Published in final edited form as:

Stat Anal Data Min. 2012 December 1; 5(6): . doi:10.1002/sam.11168.

Maximum Likelihood Estimation Over Directed Acyclic Gaussian Graphs

Yiping Yuan¹, Xiaotong Shen¹, and Wei Pan²

¹School of Statistics, University of Minnesota, Minneapolis, MN 55455, USA

²Division of Biostatistics, University of Minnesota, Minneapolis, MN 55455, USA

Abstract

Estimation of multiple directed graphs becomes challenging in the presence of inhomogeneous data, where directed acyclic graphs (DAGs) are used to represent causal relations among random variables. To infer causal relations among variables, we estimate multiple DAGs given a known ordering in Gaussian graphical models. In particular, we propose a constrained maximum likelihood method with nonconvex constraints over elements and element-wise differences of adjacency matrices, for identifying the sparseness structure as well as detecting structural changes over adjacency matrices of the graphs. Computationally, we develop an efficient algorithm based on augmented Lagrange multipliers, the difference convex method, and a novel fast algorithm for solving convex relaxation subproblems. Numerical results suggest that the proposed method performs well against its alternatives for simulated and real data.

Keywords

collapsed networks; nonconvex constraints; pairwise coordinate descent

1. INTRODUCTION

Directed acyclic graphical models have been useful in analyzing and visualizing causal relations among random variables. Major components of the models are nodes representing random variables and edges encoding conditional dependence relations between the enclosing nodes. In gene network analysis, a large amount of current biological knowledge has been organized as regulatory and other interactive networks, for instance, as a collection of pathways representing molecular and biochemical reactions in the Kyoto Encyclopedia of Genes and Genomes. In such a situation, a key question is to extract gene–gene causal effects, to advance our understanding of molecular interactions in a subnetwork of the genes. Therefore, we reconstruct multiple directed acyclic graphs (DAGs) given a ordering to detect structural changes over graphs in this article.

Reconstruction of a single DAG from data without a given ordering is challenging due to the enormous size of candidate DAGs that is super-exponential in the number of nodes. Existing approaches can be categorized into search-and-score and constraint-based methods [1]. For problems with a small or moderate number of nodes, both methods perform well [1–5]. Recently, interesting hybrid methods have been developed to reduce computational cost [6]. For high-dimensional problems with a large number of nodes, challenges remain on the

computational front. Note that even the Peter-Clark (PC) algorithm [7, 8], which has a polynomial runtime when a graph is sparse runs in the worst case in time that is exponential in the number of nodes. In fact, existing methods are impractical because the space of equivalence classes is conjectured to grow super-exponentially in the number of nodes [9].

Reconstruction of a DAG given a ordering is equivalent to sparse estimation of Cholesky decomposition of a covariance matrix, and thus is computationally feasible [10]. The ordering information is usually determined by a natural ordering of temporal observations, previous experiments, and prior knowledge. In this article, we reconstruct multiple DAGs given a known ordering. To our knowledge, estimation of structural changes over multiple DAGs has not yet been explored, although that over multiple undirected graphs has been studied [12–14]. In practice, identification of a change in causality structure arises from detecting a change corresponding to that of experimental conditions or responding to a certain event or treatment.

This paper focuses on maximum likelihood estimation of multiple DAGs under a structural equation model. It is known that maximum likelihood estimation breaks down when the number of variables exceeds the sample size. Even for a moderately sized problem, it always yields a complete graph and does not estimate graphical structures well. Therefore, different methods using penalization have been proposed for sparse estimation of graphical models [14–16, 10]. To achieve our goal of learning graphical structures, we construct two nonconvex constraints based on the truncated L_1 -function (TLP, [17]), as a computational surrogate of the L_0 -function, with one constraint imposing sparseness and the other encouraging a common structure. Computationally, with difference convex (DC) programming and augmented Lagrange multipliers, nonconvex minimization is solved through a sequence of convex subproblems iteratively. For each subproblem, we develop a fast algorithm to treat a constrained L_1 -problem, which we call pairwise coordinate descent algorithm.

The rest of the paper is organized as follows. Section 2 introduces the methodology, followed by our computational development in Section 3. Operating characteristics of the proposed method are examined on simulated and real data in Sections 4 and 5, respectively.

2. METHODOLOGY

Given L p -dimensional vectors of random variables $\mathbf{X}^{(l)} = (X_1^{(l)}, \dots, X_p^{(l)})^T$ with a known ordering, one from each population, we use L DAGs to describe causal relations within each population and to explore differences among the populations. That is, each component $X_i^{(l)}$ corresponds to one node in the l th DAG, with a directed edge between two nodes indicating a causal relation between them. Without loss of generality, we assume that $\mathbf{X}^{(l)}$ has been sorted according to its order, which means a causal relation is only possible from $X_i^{(l)}$ to $X_j^{(l)}$ for $i < j$.

