OPEN ACCESS Freely available online

PLOS ONE

Inference of Gene Regulatory Networks from Genetic Perturbations with Linear Regression Model

Zijian Dong[1,2]*, Tiecheng Song[2], Chuang Yuan[3]

1 School of Electronic Engineering, Huaihai Institute of Technology, Lianyungang, Jiangsu, China, **2** School of Information Science and Engineering, Southeast University, Nanjing, Jiangsu, China, **3** Department of Health Technology and Informatics, The Hong Kong Polytechnic University, Hang Kong, China

Abstract

It is an effective strategy to use both genetic perturbation data and gene expression data to infer regulatory networks that aims to improve the detection accuracy of the regulatory relationships among genes. Based on both types of data, the genetic regulatory networks can be accurately modeled by Structural Equation Modeling (SEM). In this paper, a linear regression (LR) model is formulated based on the SEM, and a novel iterative scheme using Bayesian inference is proposed to estimate the parameters of the LR model (LRBI). Comparative evaluations of LRBI with other two algorithms, the Adaptive Lasso (AL-Based) and the Sparsity-aware Maximum Likelihood (SML), are also presented. Simulations show that LRBI has significantly better performance than AL-Based, and overperforms SML in terms of power of detection. Applying the LRBI algorithm to experimental data, we inferred the interactions in a network of 35 yeast genes. An open-source program of the LRBI algorithm is freely available upon request.

Citation: Dong Z, Song T, Yuan C (2013) Inference of Gene Regulatory Networks from Genetic Perturbations with Linear Regression Model. PLoS ONE 8(12): e83263. doi:10.1371/journal.pone.0083263

Editor: Alberto de la Fuente, Leibniz-Institute for Farm Animal Biology (FBN), Germany

Received April 30, 2013; **Accepted** November 1, 2013; **Published** December 23, 2013

Funding: This work was jointly supported by the National Natural Science Foundation of China (No. 61271207, No. 61174013), the Natural Science Foundation of Jiangsu Province, China (No. BK2011398), the Jiangsu Overseas Research \& Training Program for University Prominent Young \& Middle-aged Teachers and Presidents, and the Priority Academic Program Development of Jiangsu Higher Education Institutions. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: dzjian@126.com

Introduction

Exploring the structure of Gene Regulatory Networks (GRN) is a key element in understanding gene functions, especially in some complex diseases [1–3]. Direct experimental methods to explore the relationships among genes are time-consuming and labor-intensive. Statistical inference on GRN is a process of identifying gene interactions from limited experimental data using computational analysis, and is much more efficient.

Several models have been applied to describe the GRN. An intuitive and frequently applied method is to model the GRN as graphs [4–6], where the genes are considered as nodes and the interactions among them represented as edges. Several graphical methods, including directed acyclic graphs and directed cyclic graphs, have been proposed in [7–9]. GRN can also be modeled by the graphical Gaussian model [10], or the Bayesian network model [11]. Information theory, for instance, mutual information and synergy, can be also used to infer the GRN [12,13]. Due to high measurement cost of gene chip technology, only limited number of samples can be obtained. This limitation may result in low inference accuracy when applying synergy or mutual information to analyze the GRN.

In the last decade, Structural Equation Modeling (SEM) [14] has been used to infer GRN [9,15,16]. Exploiting genetic perturbation data and gene expression data, the work in [16] used SEM model via an adaptive Lasso based algorithm (AL-Based) to infer the networks. With simulations, the authors showed that the AL-based method had better performance than all other existing methods. With the two same types of data, Cai et. al. introduced a sparse SEM model, and stated that their Sparsity-aware Maximum Likelihood (SML) algorithm significantly outperformed all other algorithms, including the AL-based one [17,18].

