Efficient Proximal Gradient Algorithm for Inference of Differential Gene Networks

Chen Wang[1], Feng Gao[1], Georgios B. Giannakis[2], Gennaro D'Urso[3] and Xiaodong Cai[1,4*]

[1]Department of Electrical and Computer Engineering, University of Miami, Coral Gables, Florida 33146, United States; [2]Department of Electrical and Computer Engineering, University of Minnesota, 55455 Minneapolis,MN; [3]Department of Molecular and Cellular Pharmacology, University of Miami, 33136 Miami, FL; [4]Sylvester Comprehensive Cancer Center, University of Miami, Miami, Florida 33136, United States

*x.cai@miami.edu

Abstract

Background. Gene networks in living cells can change depending on various conditions such as caused by different environments, tissue types, disease states, and development stages. Identifying the differential changes in gene networks is very important to understand molecular basis of various biological process. While existing algorithms can be used to infer two gene networks separately from gene expression data under two different conditions, and then to identify network changes, such an approach does not exploit the data jointly, and it is thus suboptimal. A desirable approach would be clearly to infer two gene networks jointly, which can yield improved estimates of network changes.

Results. In this paper, we developed a proximal gradient algorithm for differential network (ProGAdNet) inference, that jointly infers two gene networks under different conditions and then identifies changes in the network structure. Computer simulations demonstrated that our ProGAdNet outperformed existing algorithms in terms of inference accuracy, and was much faster than a similar approach for joint inference of gene networks. Gene expression data of breast tumors and normal tissues in the TCGA database were analyzed with our ProGAdNet, and revealed that 268 genes were involved in the changed network edges. Gene set enrichment analysis of this set of 268 genes identified a number of gene sets related to breast cancer or other types of cancer, which corroborated the gene set identified by ProGAdNet was very informative about the cancer disease status. A software package implementing the ProGAdNet and computer simulations is available upon request.

Conclusion. With its superior performance over existing algorithms, ProGAdNet provides a valuable tool for finding changes in gene networks, which may aid the discovery of gene-gene interactions changed under different conditions.

Keywords

Gene network; differential network; proximal gradient method

Background

Genes in living cells interact and form a complex network to regulate molecular functions and biological processes. Gene networks can undergo toplogical changes depending on the molecular context in which they operate [1, 2]. For example, it was observed that transcription factors (TFs) can bind to and thus regulate different target genes under varying environmental conditions [3, 4]. Changes of genetic interactions when cells are challenged by DNA damage as observed in [5] may also reflect the structural changes of the underlying gene network. This kind of rewiring of gene networks has been observed not only in yeast [3–6], but also in mammalian cells [7, 8]. More generally, differential changes of gene networks can occur depending on environment, tissue type, disease state, development and speciation [1]. Therefore, identification of such differential changes in gene networks is of paramount importance when it comes to understanding the

molecular basis of various biological processes.

Although a number of computational methods have been developed to infer the structure of gene regulatory networks from gene expression and related data [9–12], they are mainly concerned with the static structure of gene networks under a single condition. These methods rely on similarity measures such as the correlation or mutual information [13, 14], Gaussian graphical models (GGMs) [15, 16], Bayesian networks [17, 18], or linear regression models [19–22]. Existing methods for the analysis of *differential* gene interactions under different conditions typically attempt to identify differential co-expression of genes based on correlations between their expression levels [23]. While it is possible to use an existing method to infer a gene network under different conditions separately, and then compare the inferred networks to determine their changes, such an approach does not jointly leverage the data under different conditions in the inference; thus, it may markedly sacrifice the accuracy in the inference of network changes.

In this paper, we develop a very efficient proximal gradient algorithm for differential network (ProGAdNet) inference, that jointly infers gene networks under two different conditions and then identifies changes in these two networks. To overcome the challenge of the small sample size and a large number of unknowns, which is common to inference of gene networks, our method exploits two important attributes of gene networks: i) sparsity in the underlying connectivity, meaning that the number of gene-gene interactions in a gene network is much smaller than the number of all possible interactions [19, 24–26]; and, ii) sparsity in the structural changes, meaning that the number of interactions changed in response to different conditions is much smaller than the total number of interactions present in the network. A similar network inference setup was considered in [27] for inferring multiple gene networks, but no new algorithm was developed there; instead [27] adopted the lqa algorithm of [28] that was designed for generalized linear models. Our computer simulations demonstrated superior performance of our ProGAdNet algorithm relative to existing methods including the lqa algorithm. Analysis of a set of RNA-Seq data from normal tissues and breast tumors with ProGAdNet identified genes involved in changes of the gene network.

The differential gene-gene interactions identified by our ProGAdNet algorithm yield a list of genes alternative to the list of differentially expressed genes. This may provide additional insight into the molecular mechanism behind the phenotypical difference of the tissue under different conditions. Alternatively, the two gene networks inferred by our ProGAdNet algorithm can be used for further differential network analysis (DiNA). DiNA has received much attention recently; the performance of ten DiNA algorithms was assessed in [29] using gene networks and gene/microRNA networks. Given two networks with the same set of nodes, a DiNA algorithm computes a score for each node based on the difference of global and/or local topologies of the two networks, and then ranks nodes based on these scores. Apparently, DiNA relies on the two networks that typically need to be constructed from certain data. Our ProGAdNet algorithm provides an efficient and effective tool for constructing two gene networks of the same set of genes from gene expression data under two different conditions, which can be used by a DiNA algorithm for further analysis.

Methods

Gene Network Model

Suppose that expression levels of p genes have been measured with microarray or RNA-seq, and let X_i be the expression level of the ith gene, where $i = 1, \ldots, p$. To identify the possible regulatory effect of other genes on the ith gene, we employ the following linear regression model as also used in [19–22]

$$X_i = \sum_{j=1, j\neq i}^{p} X_j b_{ji} + E_i, \tag{1}$$

where E_i is the error term, and unknown regression coefficients (b_{ji})'s reflect the correlation between X_i and X_j after adjusting the effects of other variables, X_k's, $k \notin \{i, j\}$. This adjusted correlation may be the result of possible interaction between genes i and j. The nonzero (b_{ji})'s define the edges in the gene network. Suppose

that n samples of gene expression levels of the same organism (or the same type of tissue of an organism) under two different conditions are available, and let $n \times 1$ vectors $\mathbf{x}_i$ and $\tilde{\mathbf{x}}_i$ contain these n samples of the ith gene under two conditions, respectively. Define $n \times p$ matrices $\mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p]$ and $\tilde{\mathbf{X}} := [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_p]$, and $p \times p$ matrices $\mathbf{B}$ and $\tilde{\mathbf{B}}$ whose element on the ith column and the jth row are b_{ji} and $\tilde{b}_{ji}$, respectively. Letting $b_{ii} = \tilde{b}_{ii} = 0$, model (1) yields the following

$$\begin{aligned} \mathbf{X} &= \mathbf{X}\mathbf{B} + \mathbf{E} \\ \tilde{\mathbf{X}} &= \tilde{\mathbf{X}}\tilde{\mathbf{B}} + \tilde{\mathbf{E}}, \end{aligned} \tag{2}$$

where $n \times p$ matrices $\mathbf{E}$ and $\tilde{\mathbf{E}}$ contain error terms. Matrices $\mathbf{B}$ and $\tilde{\mathbf{B}}$ characterize the structure of the gene networks under two conditions.

Our main goal is to identify the changes in the gene network under two conditions, namely, those edges from gene j to gene i such that $b_{ji} - \tilde{b}_{ji} \neq 0$, $j \neq i$. One straightforward way to do this is to estimate $\mathbf{B}$ and $\tilde{\mathbf{B}}$ separately from two linear models in (2), and then find gene pairs (i, j) for which $b_{ji} - \tilde{b}_{ji} \neq 0$. However, this approach may not be optimal, since it does not exploit the fact that the network structure does not change significantly under two conditions, that is, most entries of $\mathbf{B}$ and $\tilde{\mathbf{B}}$ are identical. A better approach is apparently to infer gene networks under two conditions jointly, which can exploit the similarity between two network structures and thereby improve the inference accuracy.

If we denote the ith column of $\mathbf{B}$ and $\tilde{\mathbf{B}}$ as $\mathbf{b}_i$ and $\tilde{\mathbf{b}}_i$, we can also write model (2) for each gene separately as follows: $\mathbf{x}_i = \mathbf{X}\mathbf{b}_i + \mathbf{e}_i$ and $\tilde{\mathbf{x}}_i = \tilde{\mathbf{X}}\tilde{\mathbf{b}}_i + \tilde{\mathbf{e}}_i$, $i = 1, \ldots, p$, where $\mathbf{e}_i$ and $\tilde{\mathbf{e}}_i$ are the ith column of $\mathbf{E}$ and $\tilde{\mathbf{E}}$, respectively. To remove the constraints $b_{ii} = 0$, $i = 1, \ldots, p$, we define matrices $\mathbf{X}_{-i} := [\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_p]$ and $\tilde{\mathbf{X}}_{-i} := [\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_{i-1}, \tilde{\mathbf{x}}_{i+1}, \ldots, \tilde{\mathbf{x}}_p]$, vectors $\beta_i := [b_{1i}, \ldots, b_{(i-1)i}, b_{(i+1)i}, \ldots, b_{pi}]^T$ and $\tilde{\beta}_i := [\tilde{b}_{1i}, \ldots, \tilde{b}_{(i-1)i}, \tilde{b}_{(i+1)i}, \ldots, \tilde{b}_{pi}]^T$. The regression model for the gene network under two conditions can be written as

$$\begin{aligned} \mathbf{x}_i &= \mathbf{X}_{-i}\beta_i + \mathbf{e}_i \\ \tilde{\mathbf{x}}_i &= \tilde{\mathbf{X}}_{-i}\tilde{\beta}_i + \tilde{\mathbf{e}}_i, \; i = 1, \ldots, p. \end{aligned} \tag{3}$$

Based on (3), we will develop a proximal gradient algorithm to infer β_i and $\tilde{\beta}_i$ jointly, and identify changes in the network structure.