To model causality among the components of $\mathbf{X}^{(l)}$, consider a structural equation model of the form

$$\mathbf{X}^{(l)} = \mathbf{A}^{(l)} \mathbf{X}^{(l)} + \boldsymbol{\varepsilon}^{(l)}, l=1, \dots, L, \quad (1)$$

where $\mathbf{A}^{(l)}$ is an adjacency matrix in which a nonzero jk -th element of $\mathbf{A}^{(l)}$ corresponds to a directed edge from parent node k to child node j with its value $A_{jk}^{(l)}$ indicating the strength of the relationship, and $\boldsymbol{\varepsilon}^{(l)} = (\varepsilon_1^{(l)}, \dots, \varepsilon_p^{(l)})^T$ is an independent latent variable vector

representing unexplained variations in the nodes and acting as random noises. Note that $A_{i,j}^{(l)}=0$ for $i < j$, since $\mathbf{X}^{(l)}$'s are assumed to be ordered. In addition, $A_{ii}^{(l)}=0$, for all i , as a self-loop is not allowed in a DAG. Therefore, the adjacency matrices $\mathbf{A}^{(l)}$, $l = 1, \dots, L$, are lower triangular with zero diagonal elements, that is, $A_{i,j}^{(l)}=0, j \geq i$. The model basically says that each $\mathbf{X}_j^{(l)}$ depends linearly on its parent variables and some latent variable $\varepsilon_j^{(l)}$. Here we assume that $\varepsilon_1^{(l)}, \dots, \varepsilon_p^{(l)}$ follow independent normal distributions, that is, $\varepsilon_j^{(l)} \sim \text{ind}N(0, (\sigma_j^{(l)})^2)$. This implies that $\mathbf{X}^{(l)}$'s follow multivariate normal distributions. Note that Eq. (1) becomes Gaussian autoregressive model when the subscript of $\mathbf{X}_i^{(l)}$ denotes the consecutive time. Our likelihood method is readily generalizable to other distributions. The reader may refer to Ref [18] for Eq. (1) and structural equations.

In Eq. (1), nonzero entries of $\mathbf{A}^{(l)}$ are uniquely specified by the l th DAG. Thus, we estimate $(\mathbf{A}^{(1)}, \dots, \mathbf{A}^{(L)})$ to preserve a common structure and identify differences among them.

For a total of $n = \sum_{l=1}^L n_l$ random samples, n_l samples are drawn according to Eq. (1) for each l to form an $n \times p$ data matrix $\mathcal{X}^{(l)} = (\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_p^{(l)})$, where each $\mathbf{x}_j^{(l)} = (x_{1,j}^{(l)}, \dots, x_{n_l,j}^{(l)})^T$ is an n_l -dimensional column vector for each node, and samples from different populations $\mathcal{X}^{(l)}$ are assumed to be independent. Note that an arbitrary mean vector can be incorporated by adding an intercept to Eq. (1).

Let $k^- = \{j = 1, \dots, k-1\}$ be a set of indices, with $k = 1$ indicating the null set. Let $\mathcal{X}_{j-}^{(l)} = (\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_{j-1}^{(l)})$, $\mathbf{A}_{j,j-}^{(l)} = (\mathbf{A}_{j,l}^{(l)}, \dots, \mathbf{A}_{j,j-1}^{(l)})$. The likelihood of $\mathcal{X}^{(l)}$ can be written as

$$f(\mathcal{X}^{(1)}, \dots, \mathcal{X}^{(L)}) = \prod_{l=1}^L f(\mathcal{X}^{(l)}) = \prod_{l=1}^L \prod_{j=1}^p f(\mathbf{x}_j^{(l)} | \mathcal{X}_{j-}^{(l)}) = \prod_{j=1}^p \prod_{l=1}^L f(\mathbf{x}_j^{(l)} | \mathcal{X}_{j-}^{(l)}). \quad (2)$$

This yields the negative log-likelihood

$$-\log f(\mathcal{X}^{(1)}, \dots, \mathcal{X}^{(L)}) = \sum_{j=1}^p \left(\sum_{l=1}^L \left(-\log f(\mathbf{x}_j^{(l)} | \mathcal{X}_{j-}^{(l)}) \right) \right). \quad (3)$$