In this paper, we also study the gene regulatory networks with SEM model using both genetic perturbation data and gene expression data, and transfer the SEM to a Linear Regression (LR) model through matrix transformation. In this transformation process, regulatory information in GRN will not be lost. Instead of ML approaches or classic Lasso methods, we propose an approach to infer the networks via the LR model by using a Bayesian method (LRBI). Simulations show that our LRBI algorithm is effective and reliable, and offers significantly better performance than the AL-based algorithm. Compared with SML, LRBI has significantly better performance in terms of power of detection, but has slightly worse performance in false discovery rate. LRBI also has the advantages that the estimation of the initial parameters and the consideration of the data sensitivity are not needed.

Model and Methods

The LR model for gene network inference

We consider m genes, n individuals' measurement using microarray. Without loss of generality, we assume that there are m makers. As in [9,16,17], the GRN obeys the form of SEM, where genes are the nodes, and interactions among genes are the edges, i.e.

$$\mathbf{P}=\mathbf{BP}+\mathbf{AX}+\mathbf{e}, \tag{1}$$

where $\mathbf{P}$ is an $m\times n$ matrix, p_{kj} is the jth expression level of the kth gene; $\mathbf{B}$ is an $m\times m$ matrix, defining the structure of the gene regulatory networks, b_{kj} is the regulatory effect of the jth gene on the kth gene; $\mathbf{X}$ is an $m\times n$ matrix, x_{kj} is the genotype of the kth marker in the jth perturbation; $\mathbf{A}$ is an $m\times m$ matrix representing the effect of each eQTL; $\mathbf{e}$ is an $m\times n$ matrix, and e_{kj} is the jth measurement noise of the kth gene. All elements in $\mathbf{e}$ are independent and identically distributed (i.i.d).

We assume that there is no self-loop of each gene, so that all diagonal entries of $\mathbf{B}$ are zeros. We also assume that each gene has its own corresponding QTL, and the loci of the m eQTLs have been determined by an existed method, but the effects of these eQTLs are unknown yet. Therefore $\mathbf{A}$ has m unknown entries, and all other entries are zeros. Without loss of generality, we assume that all the unknowns in $\mathbf{A}$ are the diagonal entries.

With the predetermined eQTLs matrix $\mathbf{X}$ and the gene expression data $\mathbf{P}$, the inference for GRN is to determine the unknown entries of $\mathbf{B}$ and $\mathbf{A}$ with appropriate optimization methods.

Since all the unknown parameters are in $(\mathbf{B},\mathbf{A})$, (1) can be written as follows

$$\mathbf{P}=\mathbf{\Pi\Omega}+\mathbf{e} \tag{2}$$

where $\mathbf{P}$ is still the $m\times n$ matrix defined above, $\mathbf{\Pi}=(\mathbf{BA})$, $\mathbf{\Omega}=\begin{pmatrix}\mathbf{P}\\ \mathbf{X}\end{pmatrix}$. We further rewrite (2) to

$$\begin{bmatrix}\mathbf{Y}_1\\ \vdots\\ \mathbf{Y}_k\\ \vdots\\ \mathbf{Y}_m\end{bmatrix}=\begin{bmatrix}\mathbf{\Lambda}_1\\ \vdots\\ \mathbf{\Lambda}_k\\ \vdots\\ \mathbf{\Lambda}_m\end{bmatrix}\mathbf{\Omega}+\begin{bmatrix}\mathbf{\varepsilon}_1\\ \vdots\\ \mathbf{\varepsilon}_k\\ \vdots\\ \mathbf{\varepsilon}_m\end{bmatrix} \tag{3}$$

where $\mathbf{Y}_k$ is the $k$throw of $\mathbf{P}$, $\mathbf{\Lambda}_k$ is the $k$throw of $\mathbf{\Pi}$, $\mathbf{\varepsilon}_k$ is the $k$throw of $\mathbf{e}$.

By the definition and the structure of (3), we can infer the parameters row by row. Therefore, the problem can be decomposed into

$$\mathbf{Y}_k=\mathbf{\Lambda}_k\mathbf{\Omega}+\mathbf{\varepsilon}_k, k=1,2,\ldots,m \tag{4}$$

In (4), the parameters that need to be inferred are $\Lambda_{i,j}, i,j=1,2,\ldots,m, i\neq j$, and $\Lambda_{i+m,i}, i=1,2,\ldots,m$.