Network inference

Optimization Formulation

As argued in [30–32], gene regulatory networks or more general biochemical networks are sparse, meaning that a gene directly regulates or is regulated by a small number of genes relative to the total number of genes in the network. Taking into account sparsity, only a relatively small number of entries of $\mathbf{B}$ and $\tilde{\mathbf{B}}$, or equivalently entries of β_i and $\tilde{\beta}_i$, $i = 1, \ldots, p$, are nonzero. These nonzero entries determine the network structure and the regulatory effect of one gene on other genes. As mentioned earlier, the gene network of an organism is expected to have similar structure under two different conditions. For example, the gene network of a tissue in a disease (such as cancer) state may have changed, comparing to that of the same tissue under the normal condition, but such change in the network structure is expected to be small relative to the overall network structure. Therefore, it is reasonable to expect that the number of edges that change under two conditions is small comparing with the total number of edges of the network.

Taking into account sparsity in the network and also in the changes of the network under two conditions, we formulate the following optimization problem to jointly infer gene networks under two conditions:

$$\begin{aligned} (\hat{\beta}_i, \hat{\tilde{\beta}}_i) = &\arg\min_{\beta_i, \tilde{\beta}_i} \{ \| \mathbf{x}_i - \mathbf{X}_{-i}\beta_i \|^2 \\ &+ \| \tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\tilde{\beta}_i \|^2 + \lambda_1 (\| \beta_i \|_1 + \| \tilde{\beta}_i \|_1) \\ &+ \lambda_2 \| \beta_i - \tilde{\beta}_i \|_1 \}, \end{aligned} \tag{4}$$

where $\|\cdot\|$ stands for Euclidean norm, $\|\cdot\|_1$ stands for l_1 norm, and λ_1 and λ_2 are two positive constants. The objective function in (4) consists of the squared error of the linear regression model (1) and two regularization terms $\lambda_1(\|\beta_i\|_1 + \|\tilde{\beta}_i\|_1)$ and $\lambda_2\|\beta_i - \tilde{\beta}_i\|_1$. Note that unlike the GGM, the regularized least squared error approach here does not rely on the Gaussian assumption. The two regularization terms induce sparsity in the inferred networks and network changes, respectively. This optimization problem is apparently convex, and therefore it has a unique and globally optimal solution. Note that the term $\lambda_2\|\beta_i - \tilde{\beta}_i\|_1$ is reminiscent of the fused Lasso [33]. However, all regression coefficients in the fused Lasso are essentially coupled, whereas here the term $\lambda_2\|\beta_i - \tilde{\beta}_i\|_1$ only couples each pair of regression coefficients, β_{ij} and $\tilde{\beta}_{ij}$. As will be described next, this enables us to develop an algorithm to solve optimization problem (4) that is different from and more efficient than the algorithm for solving the general fused Lasso problem. Note that an optimization problem similar to (4) was formulated in [27] for inferring multiple gene networks, but no new algorithm was developed, instead the problem was solved with the lqa algorithm [28] that was developed for general penalized maximum likelihood inference of generalized linear models including the fused Lasso. Our computer simulations showed that our algorithm not only is much faster than the lqa algorithm, but also yields much more accurate results.

Proximal Gradient Solver

Define $\alpha_i := [\beta_i^T\ \tilde{\beta}_i^T]^T$, and let us separate the objective function in (4) into the differentiable part $g_1(\alpha_i)$ and the non-differentiable part $g_2(\alpha_i)$ given by

$$\begin{aligned} g_1(\alpha_i) &= \|\mathbf{x}_i - \mathbf{X}_{-i}\beta_i\|^2 + \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\tilde{\beta}_i\|^2, \\ g_2(\alpha_i) &= \lambda_1(\|\beta_i\|_1 + \|\tilde{\beta}_i\|_1) + \lambda_2\|\beta_i - \tilde{\beta}_i\|_1 . \end{aligned} \tag{5}$$

Applying the proximal gradient method [34] to solve the optimization problem (4), we obtain an expression for α_i in the rth step of the iterative procedure as follows:

$$\alpha_i^{(r+1)} = \mathrm{prox}_{\lambda^{(r)} g_2}[\alpha_i - \lambda^{(r)}\nabla g_1(\alpha_i)], \tag{6}$$

where prox stands for the proximal operator defined as $\mathrm{prox}_{\lambda f}(\mathbf{t}) := \arg\min_{\mathbf{x}} f(\mathbf{x}) + \frac{1}{2\lambda}||\mathbf{x}-\mathbf{t}||^2$ for function $f(\mathbf{x})$ and a constant vector $\mathbf{t}$, and $\nabla g_1(\alpha_i)$ is the gradient of $g_1(\alpha_i)$. Generally, the value of step size $\lambda^{(r)}$ can be found using a line search step, which can be determined from the Lipschitz constant [34]. For our problem, we will provide a closed-form expression for $\lambda^{(r)}$ later. Since $g_1(\alpha_i)$ is simply in a quadratic form, its gradient can be obtained readily as $\nabla g_1(\alpha_i) = [\nabla g_1(\beta_i)^T, \nabla g_1(\tilde{\beta}_i)^T]^T$, where $\nabla g_1(\beta_i) = 2(\mathbf{X}_{-i}^T\mathbf{X}_{-i}\beta_i - \mathbf{X}_{-i}^T\mathbf{x}_i)$ and $\nabla g_1(\tilde{\beta}_i) = 2(\tilde{\mathbf{X}}_{-i}^T\tilde{\mathbf{X}}_{-i}\tilde{\beta}_i - \tilde{\mathbf{X}}_{-i}^T\tilde{\mathbf{x}}_i)$.

Upon defining $\mathbf{t} = \beta_i - \lambda^{(r)}\nabla g_1(\beta_i)$ and $\tilde{\mathbf{t}} = \tilde{\beta}_i - \lambda^{(r)}\nabla g_1(\tilde{\beta}_i)$, the proximal operator in (6) can be written as

$$\begin{aligned} \mathrm{prox}_{\lambda^{(r)} g_2}(\mathbf{t}) = \arg\min_{\beta_i,\tilde{\beta}_i} \{&\lambda_1(\|\beta_i\|_1 + \|\tilde{\beta}_i\|_1) + \lambda_2\|\beta_i - \tilde{\beta}_i\|_1 \\ &+ \frac{1}{2\lambda^{(r)}}(\|\beta_i - \mathbf{t}\|^2 + \|\tilde{\beta}_i - \tilde{\mathbf{t}}\|^2)\}. \end{aligned} \tag{7}$$

It is seen that the optimization problem in proximal operator (7) can be decomposed into $p-1$ separate problems as follows

$$\begin{aligned} \arg\min_{\beta_{ij},\tilde{\beta}_{ij}} \{&\lambda_1(|\beta_{ij}| + |\tilde{\beta}_{ij}|) + \lambda_2|\beta_{ij} - \tilde{\beta}_{ij}| \\ &+ \frac{1}{2\lambda^{(r)}}\left((\beta_{ij} - t_j)^2 + (\tilde{\beta}_{ij} - \tilde{t}_j)^2\right)\} \\ &j = 1, \ldots, p-1, \end{aligned} \tag{8}$$

where β_{ij} and $\tilde{\beta}_{ij}$ are the jth element of β_i and $\tilde{\beta}_i$, respectively, and t_j and $\tilde{t}_j$ are the jth element of $\mathbf{t}$ and $\tilde{\mathbf{t}}$, respectively. The optimization problem (8) is in the form of the fused Lasso signal approximator (FLSA) [35]. The general FLSA problem has many variables, and numerical optimization algorithms were developed to solve the FLSA problem [35, 36]. However, our problem has only two variables, which enables us to find the solution of (8) in closed form. This is then used in each step of our proximal gradient algorithm for network inference.

Let us define a soft-thresholding operator $S(x,a)$ as follows

$$S(x,a) = \begin{cases} x-a, \text{ if } x > a \\ x+a, \text{ if } x < -a \\ 0, \text{ otherwise,} \end{cases} \tag{9}$$

where a is a positive constant. Then as shown in [35], if the solution of (8) at $\lambda_1 = 0$ is $\hat{\beta}_{ij}^{(0)}$ and $\hat{\tilde{\beta}}_{ij}^{(0)}$, the solution of (8) at $\lambda_1 > 0$ is given by

$$\begin{aligned} \hat{\beta}_{ij} &= S(\hat{\beta}_{ij}^{(0)} - \tilde{\lambda}_1) \\ \hat{\tilde{\beta}}_{ij} &= S(\hat{\tilde{\beta}}_{ij}^{(0)}, \tilde{\lambda}_1), \end{aligned} \tag{10}$$

where $\tilde{\lambda}_1 = \lambda_1 \lambda^{(r)}$. Therefore, if we can solve the problem (8) at $\lambda_1 = 0$, we can find the solution of (8) at any $\lambda_1 > 0$ from (10). It turns out that the solution of (8) at $\lambda_1 = 0$ can be found as

$$\left(\hat{\beta}_{ij}^{(0)}, \hat{\tilde{\beta}}_{ij}^{(0)}\right) = \begin{cases} \left(\frac{t_j+\tilde{t}_j}{2}, \frac{t_j+\tilde{t}_j}{2}\right), \text{ if } |t_j - \tilde{t}_j| \le 2\tilde{\lambda}_2 \\ (t_j - \tilde{\lambda}_2, \tilde{t}_j + \tilde{\lambda}_2), \text{ if } t_j - \tilde{t}_j > 2\tilde{\lambda}_2 \\ (t_j + \tilde{\lambda}_2, \tilde{t}_j - \tilde{\lambda}_2), \text{ otherwise,} \end{cases} \tag{11}$$

where $\tilde{\lambda}_2 = \lambda_2 \lambda^{(r)}$. Therefore, our proximal gradient method can solve the network inference problem (6) efficiently through an iterative process, where each step of the iteration solves the optimization problem (6) in closed form specified by (10) and (11). To obtain a complete proximal gradient algorithm, we need to find the step size $\lambda^{(r)}$ as will be described next.