Using the fact that $\mathbf{X}_j^{(l)} | \mathbf{X}_{j-}^{(l)}$ follows $N(\mathbf{A}_{j,j-}^{(l)} \mathbf{X}_{j-}^{(l)}, (\sigma_j^{(l)})^2)$ from Eq. (1), we obtain that

$$-\log f(\mathcal{X}^{(1)}, \dots, \mathcal{X}^{(L)}) = \sum_{j=1}^p \left(\sum_{l=1}^L \left(\frac{1}{2(\sigma_j^{(l)})^2} \left\| \mathbf{x}_j^{(l)} - \mathcal{X}_{j-}^{(l)} (\mathbf{A}_{j,j-}^{(l)})^T \right\|^2 + n_j \log \sigma_j^{(l)} \right) \right). \quad (4)$$

Maximizing Eq. (2), equivalently minimizing Eq. (4), may result in over-fitting and lead to fully connected DAGs, especially when the number of unknown parameters exceeds the sample size. We therefore regularize Eq. (3) through nonconvex constraints to pursue sparsity and detect structural changes. Note that the constrained approach is not equivalent to its penalized regularization counterpart because of nonconvexity in this case.

Our method is to regularize the number of nonzeros of the adjacency matrices as well as the number of pairwise differences between the corresponding entries across adjacency matrices. Let $\mathcal{S}$ be a set of index pairs in which a pair (l, s) indicates the possibility that $\mathbf{A}^{(l)}$

and $\mathbf{A}^{(s)}$ share some common entries and can be grouped or collapsed if data suggest so. Constraints are used to regularize, which are in the form:

$$\sum_{l=1}^L \|\mathbf{A}^{(l)}\|_0 \leq t_1, \quad \sum_{(l,s) \in \mathcal{L}} \|\mathbf{A}^{(l)} - \mathbf{A}^{(s)}\|_0 \leq t_2. \quad (5)$$

where $\|\mathbf{A}\|_0 := \sum_{i=1}^p \sum_{j=1}^p I(|A_{i,j}| \neq 0)$, is the L_0 -norm of $\mathbf{A}$, or the number of nonzero entries of $\mathbf{A}$, $t_1 = 0$ and $t_2 = 0$ are tuning parameters corresponding to the number of nonzeros of the adjacency matrices and the number of element-wise differences with respect to $\mathcal{L}$. A complete set $\mathcal{L} = \{(l, s) : 1 \leq l < s \leq L\}$ is used unless additional information is available. For example, a temporal set $\mathcal{L} = \{(l, l+1) : 1 \leq l \leq L-1\}$ is used in dynamic networks with l representing consecutive times.

To allow for different degrees of sparsity over different rows of lower triangular matrices $\mathbf{A}^{(l)}$, we replace (5) by p pairs of row-wise sparsity through $2p$ different tuning parameters $\{t_{1,j}, t_{2,j}\}$:

$$\sum_{l=1}^L \sum_{k=1}^{j-1} \left\| A_{j,k}^{(l)} \right\|_0 \leq t_{1,j}, \quad \sum_{(l,s) \in \mathcal{L}} \sum_{k=1}^{j-1} \left\| A_{j,k}^{(l)} - A_{j,k}^{(s)} \right\|_0 \leq t_{2,j}, j=1, \dots, p. \quad (6)$$

This is computationally feasible because the log-likelihood (3) is separable or decomposable in j . In fact, minimizing Eq. (3) subject to Eq. (6) reduces to p subproblems.

To circumvent computational difficulty of minimizing Eq. (3) subject to Eq. (6), we approximate the L_0 function there by a surrogate function, the truncated L_1 function (TLP,

[17], defined as $P_\lambda(x) = \min\left(\frac{|x|}{\lambda}, 1\right)$. As λ tends to 0, the TLP recovers the L_0 -function exactly. Now the constraints in Eq. (6) become

$$\sum_{l=1}^L \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)}) \leq t_{1,j}, \quad \sum_{(l,s) \in \mathcal{L}} \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)} - A_{j,k}^{(s)}) \leq t_{2,j}, j=1, \dots, p. \quad (7)$$

To simplify tuning, we introduce a single constraint for each row as opposed to the two constraints in Eq. (7), with new tuning parameters (κ_j, t_j) corresponding to $(t_{1,j}, t_{2,j})$,

$$\sum_{l=1}^L \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)}) + \kappa_j \sum_{(l,s) \in \mathcal{L}} \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)} - A_{j,k}^{(s)}) \leq t_j, j=1, \dots, p, \quad (8)$$

where κ_j seeks a trade-off between sparsity and grouping.