Bayesian inference for the LR models

In gene regulatory networks, most entries of $\mathbf{\Lambda}_k$ are zeros, so $\mathbf{\Lambda}_k$ is sparse. Therefore, we assume that all entries of $\mathbf{\Lambda}_k$ follow Gaussian distribution with mean zeros. We also assume that entries of $\mathbf{\varepsilon}_k$ are i.i.d, and normally distributed with mean zeros and variance $\mathbf{\Psi}_k=\varphi_{\varepsilon k}\mathbf{I}$, where $\mathbf{I}$ is an $n\times n$ identity matrix.

With known $\mathbf{\Omega}$, the parameters to be estimated in (4) are $\mathbf{\theta}_k=(\mathbf{\Lambda}_k,\mathbf{\Psi}_k)$. The joint prior distribution can be factorized as:

$$p(\mathbf{\theta}_k)=p(\Lambda_k,\Psi_k)=p(\Psi_k)p(\Lambda_k|\Psi_k) \tag{5}$$

Rewriting (5) leads to

$$p(\mathbf{\theta}_k)=p(\varphi_{\varepsilon k})p(\Lambda_k|\varphi_{\varepsilon k}) \tag{6}$$

We assume that $\left(\mathbf{\Lambda}_k,\varphi_{\varepsilon k}^{-1}\right)$ has a joint prior distribution of Gaussian-Gamma [19], with

$$\varphi_{\varepsilon k}^{-1}\sim Gamma(\alpha_{0\varepsilon k},\beta_{0\varepsilon k}) \tag{7}$$

$$\Lambda_k|\varphi_{\varepsilon k}\sim Normal\left(\Lambda_{0k},\varphi_{\varepsilon k}\mathbf{H}_{0yk}\right) \tag{8}$$

where $\alpha_{0\varepsilon k},\beta_{0\varepsilon k},\mathbf{\Lambda}_{0k}$ are hyper parameters, should be preset to fixed values. $\mathbf{H}_{0yk}$ is a symmetric positive definite matrix. We will set it to an identity matrix in the implementation of algorithm for simplification.

The likelihood is

$$p(\mathbf{Y}_k|\Lambda_k,\varphi_{\varepsilon k},\Omega)\propto|\varphi_{\varepsilon k}|^{-\frac{n}{2}}\exp\left(-\frac{\varphi_{\varepsilon k}^{-1}}{2}\sum_{i=1}^{n}(y_{ki}-\Lambda_k\Omega_i)^2\right) \tag{9}$$

where Ω_i is the ith column of $\mathbf{\Omega}$. The joint posterior distribution of $(\mathbf{\Lambda}_k,\varphi_{\varepsilon k})$ is proportional to the product of the prior and the likelihood

$$p(\mathbf{\Lambda}_k,\varphi_{\varepsilon k}|\mathbf{P},\mathbf{\Omega})\propto p(\mathbf{\Lambda}_k,\varphi_{\varepsilon k})p(\mathbf{Y}_k|\mathbf{\Lambda}_k,\varphi_{\varepsilon k},\mathbf{\Omega}) \tag{10}$$

According to the prior distribution in (7∼8) and the likelihood in (9), the joint posterior distribution (10) can be written as

$$\begin{aligned}p(\Lambda_k,\varphi_{\varepsilon k}|\mathbf{P},\Omega)&\propto\varphi_{\varepsilon k}^{-(n/2+\alpha_{0\varepsilon k}-1)}\exp\left(-\beta_{0\varepsilon k}\varphi_{\varepsilon k}^{-1}\right)\\ &*\varphi_{\varepsilon k}^{-n/2}*\exp\left(-\frac{1}{2}\varphi_{\varepsilon k}^{-1}\begin{bmatrix}(\Lambda_k-\Lambda_{ok})\mathbf{H}_{oyk}^{-1}(\Lambda_k-\Lambda_{ok})^{\mathrm{T}}\\ +\\ \sum_{i=1}^{n}(y_{ki}-\Lambda_k\Omega_i)^2\end{bmatrix}\right)\\ &=p\left(\varphi_{\varepsilon k}^{-1}|\mathbf{P},\Omega\right)*p\left(\Lambda_k|\varphi_{\varepsilon k},\mathbf{P},\Omega\right)\end{aligned} \tag{11}$$