Stepsize

As mentioned in [34], if the step size $\lambda^{(r)} \in [0, 1/L]$, where L is the Lipschitz constant of $\nabla g_1(\alpha_i)$, then the proximal gradient algorithm converges to yield the optimal solution. We next derive an expression for L. Specifically, we need to find L such that $\| \nabla g_1(\alpha_i^{(1)}) - \nabla g_1(\alpha_i^{(2)}) \|_2 \le L \| (\alpha_i^{(1)} - \alpha_i^{(2)}) \|_2$ for any $\alpha_i^{(1)} \neq \alpha_i^{(2)}$, which is equivalent to

$$2 \left\| \begin{matrix} \mathbf{X}_{-i}^T \mathbf{X}_{-i} (\beta_i^{(1)} - \beta_i^{(2)}) \\ \tilde{\mathbf{X}}_{-i}^T \tilde{\mathbf{X}}_{-i} (\tilde{\beta}_i^{(1)} - \tilde{\beta}_i^{(2)}) \end{matrix} \right\| \le L \left\| \begin{matrix} \beta_i^{(1)} - \beta_i^{(2)} \\ \tilde{\beta}_i^{(1)} - \tilde{\beta}_i^{(2)} \end{matrix} \right\| \tag{12}$$

for any $(\beta_i^{(1)}, \tilde{\beta}_i^{(1)}) \neq (\beta_i^{(2)}, \tilde{\beta}_i^{(2)})$. Let γ and $\tilde{\gamma}$ be the maximum eigenvalues of $\mathbf{X}_{-i}^T \mathbf{X}_{-i}$ and $\tilde{\mathbf{X}}_{-i}^T \tilde{\mathbf{X}}_{-i}$, respectively. It is not difficult to see that (12) will be satisfied if $L = 2(\gamma + \tilde{\gamma})$. Note that $\mathbf{X}_{-i}^T \mathbf{X}_{-i}$ and $\mathbf{X}_{-i} \mathbf{X}_{-i}^T$ have the same set of eigenvalues. And thus, γ can be found using a numerical algorithm with a computational complexity of $O((\min(n,p))^2)$. After obtaining L, the step size of our proximal gradient algorithm can be chosen to be $\lambda^{(r)} = 1/L$. Note that $\lambda^{(r)}$ does not change across iterations, and it only needs to be computed once. Since the sum of the eigenvalues of a matrix is equal to the trace of matrix, another possible value for L is $2(\text{trace}(\mathbf{X}_{-i}^T \mathbf{X}_{-i}) + \text{trace}(\tilde{\mathbf{X}}_{-i}^T \tilde{\mathbf{X}}_{-i}))$, which can save the cost of computing γ and $\tilde{\gamma}$. However, this value of L is apparently greater than $2(\gamma + \tilde{\gamma})$, which reduces the step size $\lambda^{(r)}$, and may affect the convergence speed of the algorithm.

Algorithm

The proximal gradient solver of (4) for inference of differential gene networks is abbreviated as ProGAdNet, and is summarized in the following table.

Algorithm 1 ProGAdNet algorithm for solving optimization problem (4): proxg($\mathbf{X}$, $\tilde{\mathbf{X}}$, λ_1, λ_2)

Input data $\mathbf{X}$ and $\tilde{\mathbf{X}}$, and parameters λ_1 and λ_2
Compute the maximum eigenvalues of $\mathbf{X}_{-i}\mathbf{X}_{-i}^T$ and $\tilde{\mathbf{X}}_{-i}\tilde{\mathbf{X}}_{-i}^T$,
  γ and $\tilde{\gamma}$, respectively; set step size $\lambda^{(r)} = 1/[2(\gamma+\tilde{\gamma})]$.
Set initial values of β_i and $\tilde{\beta}_i$
repeat
  Compute $\nabla g_1(\beta_i) = 2(\mathbf{X}_{-i}^T\mathbf{X}_{-i}\beta_i - \mathbf{X}_{-i}^T\mathbf{x}_i)$ and
    $\nabla g_1(\tilde{\beta}_i) = 2(\tilde{\mathbf{X}}_{-i}^T\tilde{\mathbf{X}}_{-i}\tilde{\beta}_i - \tilde{\mathbf{X}}_{-i}^T\tilde{\mathbf{x}}_i)$.
  Compute $\mathbf{t} = \beta_i - \lambda^{(r)}\nabla g_1(\beta_i)$ and $\tilde{\mathbf{t}} = \tilde{\beta}_i - \lambda^{(r)}\nabla g_1(\tilde{\beta}_i)$
  Compute $\left(\hat{\beta}_{ij}^{(0)}, \hat{\tilde{\beta}}_{ij}^{(0)}\right)$, $j = 1,\ldots,p$, from (11)
  Compute $\hat{\beta}_{ij}$ and $\hat{\tilde{\beta}}_{ij}$, $j = 1,\ldots,p$, from (10)
  Update β_i and $\tilde{\beta}_i$: $\beta_{ij} = \hat{\beta}_{ij}$ and $\tilde{\beta}_{ij} = \hat{\tilde{\beta}}_{ij}$, $j = 1,\ldots,p$
until convergence
Return β_i and $\tilde{\beta}_i$.

Maximum values of λ_1 and λ_2

The ProGAdNet solver of (4) is outlined in Algorithm 1 with a specific pair of values of λ_1 and λ_2. However, we typically need to solve the optimization problem (4) over a set of values of λ_1 and λ_2, and then either use cross validation to determine the optimal values of λ_1 and λ_2, or use the stability selection technique to determine nonzero elements of β_i and $\tilde{\beta}_i$, as will be described later. Therefore, we also need to know the maximum values of λ_1 and λ_2. In the following, we will derive expressions for the maximum values of λ_1 and λ_2.

When we determine the maximum values of λ_1, $\lambda_{1\max}$, λ_2 can be omitted in our optimization problem, since when $\lambda_1 = \lambda_{1\max}$, we have $\beta_{ij} = 0$ and $\tilde{\beta}_{ij} = 0$, $\forall i$ and j. Thus, we can use the same method for determining the maximum value of λ in the Lasso problem [37] to find $\lambda_{1\max}$, which leads to

$$\lambda_{1\max} = \max\left\{\max_{j\neq i}|\mathbf{x}_j^T\mathbf{x}_i|, \max_{j\neq i}|\tilde{\mathbf{x}}_j^T\tilde{\mathbf{x}}_i|\right\}. \tag{13}$$

The maximum value of λ_2, $\lambda_{2\max}$ depends on λ_1. It is difficult to find $\lambda_{2\max}$ exactly. Instead, we will find an upper-bound for $\lambda_{2\max}$. Let us denote the objective function in (4) as $J(\beta_i, \tilde{\beta}_i)$, and let the jth column of $\mathbf{X}_{-i}$ ($\tilde{\mathbf{X}}_{-i}$) be $\mathbf{z}_i$ ($\tilde{\mathbf{z}}_i$). If the optimal solution of (4) is $\beta_i = \tilde{\beta}_i = \beta^*$, then the subgradient of $J(\beta_i, \tilde{\beta}_i)$ at the optimal solution should contain the zero vector, which yields

$$\begin{aligned} &2\mathbf{z}_j^T(\mathbf{x}_i - \mathbf{X}_{-i}\beta^*) + \lambda_1 s_{1j} + \lambda_2 s_{2j} = 0,\ j = 1,\ldots,p-1 \\ &2\tilde{\mathbf{z}}_j^T(\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\beta^*) + \lambda_1 \tilde{s}_{1j} + \lambda_2 \tilde{s}_{2j} = 0,\ j = 1,\ldots,p-1, \end{aligned} \tag{14}$$

where $s_{1j} = 1$ if $\beta_{ij} > 0$, $= -1$ if $\beta_{ij} < 0$, or $\in [-1,1]$ if $\beta_{ij} = 0$, and $s_{2j} \in [-1,1]$, and similarly, $\tilde{s}_{1j} = 1$ if $\tilde{\beta}_{ij} > 0$, $= -1$ if $\tilde{\beta}_{ij} < 0$, or $\in [-1,1]$ if $\tilde{\beta}_{ij} = 0$, and $\tilde{s}_{2j} \in [-1,1]$. Therefore, we should have $\lambda_2 > |2\mathbf{z}_j^T(\mathbf{x}_i - \mathbf{X}_{-i}\beta^*) + \lambda_1 s_{1j}|$ and $\lambda_2 > |2\tilde{\mathbf{z}}_j^T(\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\beta^*) + \lambda_1 \tilde{s}_{1j}|$, which can be satisfied if we choose $\lambda_2 = \max_j \max\{\lambda_1 + |2\mathbf{z}_j^T(\mathbf{x}_i - \mathbf{X}_{-i}\beta^*)|, \lambda_1 + |2\tilde{\mathbf{z}}_j^T(\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\beta^*)|\}$. Therefore, the maximum value of λ_2 can be written as

$$\lambda_{2\max} = \max_{j\neq i} \max\{\lambda_1 + |2\mathbf{x}_j^T(\mathbf{x}_i - \mathbf{X}_{-i}\beta^*)|, \lambda_1 + |2\tilde{\mathbf{x}}_j^T(\tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\beta^*)|\}. \tag{15}$$

To find $\lambda_{2\max}$ from (15), we need to know β^*. This can be done by solving the Lasso problem that minimizes $J(\beta) = \| \mathbf{x}_i - \mathbf{X}_{-i}\beta \|^2 + \| \tilde{\mathbf{x}}_i - \tilde{\mathbf{X}}_{-i}\beta \|^2 + 2\lambda_1 \| \beta \|_1$ using an efficient algorithm such as glmnet [38].