On the basis of the foregoing discussion, we solve Eq. (3) subject to Eq. (8) by solving its equivalent form through p subproblems:

$$\min \sum_{l=1}^L \left(\frac{1}{2(\sigma_j^{(l)})^2} \left\| \mathbf{x}_j^{(l)} - \mathcal{X}_{j-}^{(l)} (\mathbf{A}_{j,j-}^{(l)})^T \right\|^2 + n_l \log \sigma_j^{(l)} \right), \quad \text{subject to}$$

$$\sum_{l=1}^L \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)}) + \kappa_j \sum_{(l,s) \in \mathcal{L}} \sum_{k=1}^{j-1} P_\lambda(A_{j,k}^{(l)} - A_{j,k}^{(s)}) \leq t_j, j=1, \dots, p. \quad (9)$$

These p subproblems are of the same type; hence we only need to consider a general form. Let $\mathbf{Y}^{(l)}$ be a vector of length n_l , corresponding to $\mathbf{x}_j^{(l)}$ in Eq. (9), $\mathbf{X}^{(l)}$ be an n_l by m matrix, corresponding to $\mathcal{X}_{j-}^{(l)}$ with $m = j - 1$, and $\boldsymbol{\beta}^{(l)}$ be m -dimensional vector corresponding to

$\mathbf{A}_{j,j-}^{(l)}$. Then, a general form is, $\min \sum_{l=1}^L \left(\frac{1}{2\sigma_l^2} \|\mathbf{Y}^{(l)} - \mathbf{X}^{(l)} \boldsymbol{\beta}^{(l)}\|^2 + n_l \log \sigma_l \right)$, subject to

$$\sum_{l=1}^L \sum_{j=1}^m \min \left(\frac{|\beta_j^{(l)}|}{\lambda}, 1 \right) + \kappa \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \min \left(\frac{|\alpha_j^{(ls)}|}{\lambda}, 1 \right) \leq t. \quad (10)$$

where $\alpha_j^{(ls)} = \beta_j^{(l)} - \beta_j^{(s)}$, and $\zeta = (\beta_j^{(l)}_{j=1,\dots,m}, \alpha_j^{(ls)}_{j=1,\dots,m, (l,s) \in \mathcal{L}})$ is our new set of variables to be optimized. In addition, a new constraint $T\zeta = 0$ is imposed, namely, $\beta_j^{(l)} - \beta_j^{(s)} - \alpha_j^{(ls)} = 0$, for $j = 1, \dots, m, (l,s) \in \mathcal{L}$.

3. COMPUTATION

This section develops a computational method for nonconvex minimization in Eq. (10) through DC programming, augmented Lagrange multipliers and our pairwise coordinate descent algorithm.

3.1. Difference Convex Programming

For minimization in Eq. (10), we employ DC programming, which leads to a finite-step termination due to piecewise linearity of the TLP function [17]. Here P_λ can be decomposed into a difference of two convex functions:

$$P_\lambda(x) = \min \left(\frac{|x|}{\lambda}, 1 \right) = \frac{|x|}{\lambda} - \max \left(\frac{|x|}{\lambda} - 1, 0 \right). \quad (11)$$

This in turn yields a decomposition of the left-hand side of Eq. (10) into a difference of two convex part, that is,

$$S_1(\zeta) - S_2(\zeta) \leq t,$$

where

$$S_1(\zeta) = \sum_{l=1}^L \sum_{j=1}^m \frac{|\beta_j^{(l)}|}{\lambda} + \kappa \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \frac{|\alpha_j^{(ls)}|}{\lambda}, \quad (12)$$

$$S_2(\zeta) = \sum_{l=1}^L \sum_{j=1}^m \max \left(\frac{|\beta_j^{(l)}|}{\lambda} - 1, 0 \right) + \kappa \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \max \left(\frac{|\alpha_j^{(ls)}|}{\lambda} - 1, 0 \right). \quad (13)$$

This constructs a sequence of convex approximation sets that are contained in the original nonconvex set iteratively by replacing S_2 at iteration k with its affine majorization at iteration $k - 1$. At iteration k we minimize Eq. (10) subject to $T\zeta = 0$ and a relaxed constraint

$$\begin{aligned}
& \sum_{l=1}^L \sum_{j=1}^m |\beta_j^{(l)}| \mathbf{I}(|\hat{\beta}_j^{(l)[k-1]}| \leq \lambda) \\
& + \kappa \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m |\alpha_j^{(ls)}| \mathbf{I}(|\hat{\alpha}_j^{(ls)[k-1]}| \leq \lambda) \leq \lambda \left(t - \sum_{l=1}^L \sum_{j=1}^m \mathbf{I}(|\hat{\beta}_j^{(l)[k-1]}| > \lambda) \right. \\
& \left. - \kappa \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \mathbf{I}(|\hat{\alpha}_j^{(ls)[k-1]}| > \lambda) \right),
\end{aligned} \quad (14)$$

where $\hat{\beta}_j^{(l)[k-1]}$ is the estimate of $\beta_j^{(l)}$ at the $(k-1)$ th iteration, and $\hat{\alpha}_j^{(ls)[k-1]}$ is the estimate of $\alpha_j^{(ls)}$ at the $(k-1)$ th iteration.