where

$$p\left(\varphi_{\varepsilon k}^{-1}|\mathbf{P},\mathbf{\Omega}\right)\sim Gamma\left[2^{-1}n+\alpha_{0\varepsilon k},\beta_{\varepsilon k}\right] \tag{12}$$

$$p(\mathbf{\Lambda}_k|\mathbf{P},\mathbf{\Omega})\sim Normal[\mathbf{a}_k,\varphi_{\varepsilon k}\mathbf{A}_k] \tag{13}$$

and

$$\mathbf{A}_k = \left(\mathbf{H}_{0yk}^{-1} + \mathbf{\Omega}\mathbf{\Omega}^{\mathrm{T}}\right)^{-1} \tag{14}$$

$$\mathbf{a}_k = \left[\mathbf{A}_k\left(\mathbf{H}_{0yk}^{-1}\mathbf{\Lambda}_{0k}^{\mathrm{T}} + \mathbf{\Omega}\mathbf{Y}_k^{\mathrm{T}}\right)\right]^{\mathrm{T}} \tag{15}$$

$$\beta_{\varepsilon k} = \beta_{0\varepsilon k} + 2^{-1}\left(\mathbf{Y}_k\mathbf{Y}_k^{\mathrm{T}} + \mathbf{\Lambda}_{0k}\mathbf{H}_{0yk}^{-1}\mathbf{\Lambda}_{0k}^{\mathrm{T}} - \mathbf{a}_k\mathbf{A}_k^{-1}\mathbf{a}_k^{\mathrm{T}}\right) \tag{16}$$

More details of the derivation can refer to Text S1. A similar result was shown in [14,20], where (12∼16) were parts of an iterative process to solve the Confirmatory Factor Analysis (CFA) model.

We can sample the posterior distributions in (12) and (13) to constitute an iterative process. Since the values in (14)∼(16) are all determined, the parameters of the Gamma distribution in (12), $2^{-1}n+\alpha_{0\varepsilon k}$ and $\beta_{\varepsilon k}$, are all fixed. As a result, the posterior distribution of $\varphi_{\varepsilon k}^{-1}$ will not be affected by the samples of $\mathbf{\Lambda}_k$. It is noted that only $\varphi_{\varepsilon k}^{-1}$,$\mathbf{\Lambda}_k$ can be sampled, therefore, the iterative process may be not effective and accurate. Thus, we modify the calculations of $\mathbf{a}_k$,$\beta_{\varepsilon k}$ in (15) and (16), and substitute $\mathbf{\Lambda}_{0k}$ by $\mathbf{\Lambda}_k$.

$$\mathbf{a}_k = \left[\mathbf{A}_k\left(\mathbf{H}_{0yk}^{-1}\mathbf{\Lambda}_k^{\mathrm{T}} + \mathbf{\Omega}\mathbf{Y}_k^{\mathrm{T}}\right)\right]^{\mathrm{T}} \tag{17}$$

$$\beta_{\varepsilon k} = \beta_{0\varepsilon k} + 2^{-1}\left(\mathbf{Y}_k\mathbf{Y}_k^{\mathrm{T}} + \mathbf{\Lambda}_k\mathbf{H}_{0yk}^{-1}\mathbf{\Lambda}_k^{\mathrm{T}} - \mathbf{a}_k\mathbf{A}_k^{-1}\mathbf{a}_k^{\mathrm{T}}\right) \tag{18}$$

The combination of (12, 13) and (17, 18) forms an iterative process. We execute this iterative process with a sufficient number of times, and until a steady state is reached. A sequence of sets of $\mathbf{\Lambda}_k^{(i)}$ are obtained by sampling from the posterior distribution in (13), which are then averaged to get the estimated parameters of $\mathbf{\Lambda}_k$. To get accurate results, we must guarantee that the iteration reaches its steady state. A simple stopping condition is to test the value of the square difference of the inferred parameters between two successive iterations, i.e. $\sum\left(\mathbf{\Lambda}_k^{(i+1)} - \mathbf{\Lambda}_k^{(i)}\right)^2$. If the difference is small enough (say $\tau < 0.001$), the iteration has reached a stable state. The choice of τ can influence the accuracy. The smaller τ is, the higher the accuracy of the parameter approximation is, naturally at the cost of more iterations and increased computational time.