Stability Selection

As mentioned earlier, parameter λ_1 encourages sparsity in the inferred gene network, while λ_2 induces sparsity in the changes of the network under two conditions. Generally, larger values of λ_1 and λ_2 induce a higher level of sparsity. Therefore, appropriate values of λ_1 and λ_2 need to be determined, which can be done with cross validation [38]. However, the nonzero entries of matrices $\mathbf{B}$ and $\tilde{\mathbf{B}}$, estimated with a specific pair of values of λ_1 and λ_2 determined by cross validation, may not be stable in the sense that small perturbation in the data may result in considerably different $\mathbf{B}$ and $\tilde{\mathbf{B}}$. We can employ an alternative technique, named stability selection [39], to select stable variables, as described in the following.

We first determine the maximum value of λ_1, namely $\lambda_{1\max}$, using the method described earlier, then choose a set of k_1 values for λ_1, denoted as $\mathscr{S}_1 = \{\lambda_{1\max}, \alpha_1\lambda_{1\max}, \alpha_1^2\lambda_{1\max}, \ldots, \alpha_1^{k_1-1}\lambda_{1\max}\}$, where $0 < \alpha_1 < 1$. For each value $\lambda_1 \in \mathscr{S}_1$, we find the maximum value of λ_2, namely $\lambda_{2\max}(\lambda_1)$, and then choose a set of k_2 values for λ_2, denoted $\mathscr{S}_2(\lambda_1) = \{\lambda_{2\max}(\lambda_1), \alpha_2\lambda_{2\max}(\lambda_1), \ldots, \alpha_2^{k_2-1}\lambda_{2\max}(\lambda_1)\}$, where $0 < \alpha_2 < 1$. This gives a set of $K = k_1k_2$ pairs of (λ_1, λ_2). After we create the parameter space, for each (λ_1, λ_2) pair in the space, we randomly divide the data $(\mathbf{X}, \tilde{\mathbf{X}})$ into two subsets of equal size, and infer the network with our proximal gradient algorithm using each subset of the data. We repeat this process for N times, which yields $2N$ estimated network matrices, $\hat{\mathbf{B}}$ and $\hat{\tilde{\mathbf{B}}}$. Typically, $N = 50$ is chosen.

Let $m_{ij}^{(k)}$, $\tilde{m}_{ij}^{(k)}$, and $\Delta m_{ij}^{(k)}$ be the number of nonzero $\hat{b}_{ij}$'s and $\hat{\tilde{b}}_{ij}$'s, and $(\hat{b}_{ij} - \hat{\tilde{b}}_{ij})$'s, respectively, obtained with the kth pair of (λ_1, λ_2). Then, $r_{ij} = \sum_{k=1}^K m_{ij}^{(k)}/(NK)$, $\tilde{r}_{ij} = \sum_{k=1}^K \tilde{m}_{ij}^{(k)}/(NK)$, and $\Delta r_{ij} = \sum_{k=1}^K \Delta m_{ij}^{(k)}/(NK)$ give the frequency of an edge from gene j to gene i detected under two conditions, and the frequency of the changes for an edge from gene j to gene i, respectively. A larger r_{ij}, $\tilde{r}_{ij}$, or Δr_{ij} indicates a higher likelihood that an edge from gene j to gene i exists, or the edge from gene j to gene i has changed. Therefore, we will use r_{ij}, $\tilde{r}_{ij}$ and Δr_{ij} to rank the reliability of the detected edges and the changes of edges, respectively. Alternatively, we can declare an edge from gene j to gene i exists if $r_{ij} \geq c$ or $\tilde{r}_{ij} \geq c$; and similarly the edge between gene j to gene i has changed if $\Delta r_{ij} \geq c$, where c is constant and can be any value in $[0.6, 0.9]$ [39].

Software glmnet and lqa

Two software packages, glmnet and lqa, were used in computer simulations. The software glmnet [38] for solving the Lasso problem is available at https://cran.r-project.org/web/packages/glmnet. The software lqa [28] used in [27] for inferring multiple gene networks is available at https://cran.r-project.org/web/packages/lqa/.

Results and Discussion

Computer Simulation with Linear Regression Model

We generated data from one of p pairs of linear regression models in (3) instead of all p pairs of simultaneous equations in (2), or equivalently (3), as follows. Without loss of generality, let us consider the first equation in (3). The goal was to estimate β_1 and $\tilde{\beta}_1$, and then identify pairs $(\beta_{i1}, \tilde{\beta}_{i1})$, where $\beta_{i1} \neq \tilde{\beta}_{i1}$. Entries of $n \times (p-1)$ matrices $\mathbf{X}_{-1}$ and $\tilde{\mathbf{X}}_{-1}$ were generated independently from the standardized Gaussian distribution. In the first simulation setup, we chose $n = 100$ and $p - 1 = 200$. Twenty randomly selected entries of β_1 were generated from a random variable uniformly distributed over the intervals $[0.5, 1.5]$ and $[-1.5, -0.5]$, and remaining entries were set to zero; $\tilde{\beta}_1$ was generated by randomly changing 10 entries of β_1 as follows: 4 randomly selected nonzero entries were set to zero, and 6 randomly selected zero entries were changed to a value uniformly distributed over the intervals $[0.5, 1.5]$ and $[-1.5, -0.5]$. The noise vectors $\mathbf{e}_1$ and $\tilde{\mathbf{e}}_1$ were generated from a Gaussian distribution with mean zero and variance σ^2 varying from 0.01 to 0.05, 0.1, and 0.5, and then $\mathbf{x}_1$ and $\tilde{\mathbf{x}}_1$ were calculated from (3).

Simulated data $\mathbf{x}_1$, $\tilde{\mathbf{x}}_1$, $\mathbf{X}_{-1}$ and $\tilde{\mathbf{X}}_{-1}$ were analyzed with our ProGAdNet, lqa [28] and glmnet [38]. Since lqa was employed by [27], the results of lqa represent the performance of the network inference approach in [27]. The glmnet algorithm implements the Lasso approach in [40]. Both ProGAdNet and lqa estimate β_1 and $\tilde{\beta}_1$ jointly by solving the optimization problem (4), but glmnet estimates β_1 and $\tilde{\beta}_1$ separately, by solving the following two problems separately, $\hat{\beta}_1 = \arg\min_{\beta_1} \{\| \mathbf{x}_1 - \mathbf{X}_{-1}\beta_1 \|^2 + \lambda_1 \| \beta_1 \|_1$, and $\hat{\tilde{\beta}}_1 = \arg\min_{\tilde{\beta}_1} \| \tilde{\mathbf{x}}_1 - \tilde{\mathbf{X}}_{-1}\tilde{\beta}_1 \|^2 + \lambda_2 \| \tilde{\beta}_1 \|_1$. The lqa algorithm uses a local quadratic approximation of the nonsmooth penalty term [41] in the objective function, and therefore, it cannot shrink variables to zero exactly. To alleviate this problem, we set $\hat{\beta}_{i1} = 0$ if $|\hat{\beta}_{i1}| < 10^{-4}$, and similarly $\hat{\tilde{\beta}}_{i1} = 0$ if $|\hat{\tilde{\beta}}_{i1}| < 10^{-4}$, where $\hat{\beta}_{i1}$ and $\hat{\tilde{\beta}}_{i1}$ represent the estimates of β_{i1} and $\tilde{\beta}_{i1}$, respectively. Five fold cross validation was used to determine the optimal values of parameters λ_1 and λ_2 in the optimization problem. Specifically, for ProGAdNet and lqa, the prediction error (PE) was estimated at each pair of values of λ_1 and λ_2, and the smallest PE along with the corresponding values of λ_1 and λ_2, $\lambda_{1\,\min}$ and $\lambda_{2\,\min}$, were determined. Then, the optimal values of λ_1 and λ_2 were the values corresponding to the PE that was two standard error (SE) greater than the minimum PE, and were greater than $\lambda_{1\,\min}$ and $\lambda_{2\,\min}$, respectively. For glmnet, the optimal values of λ_1 and λ_2 were determined separately also with the two-SE rule.

The inference process was repeated for 50 replicates of the data, and the detection power and the false discovery rate (FDR) for $(\beta_1, \tilde{\beta}_1)$ and $\Delta\beta = \beta_1 - \tilde{\beta}_1$ calculated from the results of the 50 replicates in the first simulation setup are plotted in Figure 1. It is seen that all three algorithms offer almost identical power equal or close to 1, but exhibit different FDRs. The FDR of lqa is the highest, whereas the FDR of ProGAdNet is almost the same as that of glmnet for β_1 and $\tilde{\beta}_1$, and the lowest for $\Delta\beta_1$.