3.2. Augmented Lagrange Multipliers

The constraint $T\zeta = 0$ defined by slack variables $\alpha_j^{(ls)}$ is treated through the augmented Lagrange multipliers, which is designed to convert a constrained problem to an unconstrained one. At iteration w , we minimize $S(\zeta)$ subject to Eq. (14)

$$\begin{aligned}
S(\zeta) = & \sum_{l=1}^L \left(\frac{1}{2\sigma^{(l)^2}} \|\mathbf{Y}^{(l)} - \mathbf{X}^{(l)}\beta^{(l)}\|^2 + n_l \log \sigma^{(l)} \right) \\
& + \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \tau_j^{(ls)[w]} (\beta_j^{(l)} \\
& - \beta_j^{(s)} - \alpha_j^{(ls)}) + \frac{1}{2} \sum_{(l,s) \in \mathcal{L}} \sum_{j=1}^m \nu_j^{(ls)[w]} (\beta_j^{(l)} - \beta_j^{(s)} - \alpha_j^{(ls)})^2,
\end{aligned} \quad (15)$$

where $\tau_j^{(ls)[w]}$ are Lagrangian multipliers for $T\zeta = 0$ and $\nu_j^{(ls)[w]}$ control the convergence speed of the algorithm. They are updated until convergence:

$$\tau_j^{(ls)[w+1]} = \tau_j^{(ls)[w]} + \mu_j^{(ls)[w]} (\beta_j^{(l)} - \beta_j^{(s)} - \alpha_j^{(ls)}), \mu_j^{(ls)[w+1]} = \rho \mu_j^{(ls)[w]},$$

where $\rho \in (1, 2)$ is pre-determined. At iteration k in the DC loop and at iteration w in the augmentation loop, we minimize Eq. (15) subject to Eq. (14). This weighted Lasso problem is solved by a pairwise coordinate descent algorithm that is introduced next.

3.3. Pairwise Coordinate Descent

A Lasso problem [19] solves

$$\min_{\beta_1, \dots, \beta_m} f(\beta), \text{ subject to } \sum_{j=1}^m |\beta_j| \leq t, \quad (16)$$

where f is a convex cost function.

For Eq. (16), coordinate descent methods are applicable to its regularization version [20, 21]. However, such a method breaks down for Eq. (16), as its solution may be trapped [20]. Here we develop a directional blockwise coordinate descent method, to solve Eq. (15) subject to Eq. (14). The main idea is to seek an optimal solution only over the simplex

boundary, where the solution lies. This directional search strategy overcomes the difficulty for an optimal solution to be trapped at the constraint boundary.

A solution of Eq. (16) exists when f is continuous since a feasible region of β is compact. For sparse learning, the Lasso constraint is active only when the feasible region defined by the constraint excludes all global minima of $f(\beta)$, in which any solution of Eq. (16) lies on its boundary. Instead of searching coordinatewise, we move along directions $\Delta|\beta_i| = -\Delta|\beta_j|$, to search over the boundary, where $\Delta|\beta_i|$ is the change in the absolute value of β_i after a step. Note that the pair of (i, j) is chosen so that the pair gives the steepest descent in the cost function value among all pairs.

The algorithm is summarized as follows.

Algorithm 1 Pairwise Coordinate Descent

Step 1. (Initialization) Input a initial value β^0 for β satisfying $\|\beta_0\|_1 = t$.

Step 2. (Iteration) Find the steeping direction satisfying $\Delta|\beta_i| = -\Delta|\beta_j|$. Perform an exact line search to determine the best step length and update.

Step 3. (Stopping rule) Terminate when the following subdifferential condition is satisfied: there exists $\lambda > 0$, s.t. $-\text{sign}$

$$(\beta_j) \frac{f(\beta)}{\beta_j} = \lambda \text{ for } \beta_j \neq 0, \text{ and } \left| \frac{f(\beta)}{\beta_j} \right| < \lambda \text{ for } \beta_j = 0.$$

Convergence of the algorithm is assured by Theorem 1.

THEOREM 1: For Eq. (16), stationary points and minimum points coincide. If the minimizer is unique, the algorithm converges to it.

If the cost function has nuisance parameters in addition to β , for example, $\sigma^{(l)}$, then we need to treat them as unconstrained coordinates and update them coordinately in every iteration.