The sketch of an algorithm is as follows:

Input the eQTLs matrix $\mathbf{X}$*, and the gene expression data* $\mathbf{P}$*; Set the initial hyperparameters* $\alpha_{0\varepsilon k} = n/5$, $\beta_{0\varepsilon k} = 1$, $\mathbf{H}_{0yk} = \mathbf{I}$. $\mathbf{\Lambda}_{0k}$ *is set to a* $1 * 2m$*vector, where only the* kth *entry is 1, all other entries are zeros.* $k = 1,2,\ldots,m$*; Assign a small value to* τ.

For $k = 1,2,\ldots,m$

Calculate $\mathbf{A}_k$,$\mathbf{a}_k$,$\beta_{\varepsilon k}$ *by (14∼16);*

Repeat:

1. *Get the sample of* $\varphi_{\varepsilon k}^{-1}$ *from the Gamma distribution by (12);*
2. *Get the sample of* $\mathbf{\Lambda}_k$ *from the Normal distribution by (13);*
3. *Calculate* $\mathbf{a}_k$,$\beta_{\varepsilon k}$ *by (17)(18);*
4. *Calculate* $S = \sum\left(\mathbf{\Lambda}_k^{(i+1)} - \mathbf{\Lambda}_k^{(i)}\right)^2$*;*
5. *If* $S < \tau$*,then end the iteration, else go to step 1;*

End for

More details of the algorithm implementation can refer to the software package F1.

It should be noted that it would be better to choose the second half of the samples and average them to get accurate result. The reason is that at the beginning of the iteration, the gap between the estimated $\mathbf{\Lambda}_k^{(i)}$ and the true values is large.

Results

Simulations

Let N_E be the number of edges in $\mathbf{B}$ (the original network), N_{IE} be the number of edges in $\mathbf{B}'$ (the inferred network), N'_{false} be the number of edges which exist in $\mathbf{B}'$ but not in $\mathbf{B}$, N'_{true} be the number of edges which exist in both $\mathbf{B}'$ and $\mathbf{B}$, therefore $N'_{false} + N'_{true} = N_{IE}$. Define Power of Detection $PD = N'_{true}/N_E$, and False Discovery Rate $FDR = N'_{false}/N_{IE}$.

Logsdon and Mezey [16] had shown that the AL-based algorithm outperformed the PC-algorithm [21,22], the QTLnet algorithm [23], and had comparable performance with the QDG algorithm [24]. Cai et al. stated that their SML algorithm offered significantly better performance than the AL-based algorithm and the QTL algorithm in PD and FDR [17,18]. Therefore, we shall compare our LRBI algorithm with SML and AL-based.

Firstly, we carried out simulations following the setups in [16]. We simulated two types of directed acyclic gene networks: one with 10 genes and the other one with 30 genes. Averaged $N_e = 3$ edges were created per gene, which meant that there were on average 3 edges created between one gene and all other genes. If an edge existed from node j to node i, then b_{ij} was sampled from a uniform distribution on the interval $(-1-0.5)\cup(0.51)$; otherwise b_{ij} was set to 0. Entries of $\mathbf{X}$ took values from the set $\{1,2,3\}$with the corresponding probabilities 0.25, 0.5, and 0.25 respectively. Each gene has its own corresponding QTL, and $\mathbf{A}$ is assumed to be an identity matrix. Each entry of $\mathbf{e}$ in (1) was sampled from a Gaussian distribution $N(0,0.01)$. $\mathbf{P}$ was calculated by (1).