In the second simulation setup, we let sample size $n = 150$, noise variance $\sigma^2 = 0.1$, and the number of variables $p-1$ be 500, 800, and 1,000. Detection power and FDR are depicted in Figure 2. Again, the three algorithms have almost identical power, and ProGAdNet offers an FDR similar to that of glmnet, but lower than that of lqa for β_1 and $\tilde{\beta}_1$, and the lowest FDR for $\Delta\beta_1$. Simulation results in Figures 1 and 2 demonstrate that our ProGAdNet offers the best performance when compared with glmnet and lqa. The CPU times of one run of ProGAdNet, lqa, and glmnet for inferring a linear model with $n = 150$, $p-1 = 1,000$, and $\sigma^2 = 0.1$ at the optimal values of λ_1 and λ_2 were 5.82, 145.15, and 0.0037 seconds, respectively.

Note that although ProGAdNet and lqa solve the same optimization problem, ProGAdNet significantly outperforms lqa. The performance gap is due to the fact that the lqa algorithm uses an approximation of the objective function, whereas our algorithm solves optimization problem (4) exactly. In other words, our ProGAdNet algorithm can always find the optimal solution to the optimization problem, since the objective function is convex, but the lqa algorithm generally cannot find the optimal solution. Moreover, our computer simulations show that our ProGAdNet algorithm is much faster than the lqa algorithm.

Computer Simulation with Gene Networks

The GeneNetWeaver software [42] was used to generate gene networks whose structures are similar to those of real gene networks. Note that GeneNetWeaver was also employed by the DREAM5 challenge for gene network inference to simulate golden standard networks [12]. GeneNetWeaver outputs an adjacency matrix to characterize a specific network structure. We chose the number of genes in the network to be $p = 50$, and obtained a $p \times p$ adjacency matrix $\mathbf{A}$ through GeneNetWeaver. The number of nonzero entries of $\mathbf{A}$, which determined the edges of the network, was 62. Hence the network is sparse, as the total number of possible edges is $p(p-1) = 2,450$. We randomly changed 6 entries of $\mathbf{A}$ to yield another matrix $\tilde{\mathbf{A}}$ as the adjacency matrix of the gene network under another condition. Note that the number of changed edges is small relative to the number of existing edges.

After the two network topologies were generated, the next step was to generate gene expression data. Letting a_{ij} be the entry of $\mathbf{A}$ on the ith row and the jth column, we generated a $p \times p$ matrix $\mathbf{B}$ such that $b_{ij} = 0$ if $a_{ij} = 0$, and b_{ij} was randomly sampled from a uniform random variable on the intervals $[-1, 0)$ and

$(0,1]$ if $a_{ij} \neq 0$. Another $p \times p$ matrix $\tilde{\mathbf{B}}$ was generated such that $\tilde{b}_{ij} = b_{ij}$ if $\tilde{a}_{ij} = a_{ij}$, or $\tilde{b}_{ij}$ was randomly generated from a uniform random variable on the intervals $[-1,0)$ and $(0,1]$ if $\tilde{a}_{ij} \neq a_{ij}$. Note that (2) gives $\mathbf{X} = (\mathbf{I}-\mathbf{B})^{-1}\mathbf{E}$ and $\tilde{\mathbf{X}} = (\mathbf{I}-\tilde{\mathbf{B}})^{-1}\tilde{\mathbf{E}}$. These relationships suggest generating first entries of $\mathbf{E}$ and $\tilde{\mathbf{E}}$ independently from a Gaussian distribution with zero mean and unit variance, and then finding matrices $\mathbf{X}$ and $\tilde{\mathbf{X}}$ from these two equations, respectively. With real data, gene expression levels $\mathbf{X}$ and $\tilde{\mathbf{X}}$ are measured with techniques such as microarray or RNA-seq, and there are always measurement errors. Therefore, we simulated measured gene expression data as $\mathbf{Y} = \mathbf{X} + \mathbf{V}$ and $\tilde{\mathbf{Y}} = \tilde{\mathbf{X}} + \tilde{\mathbf{V}}$, where $\mathbf{V}$ and $\tilde{\mathbf{V}}$ model measurement errors that were independently generated from a Gaussian distribution with zero mean and variance σ^2 that will be specified later. Fifty pairs of network replicates and their gene expression data were generated independently.

Finally, gene networks were inferred with our ProGAdNet algorithm by solving the optimization problem (4), where $\mathbf{x}_i$, $\mathbf{X}_{-i}$, $\tilde{\mathbf{x}}_i$, and $\tilde{\mathbf{X}}_{-i}$ were replaced with the measured gene expression data $\mathbf{y}_i$, $\mathbf{Y}_{-i}$, $\tilde{\mathbf{y}}_i$, and $\tilde{\mathbf{Y}}_{-i}$. Stability selection was employed to rank the edges that were changed under two conditions. As comparison, we also used Lasso to infer the network topology under each condition by solving the following optimization problems

$$\begin{aligned} \hat{\mathbf{B}} =& \arg\min_{\mathbf{B}} \| \mathbf{Y} - \mathbf{YB} \|^2 + \lambda_1 \| \mathbf{B} \|_1 \\ & \text{subject to } b_{ii} = 0, i = 1, \ldots, p, \\ \hat{\tilde{\mathbf{B}}} =& \arg\min_{\tilde{\mathbf{B}}} \| \tilde{\mathbf{Y}} - \tilde{\mathbf{Y}}\tilde{\mathbf{B}} \|^2 + \lambda_1 \| \tilde{\mathbf{B}} \|_1 \\ & \text{subject to } \tilde{b}_{ii} = 0, i = 1, \ldots, p. \end{aligned} \tag{16}$$

Note that each optimization problem can be decomposed into p separate problems that can be solved with Lasso. The glmnet algorithm [38] was again used to implement Lasso. The stability selection technique was employed again to rank the differential edges detected by Lasso. The lqa algorithm was not considered to infer simulated gene networks, because it is very slow and its performance is worse than ProGAdNet and Lasso as shown in the previous section. We also employed the GENIE3 algorithm in [43] to infer $\mathbf{B}$ and $\tilde{\mathbf{B}}$ separately, because GENIE3 gave the best overall performance in the DREAM5 challenge [12]. Finally, following the performance assessment procedure in [12], we used the precision-recall (PR) curve and the area under the PR curve (AUPR) to compare the performance of ProGAdNet with that of Lasso and GENIE3. For ProGAdNet and Lasso, the estimate of $\Delta\mathbf{B} = \mathbf{B} - \tilde{\mathbf{B}}$ was obtained, and the nonzero entries of $\Delta\mathbf{B}$ were ranked based on their frequencies obtained in stability selection. Then, the PR curve for changed edges was obtained from the ranked entries of $\Delta\mathbf{B}$ from pooled results for the 50 network replicates. Two lists of ranked network edges were obtained from GENIE3: one for $\mathbf{B}$ and the other for $\tilde{\mathbf{B}}$. For each cutoff value of the rank, we obtain an adjacency matrix $\mathbf{A}$ from $\mathbf{B}$ as follows: the (i,j)th entry of $\mathbf{A}$ $a_{ij} = 1$ if b_{ij} is above the cutoff value, and otherwise $a_{ij} = 0$. Similarly, another adjacency matrix $\tilde{\mathbf{A}}$ was obtained from $\tilde{\mathbf{B}}$. Then, the PR curve for changed edges detected by GENIE3 was obtained from $\mathbf{A} - \tilde{\mathbf{A}}$, again from pooled results for the 50 network replicates.

Figures 3 and 4 depict the PR curves of ProGAdNet, Lasso, and GENIE3 for measurement noise variance $\sigma^2 = 0.05$ and 0.5, respectively. The number of samples varies from 50, 100, 200 to 300. It is seen from Figure 3 that our ProGAdNet offers much better performance than Lasso and GENIE3. When the noise variance increases from 0.05 to 0.5, the performance of all three algorithms degrades, but our ProGAdNet still outperforms Lasso and GENIE3 considerably, as shown in Figure 4. Table 1 lists AUPRs of ProGAdNet, Lasso and GENIE3, which again shows that our ProGAdNet outperforms Lasso and GENIE3 consistently at all sample sizes.

Real Data Analysis

We next use the ProGAdNeT algorithm to analyze RNA-seq data of breast tumors and normal tissues. In The Cancer Genome Atlas (TCGA) database, there are RNA-seq data for 1,101 breast invasive carcinoma (BRCA)

samples and 113 normal tissues. Although the TCGA database contains RNA-seq data of a number of other cancers, only the BRCA dataset has more than 100 normal tissue samples, and all other datasets contain less than 60 normal tissue samples. Since a small number of samples may compromise the accuracy of network inference, we only analyzed the BRCA dataset. The RNA-seq level 3 data for 113 normal tissues and their matched BRCA tumors were downloaded. The TCGA IDs of these 226 samples are given in Additional file 1. The scaled estimates of gene expression levels in the dataset were extracted, and they were multiplied by 10^6, which yielded transcripts per million (TPM) value of each gene. The batch effect was corrected with the removeBatchEffect function in the Limma package [44] based on the batch information in the TCGA barcode of each sample (the "plate" field in the barcode). The RNA-seq data include expression levels of 20,531 genes. Two filters were used to obtain informative genes for further network analysis. First, genes with their expression levels in the lower 30 percentile were removed. Second, the coefficient of variation (CoV) was calculated for each of the remaining genes, and then genes with their CoVs in the lower 70 percentile were discarded. This resulted in 4,310 genes, and their expression levels in 113 normal tissues and 113 matched tumor tissues were used by the ProGAdNet algorithm to jointly infer the gene networks in normal tissues and tumors, and then to identify the difference in the two gene networks."