A DC loop and an augmentation loop converge in only a few steps in practice [17]. Taking advantage of Algorithm 3.3 in the inner loop, our method is capable of treating multiple graphs of over thousands of variables in real time. This is desirable for our nonconvex minimization problem.

4. NUMERICAL EXAMPLES

This section examines operating characteristics of the proposed method, and contrasts it against its competitors through simulations. In particular, the proposed method, denoted by “nonconvex,” is examined together with its convex counterpart—our method with the L_1 -function replacing the TLP, denoted by ‘convex’, and a sparse L_1 method from Ref [17] for DAGs individually, denoted by ‘DAGlasso’.

In simulations, two DAGs are considered and a complete set $\mathcal{L} = \{(1, 2)\}$ is used for possible grouping. All simulations are performed in R. Performance metrics for sparsity and grouping pursuit are the number of false positives (FP), the number of false negatives (FN), the number of correctly identified differences between graphs (TD), and the number of falsely identified differences between graphs (FD). Overall, we use the Matthews Correlation Coefficient (MCC) [22] as a performance metric, which is commonly used in binary classification, and is defined as,

$$\text{MMC} = \frac{(\text{TP} \times \text{TN}) - (\text{FP} \times \text{FN})}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \quad (17)$$

where TP, TN, FP, and FN correspond to true positives, true negatives, FP and FN, respectively. A larger value of MCC gives better a fit with 1 being the best and -1 being the worst.

Two graphs are generated as follows. The first DAG $\mathcal{G}_1$ is generated at random, with the number of nodes $p = 50, 100, 200$, having the average probability of connecting one node to another with higher ordering: 0.02. The second DAG $\mathcal{G}_2$ is obtained by removing a number of edges from $\mathcal{G}_1$ and this number is controlled to be less than 1% of the total edges in $\mathcal{G}_1$. For $p = 50$, the two DAGs are just identical. For $p = 100, 200$, removal is done by deleting all edges connecting from a specific node, which mimics a situation when a certain gene from $\mathcal{G}_1$ becomes isolated due to some treatment effect. Finally, we generate a random sample from $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively, with equal sample size $n_1 = n_2 = n$.

For the proposed method, there are three data-dependent tuning parameters (λ, κ, t) in Eq. (10). It has been shown in Ref [17] that estimation based on the TLP is not sensitive to the choice of λ as long as λ is small enough. A common set of values for λ is $\{0.1, 0.01, 0.001\}$, and κ is a tuning parameter ranged between 0 and 1, with three to five values, based on our limited numerical experience. This makes tuning three parameters much easier with an effort focused on tuning t . In this simulation study, tuning parameters for all methods are optimized for prediction over an independent data set of size 1000 for each graph.

As suggested in Table 1, the proposed method compares favorably against its competitors across all the situations, in terms of the MCCs, accuracy of estimation, and detection of a structural change. Interestingly, seeking common structures or detection of structural changes is more critical for multiple graphs than a single graph, especially when they share some common structures. In addition, our nonconvex method improves significantly over its convex counterpart. In most cases, our nonconvex method yields a MMC value close to 1, suggesting almost perfect learning of structures. In this sense, a nonconvex method is useful in estimating directed graphs.

5. ANALYSIS OF CELL SIGNALING DATA

This section applies the proposed method to analyze multivariate flow cytometry data in Ref [23]. Data were collected after a series of stimulatory cues and inhibitory interventions, with cell reactions stopped at 15 minutes after stimulation by fixation, to profile the effects of each condition on the intracellular signaling networks of human primary naive CD4+ T cells. Over 10,000 flow cytometry measurements were made over 11 phosphorylated proteins and phospholipids from 14 experimental conditions. The main purpose of this experiment is to infer casual influences in cellular signaling networks through perturbations with molecular interventions. The simultaneous measurements permit multivariate as opposed to univariate approaches. Data sets were available; see Ref [23] for more details. The DAG representation of the network is displayed in Fig. 1 [23]. A direction from node $\mathbf{X}$ to node $\mathbf{Y}$ is interpreted as a causal influence from $\mathbf{X}$ to $\mathbf{Y}$.

A DAG model was fit in Ref [23] with data from the first nine conditions, following called Group 1, whereas the rest five conditions, denoted as Group 2 were not used for estimation. Note that all five conditions in Group 2 employed ICAM-2, a general perturbation, in addition to perturbations used in Group 1. In our analysis, we are interested in whether the usage of ICAM-2 in Group 2 activate or inhibit some causal relationship in the network.

