et al.* (2006) Transcriptome network component analysis with limited microarray data. *Bioinformatics*, **22**, 1886–1894.

Giannakis,G. *et al.* (2011) USPACOR: Universal sparsity-controlling outlier rejection. In: *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference*. pp. 1952–1955.

Jacklin,N. *et al.* (2012) Noniterative convex optimization methods for network component analysis. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **9**, 1472–1481.

Jajamovich,G.H. *et al.* (2011) Bayesian multiple-instance motif discovery with bambi: inference of recombinase and transcription factor binding sites. *Nucleic Acids Res.*, **39**, e146.

Jolliffe,I. (1986) *Principal Component Analysis*. Springer-Verlag, New York, p. 487.

Kao,K.C. *et al.* (2004) Transcriptome-based determination of multiple transcription regulator activities in *escherichia coli* by using network component analysis. *Proc. Natl. Acad. Sci. USA*, **101**, 641–646.

Kekatos,V. and Giannakis,G.B. (2011) From sparse signals to sparse residuals for robust sensing. *IEEE Trans. Signal. Process*, **59**, 3355–3368.

Lähdesmäki,H. *et al.* (2008) Probabilistic inference of transcription factor binding from multiple data sources. *PLoS One*, **3**, e1820.

Lee,T.I. *et al.* (2002) Transcriptional regulatory networks in *saccharomyces cerevisiae*. *Sci. Signal.*, **298**, 799.

Liao,J. *et al.* (2003) Network component analysis: reconstruction of regulatory signals in biological systems. *Proc. Natl. Acad. Sci. USA*, **100**, 15522–15527.

Mateos,G. and Giannakis,G.B. (2012) Robust PCA as bilinear decomposition with outlier-sparsity regularization. *IEEE Trans. Signal Process.*, **60**, 5176–5190.

Spellman,P.T. *et al.* (1998) Comprehensive identification of cell cycle–regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Tran,L. *et al.* (2005) gNCA: a framework for determining transcription factor activity based on transcriptome: identifiability and numerical implementation. *Metab. Eng.*, **7**, 128–141.

Tropp,J.A. (2006) Just relax: convex programming methods for identifying sparse signals in noise. *IEEE Trans. Inf. Theory*, **52**, 1030–1051.

Tseng,P. (2001) Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.*, **109**, 475–494.

Yang,Y.-L. *et al.* (2005) Inferring yeast cell cycle regulators and interactions using transcription factor activities. *BMC Genomics*, **6**, 90.

Yuan,M. and Lin,Y. (2005) Model selection and estimation in regression with grouped variables. *J. R. Stat. Soc. Series B Stat. Methodol.*, **68**, 49–67.