Since small changes in b_{ji} in the network model (1) may not have much biological effect, we regarded the regulatory effect from gene j to gene i to be changed using the following two criteria rather than the simple criterion $\tilde{b}_{ji} \neq b_{ji}$. The first criterion is $|\tilde{b}_{ji} - b_{ji}| \geq \min\{|\tilde{b}_{ji}|, |b_{ji}|\}$, which ensures that there is at least one-fold change relative to $\min\{|\tilde{b}_{ji}|, |b_{ji}|\}$. However, when one of $\tilde{b}_{ji}$ and b_{ji} is zero or near zero, this criterion does not filter out very small $|\tilde{b}_{ji} - b_{ji}|$. To avoid this problem, we further considered the second criterion. Specifically, nonzero $\tilde{b}_{ji}$ and b_{ji} for all j and i were obtained, and the 20 percentile value of all $|\tilde{b}_{ji}|$ and $|b_{ji}|$, T, was found. Then, the second criterion is $\max\{|\tilde{b}_{ji}|, |b_{ji}|\} \geq T$. As in computer simulations, the stability selection was employed to identify network changes reliably. As the number of genes, 4,310, is quite big, it is time consuming to repeat 100 runs per λ_1 and λ_2 pair. To reduce the computational burden, we used five-fold cross validation to choose the optimal values of λ_1 and λ_2 based on the two-SE rule used in computer simulation, and then performed stability selection with 100 runs for the pair of optimal values. Note that stability selection at an appropriate point of hyperparameters is equally valid compared with that done along a path of hyperparameters [39]. The threshold for Δr_{ij} for determining network changes as described in the Method section was chosen to be $c = 0.9$.

Our network analysis with ProGAdNeT identified 268 genes that are involved in at least one changed edge. Names of these genes are listed in supplementary file 2. We named the set of these 268 genes as the dNet set. To assess whether the dNet genes relate to the disease status, we performed gene set enrichment analysis (GSEA) with the C2 gene sets in the molecular signatures database (MSigDB) [45, 46]. C2 gene sets consist of 4,738 gene sets that include pathways in major pathway dabases such as KEGG [47], REACTOME [48], and BIOCARTA [49]. After excluding gene sets with more than 268 genes or less than 15 genes, 2,469 gene sets remained. Searching over the names of these 2,469 gene sets with key words "breast cancer", "breast tumor", "breast carcinoma" and "BRCA" identified 139 gene sets that are related to breast cancer. Using Fisher's exact test, we found that 78 of 2,469 C2 gene sets were enriched in the dNet gene set at a FDR of $< 10^{-4}$. The list of the 78 gene sets is in Additional file 3. Of these 78 gene sets, 24 are among the 139 breast cancer gene sets, which is highly significant (at Fisher's exact test p-value 2.3×10^{-9}). The top 20 enriched gene sets are listed in Table 2. As seen from names of these gene sets, 11 of the 20 gene sets are breast cancer gene sets, and 7 sets are related to other types of cancer. These GSEA results clearly show that the dNet gene set that our ProGAdNet algorithm identified is very relevant to the cancer disease status.

Conclusion

In this paper, we developed a very efficient algorithm, named ProGAdNet, for inference of two gene networks based on gene expression data under two different conditions, which were further used to identify differential

changes in the network. Computer simulations showed that our ProGAdNet offered much better inference accuracy than existing algorithms. Analysis of a set of RNA-seq data of breast tumors and normal tissues with ProGAdNet identified a set of genes involved in differential changes of the gene network. A number of gene sets of breast cancer or other types of cancer are significantly enriched in the identified gene set, which shows that the identified gene set is very informative about the disease status of the tissues. As gene network rewiring occurs frequently under different molecular context, our ProGAdNet algorithm provides a valuable tool for identifying changed gene-gene interactions.

Funding

This work was supported by the National Science Foundation (CC-1319981 to XC) and National Institute of General Medical Sciences (5R01GM104975 to XC).

Availability of data and materials

The software package implementing the ProGAdNet algorithm and computer simulations is available upon request. The RNA-Seq data of 113 breast tumors and 113 normal tissues used in differential gene network analysis can be downloaded from the website of Gnomic Data Commons at https://Gd.cancer.gov/ using the TCGA sample IDs in Additional file 1.

Author's contributions

XC conceived the idea and supervised the project. XC, CW, and FG designed the algorithm. CW performed computer simulation with linear regression models and analysis of real data. FG performed computer simulation with gene networks. GBG contributed to the development of the algorithm and computer simulations. GD contributed to real data analysis. XC, CW, and FG wrote the manuscript. All authors read and approved the final manuscript.

References

1. Ideker, T., Krogan, N.J.: Differential network biology. Mol. Syst. Biol. **8**, 565 (2012)

2. Califano, A.: Rewiring makes the difference. Mol. Syst. Biol. **7**, 463 (2011)

3. Harbison, C.T., Gordon, D.B., Lee, T.I., Rinaldi, N.J., Macisaac, K.D., Danford, T.W., Hannett, N.M., Tagne, J.B., Reynolds, D.B., Yoo, J., Jennings, E.G., Zeitlinger, J., Pokholok, D.K., Kellis, M., Rolfe, P.A., Takusagawa, K.T., Lander, E.S., Gifford, D.K., Fraenkel, E., Young, R.A.: Transcriptional regulatory code of a eukaryotic genome. Nature **431**(7004), 99–104 (2004)

4. Workman, C.T., Mak, H.C., McCuine, S., Tagne, J.B., Agarwal, M., Ozier, O., Begley, T.J., Samson, L.D., Ideker, T.: A systems approach to mapping DNA damage response pathways. Science **312**(5776), 1054–9 (2006)

5. Bandyopadhyay, S., Mehta, M., Kuo, D., Sung, M.K., Chuang, R., Jaehnig, E.J., Bodenmiller, B., Licon, K., Copeland, W., Shales, M., Fiedler, D., Dutkowski, J., Guenole, A., van Attikum, H., Shokat, K.M., Kolodner, R.D., Huh, W.K., Aebersold, R., Keogh, M.C., Krogan, N.J., Ideker, T.: Rewiring of genetic networks in response to DNA damage. Science **330**(6009), 1385–9 (2010)

6. Luscombe, N.M., Babu, M.M., Yu, H., Snyder, M., Teichmann, S.A., Gerstein, M.: Genomic analysis of regulatory network dynamics reveals large topological changes. Nature **431**(7006), 308–12 (2004)

7. Mani, K.M., Lefebvre, C., Wang, K., Lim, W.K., Basso, K., Dalla-Favera, R., Califano, A.: A systems biology approach to prediction of oncogenes and molecular perturbation targets in B-cell lymphomas. Mol. Syst. Biol. **4** (2008). 10.1038/msb.2008.2

8. Wang, K., Saito, M., Bisikirska, B.C., Alvarez, M.J., Lim, W.K., Rajbhandari, P., Shen, Q., Nemenman, I., Basso, K., Margolin, A.A., Klein, U., Dalla-Favera, R., Califano, A.: Genome-wide identification of post-translational modulators of transcription factor activity in human B cells. Nat. Biotechnol. **27**(9), 829–39 (2009)

9. Bansal, M., Belcastro, V., Ambesi-Impiombato, A., di Bernardo, D.: How to infer gene networks from expression profiles. Mol. Syst. Biol. **3**, 78 (2007)

10. Markowetz, F., Spang, R.: Inferring cellular networks - a review. BMC Bioinformatics **8**(Suppl 6), 5 (2007)

11. Sun, N., Zhao, H.: Reconstructing transcriptional regulatory networks through genomics data. Stat. Methods Med. Res. **18**(6), 595–617 (2009)

12. Marbach, D., Costello, J.C., Kuffner, R., Vega, N.M., Prill, R.J., Camacho, D.M., Allison, K.R., Kellis, M., Collins, J.J., Stolovitzky, G.: Wisdom of crowds for robust gene network inference. Nat Meth **9**(8), 796–804 (2012)

13. Butte, A.J., Tamayo, P., Slonim, D., Golub, T.R., Kohane, I.S.: Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. Proc. Natl. Acad. Sci. USA **97**(22), 12182–6 (2000)

14. Basso, K., Margolin, A.A., Stolovitzky, G., Klein, U., Dalla-Favera, R., Califano, A.: Reverse engineering of regulatory networks in human B cells. Nat. Genet. **37**(4), 382–90 (2005)

15. Dobra, A., Hans, C., Jones, B., Nevins, J.R., Yao, G., West, M.: Sparse graphical models for exploring gene expression data. J. Multivar. Anal. **90**, 196–212 (2004)

16. Schäfer, J., Strimmer, K.: An empirical Bayes approach to inferring large-scale gene association networks. Bioinform. **21**(6), 754–764 (2005)

17. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian network to analyze expression data. J. Comput. Biol. **7**, 601–620 (2000)

18. Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., Fridman, N.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat. Genet. **34**, 166–178 (2003)

19. Gardner, T.S., di Bernardo, D., Lorenz, D., Collins, J.J.: Inferring genetic networks and identifying compound mode of action via expression profiling. Science **301**, 102–105 (2003)

20. di Bernardo, D., Thompson, M.J., Gardner, T.S., Chobot, S.E., Eastwood, E.L., Wojtovich, A.P., Elliott, S.J., Schaus, S.E., Collins, J.J.: Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. Nature biotechnology **23**(3), 377–383 (2005)

21. Schäfer, J., Strimmer, K.: A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. Stat. Appl. Genet. Mol. Biol. **4**, 32 (2005)