We generated cyclic or acyclic networks for simulations, and used LRBI to infer the parameters of the simulated networks. For cyclic networks,LRBI can obtain the steady-state solutions naturally. By inference, the steady regulatory relations can be got, if some cyclic regulatory relations among genes existed.

Due to the inference characteristic of Bayesian methods, the estimated parameters are not regressed to zeros as in Lasso methods. Therefore, an edge from gene j to gene i is considered to be present if $\left|b'_{ij}\right| > 0.05$, otherwise, there is no edge from gene j to gene i.

Simulation results for the setups described above are shown in Figure 1, where (a) and (b) are for the gene network of $m = 10$, (c) and (d) are for the gene network of $m = 30$. LRBI has a better performance than SML in terms of PD, but SML outperforms LRBI algorithm in terms of FDR. Both LRBI and SML significantly outperform the AL-Based algorithm in terms of PD and FDR. The PD of LRBI reaches 1 when the number of samples is 20 or more for both the two scenarios $m = 10$ and $m = 30$.

Secondly, we simulated two types of directed cyclic gene networks: one with 10 genes and the other one with 30 genes. Averaged $N_e = 3$ edges were created per gene. We employed the same procedure used in the acyclic scenario to generate

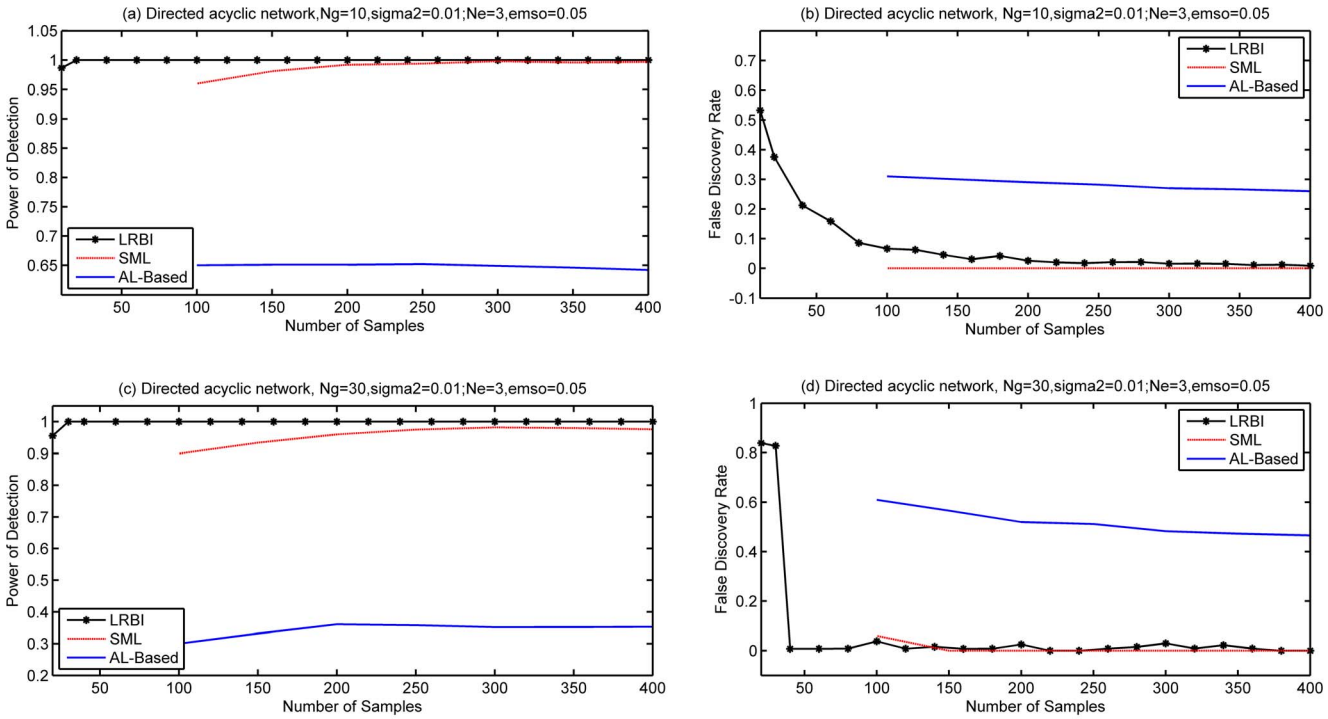

Figure 1. Performance of LRBI for directed acyclic networks. The performance of SML and AL-Based algorithms is also shown for comparisons. The average number of edges per node is $N_e=3$, the variance of noise is 0.01, and no edge exists if $\left|B'_{i,j}\right|\leq 0.05$ for decision. (a) and (b) are for a gene network of $m=10$, (c) and (d) are for a gene network of $m=30$.
doi:10.1371/journal.pone.0083263.g001

B,**A**,**X**,**P**,**e**. Simulation results are shown in Figure 2, where (a) and (b) are for the gene network of $m=10$, (c) and (d) are for the gene network of $m=30$. LRBI has significantly better performance than SML in terms of PD, and SML outperforms LRBI algorithm in terms of FDR. When the number of samples is large enough, the FDRs of LRBI and SML are all close to zeros. Both LRBI and SML significantly outperform the AL-Based algorithm in PD and FDR.