Thus, data have been split into two datasets: 7466 samples from Group 1 and 4206 samples from Group 2. We fit a two-DAG model with 1/10th samples for training and 9/10th for tuning. The graphical result of reconstructed networks are displayed in Fig. 2, where the DAGlasso [10] based on individual graphs is used for comparison. Correct edges are marked with solid arrows, while FP are indicated by long dashed arrows.

Our method is more reliable in that it gives much fewer FP with almost the same number of TP as compared to the DAGlasso, and identical links and differences between the two groups are likely to be real, which may be cross-validated experimentally. By comparison the two graph reconstructed by the DAGlasso are not so consistent and the estimated differences are mainly false discoveries.

In the analysis, several known links were missed by the proposed method and the DAGlasso method. This is because many causal relations in protein-signaling networks are believed to be non-linear, which may not be detectable by methods based on linear models, such as our method and the DAGlasso. In fact, to our knowledge, no linear methods could reconstruct most links. This non-linearity goes beyond the scope of the linear casual effect specified in Equation (1).

Acknowledgments

Research supported in part by NSF grant DMS-1207771, NIH grants R02GM08153 and R01HL105397. The authors would like to thank reviewers and editors for helpful comments and suggestions.

APPENDIX

Proof of Theorem 1: Let β^m be the solution at iteration m . Convergence follows directly from the fact that $f(\beta^m)$ is decreasing in m . To prove the limit of β^* satisfies the

subdifferential condition, that is, there exists $\lambda > 0$, s.t. $-\text{sign}(\beta_j) \frac{\partial f(\beta)}{\partial \beta_j} = \lambda$ for $\beta_j \neq 0$, and

$\left| \frac{\partial f(\beta)}{\partial \beta_j} \right| < \lambda$ for $\beta_j = 0$, suppose the subdifferential condition does not hold for the

convergence point β^* . Let $i = \arg\max_{k \in \{1, \dots, p\}} \left| \frac{\partial f(\beta^*)}{\partial \beta_k} \right|$, $j = \arg\min_{\{k \mid \beta_k^* \neq 0\}} \left| \frac{\partial f(\beta^*)}{\partial \beta_k} \right|$. Then

$\left| \frac{\partial f(\beta^*)}{\partial \beta_i} \right| > \left| \frac{\partial f(\beta^*)}{\partial \beta_j} \right|$. This implies that we can further reduce the value of the objection function, which contradicts to convergence. From Ref [24], this subdifferential condition is sufficient and necessary for β^* to be the minimizer of the Lasso. This completes the proof.

REFERENCES

1. Brown L, Tsamardinos I, Aliferis C. A comparison of novel and state-of-the-art polynomial Bayesian network learning algorithms. Proceedings of the 20th National Conference on Artificial Intelligence (AAAI). 2005:739–745.
2. Heckerman D, Geiger D, Chickering D. Learning Bayesian networks: the combination of knowledge and statistical data. Mach Learn. 1995; 20(3):197–243.
3. Chickering D. Optimal structure identification with greedy search. J Mach Learn Res. 2003; 3:507–554.
4. Spiegelhalter D, Dawid A, Lauritzen S, Cowell R. Bayesian analysis in expert systems. Stat Sci. 1993; 8(3):219–247.
5. Cheng J, Greiner R, Kelly J, Bell D, Liu W. Learning Bayesian networks from data: an information-theory based approach. Artif Intell. 2002; 137(1–2):43–90.