22. Bonneau, R., Reiss, D., Shannon, P., Facciotti, M., Hood, L., Baliga, N., Thorsson, V.: The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*. Genome Biol. **7**(5), 36 (2006)

23. de la Fuente, A.: From 'differential expression' to 'differential networking' – identification of dysfunctional regulatory networks in diseases. Trends Genet. **26**(7), 326–333 (2010)

24. Tegner, J., Yeung, M.K., Hasty, J., Collins, J.J.: Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proc. Natl. Acad. Sci. USA **100**(10), 5944–9 (2003)

25. Jeong, H., Mason, S.P., Barabássi, A.L., Oltvai, Z.N.: Lethality and centrality in protein networks. Nature **411**(6833), 41–42 (2001)

26. Thieffry, D., Huerta, A.M., Pérez-Rueda, E., Collado-Vides, J.: From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in *Escherichia coli*. Bioessays **20**(5), 433–440 (1998)

27. Omranian, N., Eloundou-Mbebi, J.M., Mueller-Roeber, B., Nikoloski, Z.: Gene regulatory network inference using fused lasso on multiple data sets. Scientific reports **6**, 20533 (2016)

28. Ulbricht, J.: lqa: penalized likelihood inference for glms. URL http://CRAN. R-project. org/package= lqa (2012)

29. Lichtblau, Y., Zimmermann, K., Haldemann, B., Lenze, D., Hummel, M., Leser, U.: Comparative assessment of differential network analysis methods. Briefings in bioinformatics, 061 (2016)

30. Thieffry, D., Huerta, A.M., Pérez-Rueda, E., Collado-Vides, J.: From specific gene regulation to genomic networks: a global analysis of transcriptional regulation in escherichia coli. Bioessays **20**(5), 433–440 (1998)

31. Gardner, T.S., di Bernardo, D., Lorenz, D., Collins, J.J.: Inferring genetic networks and identifying compound mode of action via expression profiling. Science **301**(5629), 102–5 (2003)

32. Tegner, J., Yeung, M.S., Hasty, J., Collins, J.J.: Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. Proceedings of the National Academy of Sciences **100**(10), 5944–5949 (2003)

33. Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., Knight, K.: Sparsity and smoothness via the fused lasso. J. R. Stat. Soc. Series B Stat. Methodol. **67**, 91–108 (2005)

34. Parikh, N., Boyd, S.: Proximal algorithms. Foundations and Trends in Optimization, 1–96 (2013)

35. Friedman, J., Hastie, T., Hofling, H., Tibshirani, R.: Pathwise coordinate optimization. Ann. Appl. Stat. **1**(2), 302–332 (2007)

36. Hoefling, H.: A path algorithm for the fused lasso signal approximator. J. Comput. Graph. Stat. **19**(4), 984–1006 (2010)

37. Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., Tibshirani, R.J.: Strong rules for discarding predictors in lasso-type problems. J. R. Statist. Soc. B **74**, 245–266 (2012)

38. Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. J. Stat. Softw. **33**(1), 1–1722 (2010)

39. Meinshausen, N., Bühlmann, P.: Stability selection. J. R. Statist. Soc. B **72**(4), 417–473 (2010)

40. Tibshirani, R.: Regression shrinkage and selection via the Lasso. J. R. Statistical Soc. Ser. B **58**, 267–288 (1996)

41. Fan, J., Li, R.: Variable selection via nonconcave penalized likelihood and its oracle properties. J. Amer. Stat. Assoc. **96**, 1348–1360 (2001)

42. Schaffter, T., Marbach, D., Floreano, D.: GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics **27**(16), 2263–2270 (2011)

43. Irrthum, A., Wehenkel, L., Geurts, P., *et al.*: Inferring regulatory networks from expression data using tree-based methods. PloS one **5**(9), 12776 (2010)

44. Smyth, G.K.: Limma: linear models for microarray data. In: Bioinformatics and Computational Biology Solutions Using R and Bioconductor, pp. 397–420. Springer, ??? (2005)

45. Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., *et al.*: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proceedings of the National Academy of Sciences **102**(43), 15545–15550 (2005)

46. Liberzon, A., Birger, C., Thorvaldsdóttir, H., Ghandi, M., Mesirov, J.P., Tamayo, P.: The molecular signatures database hallmark gene set collection. Cell systems **1**(6), 417–425 (2015)

47. Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., Tanabe, M.: Kegg for integration and interpretation of large-scale molecular data sets. Nucleic acids research **40**(D1), 109–114 (2011)

48. Fabregat, A., Sidiropoulos, K., Garapati, P., Gillespie, M., Hausmann, K., Haw, R., Jassal, B., Jupe, S., Korninger, F., McKay, S., *et al.*: The reactome pathway knowledgebase. Nucleic acids research **44**(D1), 481–487 (2015)

49. Nishimura, D.: Biocarta. Biotech Software & Internet Report: The Computer Software Journal for Scient **2**(3), 117–120 (2001)

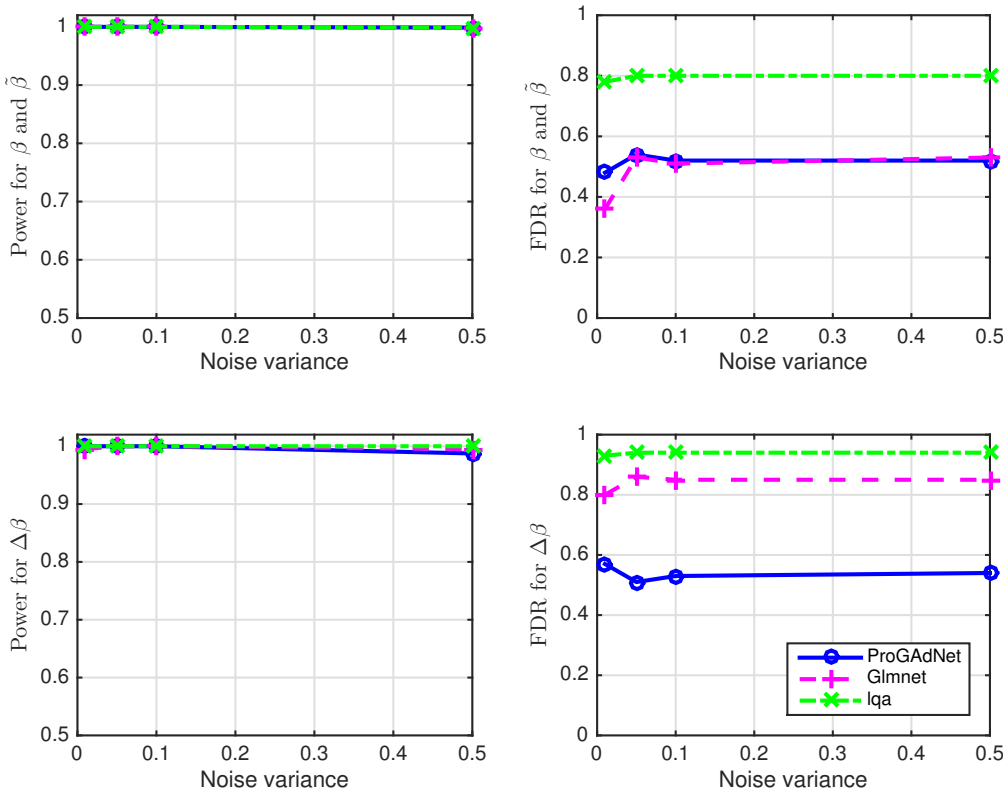

Figure 1. Performance of ProGAdNet, lqa, and Lasso in the inference of linear regression models. Number of samples $n = 100$, and number of variables $p - 1 = 200$.

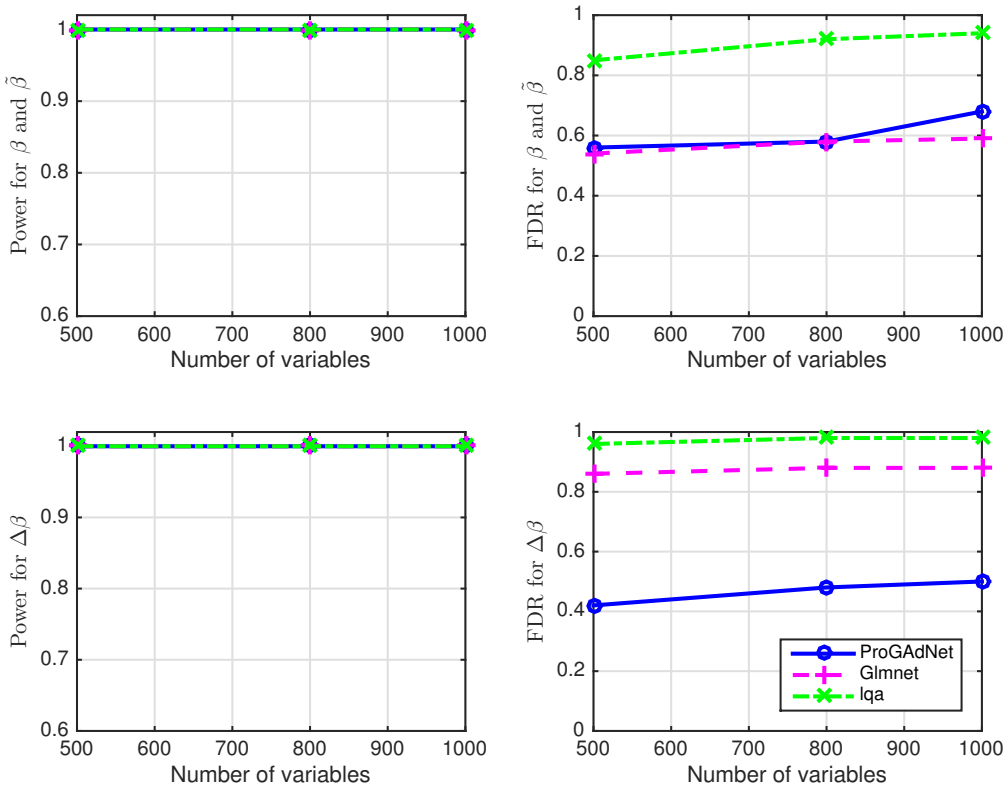

Figure 2. Performance of ProGAdNet, lqa, and Lasso in the inference of linear regression models. Number of samples $n = 150$ and noise variance $\sigma^2 = 0.1$.