Thirdly, we simulated the impact of different decision thresholds on performances. We used a bigger network $m=100$. Averaged $N_e=3$ edges were created per gen, and the variance of noise was 0.01. Three decision thresholds $\xi=0.05,0.1,0.2$ were simulated. In simulations, if we found that $\left|b'_{ij}\right|\leq\xi$, then we set $b'_{ij}=0$. A directed acyclic network and a directed cyclic network were simulated and the results were separately shown in Figure 3 (a) (b) and Figure 3 (c) (d).

We continued the simulations with a even bigger network with $N_e=3$, $m=300$ to study the impact of decision thresholds on performance. The variance of noise was 0.01. Two decision thresholds $\xi=0.1$ and $\xi=0.2$ were simulated respectively. Both directed acyclic network and directed cyclic network were simulated, and the results were separately shown in Figure 4 (a) (b) and Figure 4 (c) (d). As confirmed by Figure 3 and Figure 4, a large decision threshold can reduce the FDR, but it also lowers the PD. Therefore, the decision thresholds used in simulations or applications should be chosen carefully.

Finally, we evaluated the impact of noise levels on the performance of LRBI. Here, we used networks with $m=100$, $N_e=3$. Again, we applied LRBI to both directed acyclic and directed cyclic networks. The variance of noise was set to 0.01, 0.05, and 0.1 respectively. The simulation results are shown in Figure 5, where (a) and (b) are for directed acyclic network, (c) and (d) are for directed cyclic network. We find that the PD performance is always excellent, but the FDR of LRBI is worse when the noise level increases, even when the number of samples is relatively large.

We have stated that LRBI cannot infer parameters to zeros automatically, but can infer them with high precision. In most of the simulations we conducted, the decision thresholds are 0.05. That is to say, if the value inferred is lower than 0.05, the entry is considered to be zero. This implies that the numerical difference between the inferred value and the original value is less than 0.05 for most of the entries in regulatory networks. We define that numerical difference as $INEr(i,j)=\left|b_{ij}-b'_{ij}\right|$. Through simulations, we found that *INEr* was also very small for the entry whose value is nonzero in the original network. This feature is very meaningful, because the inferred parameters can accurately indicate the regulatory relationships among genes. An acyclic network was simulated, with $m=30,N_e=3$, sigma2 = 0.01, decision threshold $\xi=0.1$. Some results are shown in Table 1.