6. Tsamardinos I, Brown LE, Aliferis CF. The max-min hill-climbing Bayesian network structure learning algorithm. *Mach Learn*. 2006; 65:31–78.
7. Spirtes, P.; Glymour, C.; Scheines, R. Causation, Prediction, and Search. Vol. Vol. 81. Cambridge: The MIT Press; 2000.
8. Kalisch M, Bühlmann P. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *J Mach Learn Res*. 2007; 8:613–636.
9. Gillispie, S.; Perlman, M. Enumerating Markov equivalence classes of acyclic digraph models. In: Breese, J.; Koller, D., editors. *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*. San Francisco: Morgan Kaufmann; 2001. p. 171-177.
10. Shojaie A, Michailidis G. Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*. 2010; 97:519–538. [PubMed: 22434937]
11. Zhou S, Lafferty J, Wasserman L. Time varying undirected graphs. *Arxiv preprint arXiv: 0802.2758*. 2008
12. Kolar M, Song L, Ahmed A, Xing E. Estimating time-varying networks. *Ann Appl Stat*. 2010; 4(1):94–123.
13. Guo J, Levina E, Michailidis G, Zhu J. Joint estimation of multiple graphical models. *Biometrika*. 2011; 98(1):1. [PubMed: 23049124]
14. Meinshausen N, Bühlmann P. High-dimensional graphs and variable selection with the lasso. *Ann Stat*. 2006; 34(3):1436–1462.
15. Yuan M, Lin Y. Model selection and estimation in the Gaussian graphical model. *Biometrika*. 2007; 94(1):19.
16. Friedman J, Hastie T, Tibshirani R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*. 2008; 9(3):432. [PubMed: 18079126]
17. Shen X, Pan W, Zhu Y. Likelihood-based selection and sharp parameter estimation. *J Am Stat Assoc*. 2012; 107:223–232. [PubMed: 22736876]
18. Pearl, J. Reasoning and Inference. London: Cambridge University Press; 2000. Causality: Models.
19. Tibshirani R. Regression shrinkage and selection via the lasso. *J Roy Stat Soc Ser B (Methodological)*. 1996; 32(2):267–288.
20. Friedman J, Hastie T, Tibshirani R. Pathwise coordinate optimization. *Ann Appl Stat*. 2007; 1(2):302–332.
21. Wu T, Lange K. Coordinate descent algorithms for lasso penalized regression. *Ann Appl Stat*. 2008; 2(1):224–244.
22. Baldi P, Brunak S, Chauvin Y, Andersen C, Nielsen H. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*. 2000; 16(5):412–424. [PubMed: 10871264]
23. Sachs K, Perez O, Pe'er D, Lauffenburger D, Nolan G. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*. 2005; 308(5721):523. [PubMed: 15845847]
24. Osborne M, Presnell B, Turlach B. On the lasso and its dual. *J Comput Graph Stat*. 2000; 9(2): 319–337.

Fig. 1.
DAG representation for 11 proteins.

Fig. 2. Analysis of cell signaling data: Correct edges are marked with solid arrows, while FP are indicated by long dashes. (a) Group 1 by nonconvex. (b) Group 2 by nonconvex. (c) Group 1 by DAGlasso. (d) Group 2 by DAGlasso.

Table 1

Estimated quantities and their corresponding estimated standard errors (in parentheses) based on 100 simulation replications.

<i>n</i>	<i>p</i>	method	FP	FN	TD	FD	MCC
50	50	DAGlasso	15.7(3.9)	0.2(0.4)	0(0)	15.3(3.8)	0.84(0.03)
		Convex	3.9(2.7)	0(0)	0(0)	0.3(0.5)	0.95(0.03)
		Nonconvex	0.3(0.8)	0.1(0.3)	0(0)	0(0.2)	0.99(0.01)
100	100	DAGlasso	73.7(8.6)	1.1(1)	2(0)	72.3(8.2)	0.83(0.02)
		Convex	18.9(6.2)	0.4(0.9)	1.7(0.5)	0.9(1)	0.95(0.02)
		Nonconvex	4.7(3.5)	1.6(2)	1.6(0.6)	0.2(0.5)	0.98(0.02)
200	200	DAGlasso	406.5(21.7)	14.5(3.8)	4.9(0.3)	402.1(20.8)	0.80(0.01)
		Convex	122.3(15.3)	46.7(9.2)	2.5(1)	1.4(1.2)	0.90(0.01)
		Nonconvex	30.1(11.2)	32.9(9.5)	3.2(1.2)	0.6(0.9)	0.96(0.01)
100	50	DAGlasso	6.7(2.6)	0(0)	0(0)	6.6(2.6)	0.92(0.03)
		Convex	1.2(1.5)	0(0)	0(0)	0.1(0.3)	0.98(0.02)
		Nonconvex	0(0.3)	0(0)	0(0)	0(0.2)	1.00(0.01)
100	100	DAGlasso	32.3(5.9)	0(0.1)	2(0)	31.8(6)	0.92(0.01)
		Convex	6.3(3.2)	0(0)	1.9(0.3)	0.5(0.7)	0.98(0.01)
		Nonconvex	1(1.3)	0(0)	2(0.1)	0.1(0.3)	1.00(0.01)
200	200	DAGlasso	162.7(14.0)	0.1(0.3)	5(0)	158.2(13.7)	0.91(0.01)
		Convex	28.9(7.1)	0.1(0.4)	4.5(0.7)	1(0.9)	0.98(0.01)
		Nonconvex	3.7(3.2)	1(1.8)	4.8(0.4)	0.3(0.7)	1.00(0.01)

Notes: For $p = 50$, there are 20 edges in $\mathcal{G}_1$ and 20 edges in $\mathcal{G}_2$ with no differences. For $p = 100$, there are 88 edges in $\mathcal{G}_1$ and 86 edges in $\mathcal{G}_2$ with two differences. For $p = 200$, there are 427 edges in $\mathcal{G}_1$ and 422 edges in $\mathcal{G}_2$ with five differences.