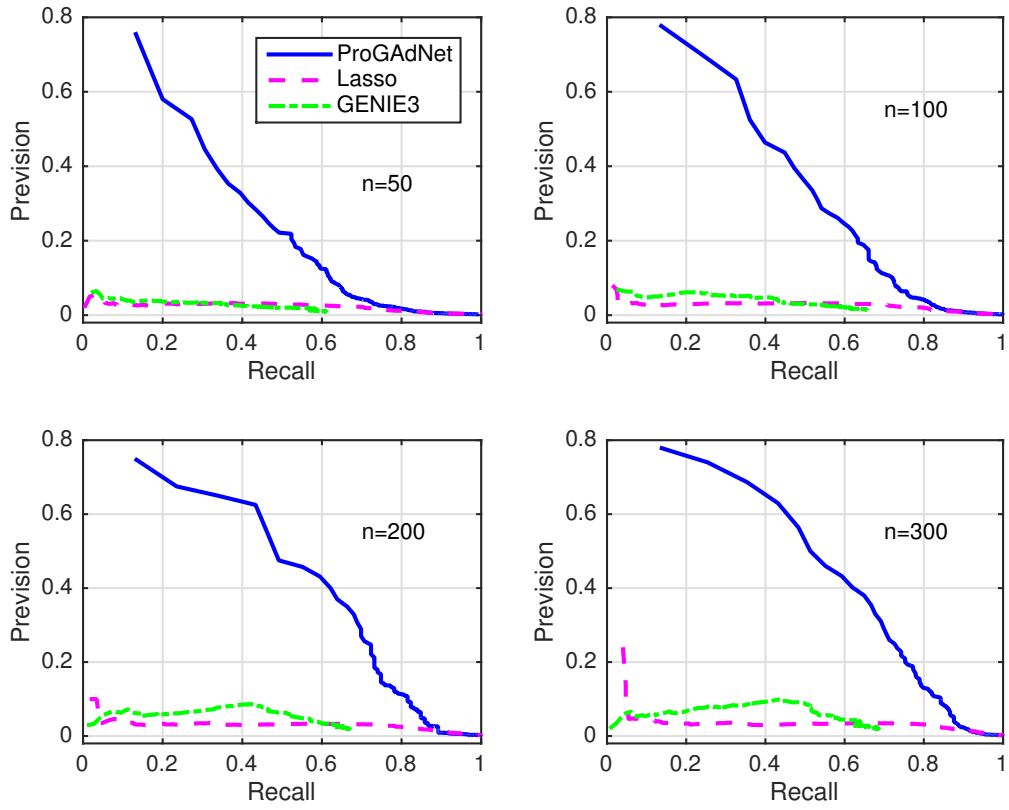

Figure 3. Precision-recall curves for ProGAdNet, Lasso, and GENIE3 in detecting changed edges of simulated gene networks. Variance of the measurement noise is $\sigma^2 = 0.05$, and sample size n=50, 100, 200, and 300.

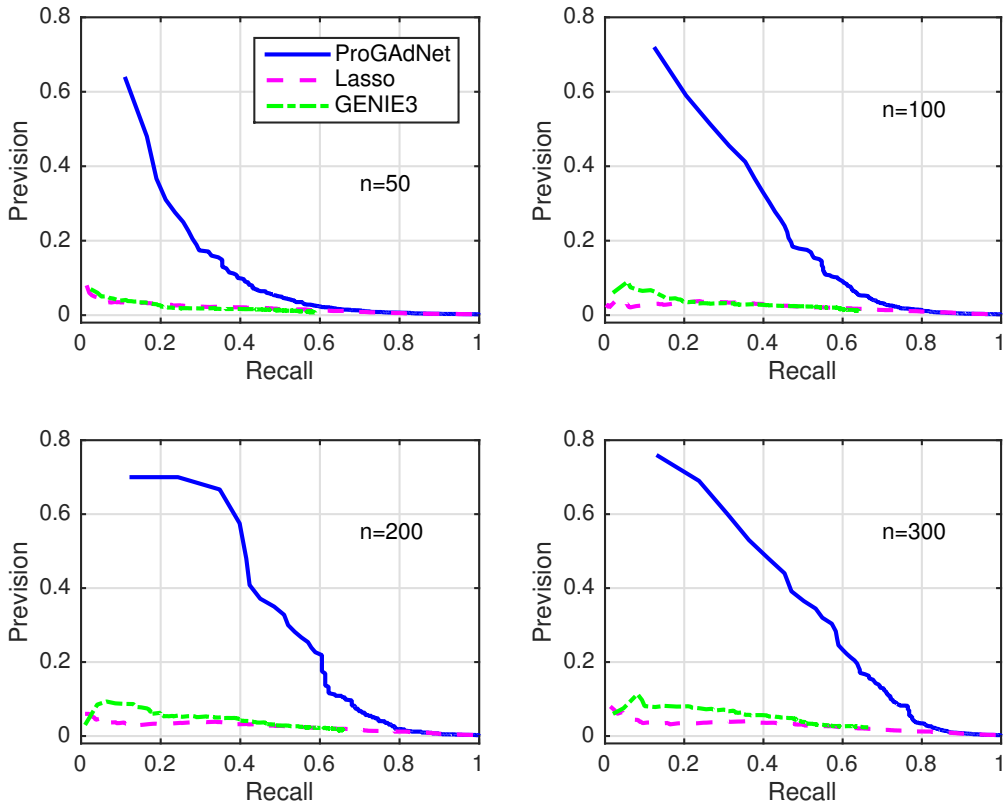

Figure 4. Precision-recall curves for ProGAdNet, Lasso, and GENIE3 in detecting changed edges of simulated gene networks. Variance of the measurement noise is $\sigma^2 = 0.5$, and sample size n=50, 100, 200, and 300.

Table 1. AUPRs of ProGAdNet, Lasso, and GENIE3 for detecting the changed edges of simulated gene networks

	$\sigma^2 = 0.05$			$\sigma^2 = 0.5$		
# samples	ProGAdNet	Lasso	GENIE3	ProGAdNet	Lasso	GENIE3
50	0.206	0.023	0.018	0.106	0.018	0.014
100	0.288	0.025	0.028	0.202	0.021	0.022
200	0.356	0.030	0.039	0.280	0.024	0.031
300	0.380	0.031	0.044	0.289	0.026	0.038

Table 2. Top 20 MSigDB C2 gene sets that are enriched in the dNet gene set.

Gene sets	q-value
SMID_**BREAST**_**CANCER**_LUMINAL_A_UP	1.55E-27
NAKAYAMA_SOFT_TISSUE_TUMORS_PCA2_DN	1.73E-20
TURASHVILI_**BREAST**_**DUCTAL**_**CARCINOMA**_VS_DUCTAL_NORMAL_DN	1.40E-16
SMID_**BREAST**_**CANCER**_RELAPSE_IN_LUNG_DN	1.40E-16
POOLA_INVASIVE_**BREAST**_**CANCER**_UP	1.25E-12
SMID_**BREAST**_**CANCER**_RELAPSE_IN_BONE_UP	2.35E-12
TURASHVILI_**BREAST**_**DUCTAL**_**CARCINOMA**_VS_LOBULAR_NORMAL_DN	2.72E-12
VECCHI_GASTRIC_CANCER_ADVANCED_VS_EARLY_UP	3.04E-12
MCLACHLAN_DENTAL_CARIES_UP	3.04E-12
POOLA_INVASIVE_**BREAST**_**CANCER**_DN	3.04E-12
TURASHVILI_**BREAST**_**LOBULAR**_**CARCINOMA**_VS_DUCTAL_NORMAL_DN	9.32E-12
CROMER_TUMORIGENESIS_UP	1.26E-11
TURASHVILI_**BREAST**_**LOBULAR**_**CARCINOMA**_VS_LOBULAR_NORMAL_UP	1.52E-11
DOANE_**BREAST**_**CANCER**_ESR1_UP	1.52E-11
LIEN_**BREAST**_**CARCINOMA**_METAPLASTIC_VS_DUCTAL_DN	2.28E-11
SABATES_COLORECTAL_ADENOMA_DN	3.70E-09
JAEGER_METASTASIS_DN	4.31E-09
WALLACE_PROSTATE_CANCER_RACE_UP	4.46E-09
ANASTASSIOU_MULTICANCER_INVASIVENESS_SIGNATURE	4.68E-09
KORKOLA_TERATOMA	7.24E-09

Additional files

Additional file 1: the list of the TCGA IDs of 113 breast tumors and 113 normal tissue samples.

Additional file 2: the list of 268 genes that are involved in at least one changed network edges identified from the gene expression data of breast tumor and normal tissues.

Additional file 3: the list of 78 MSigDB C2 gene sets that are significantly enriched in the gene sets in Additional file 2.