Case study

Here, we applied LRBI to infer the gene regulatory networks using the gene expression data and the genetic makers, which were assayed in 112 segregants of a cross between the yeast strains BY4716 and RM11-1a [25]. The cross had 5727 genes with small number of samples, so a pretreatment process was needed to select strong cis-eQTLs and interactions among genes [16]. We dealt with the filtered dataset provided by Logsdon [16], in which only 35 genes were used. The 35 yeast genes are SEO1, NUP60, RCY1, IRC18, TPK3, PHD1, JLP1, SNF7, PCD1, RPL19A, SEN1, OST6, BUB2, BUL1, PHA2, ORC5, FYV6, SLM2, HAL9, RDL1, POC4, ASA1, ECM13, TYR1, RNQ1, SFA1,

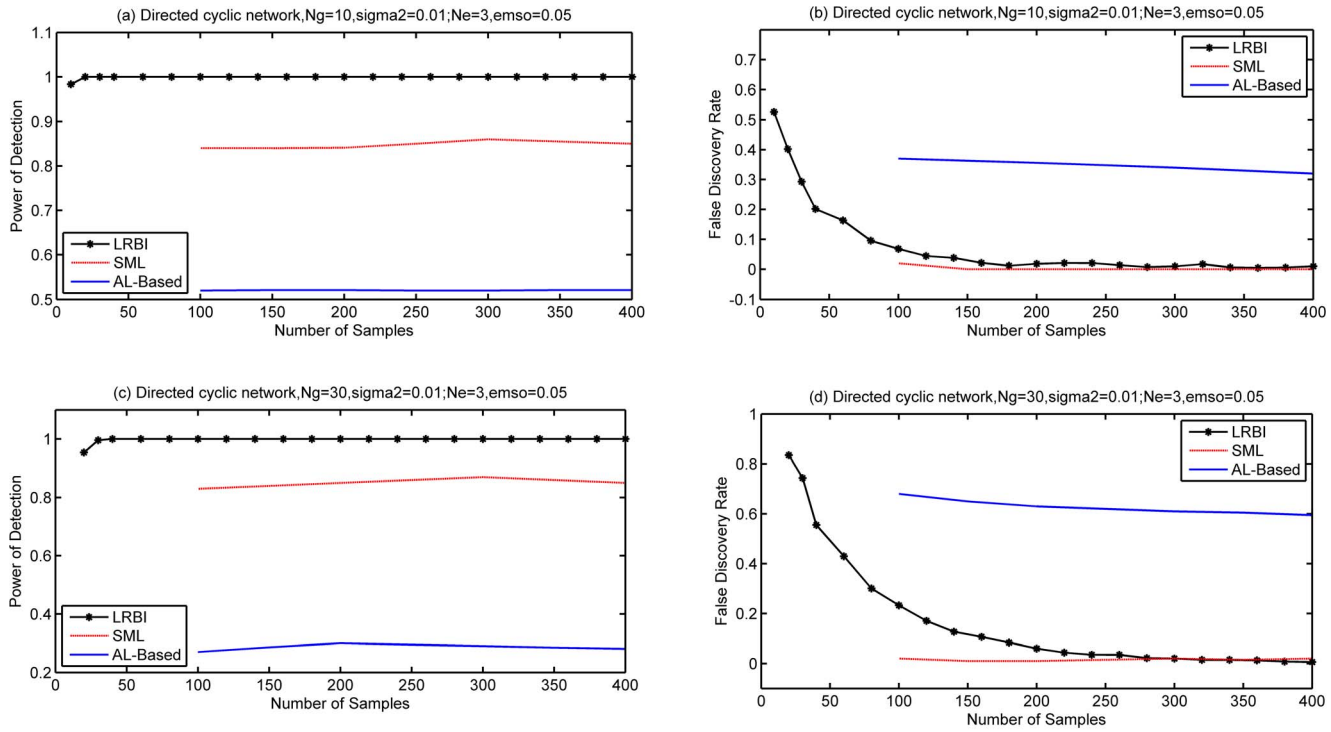

Figure 2. Performance of our LRBI algorithms for directed cyclic networks. The performance of SML and AL-Based algorithms is also shown for comparisons. The average number of edges per node is $N_e = 3$, the variance of noise is 0.01, and no edge exists if $\left|B_{ij}^{'}\right| \leq 0.05$ for decision. (a) and (b) are for a gene network of $m = 10$, (c) and (d) are for a gene network of $m = 30$.
doi:10.1371/journal.pone.0083263.g002

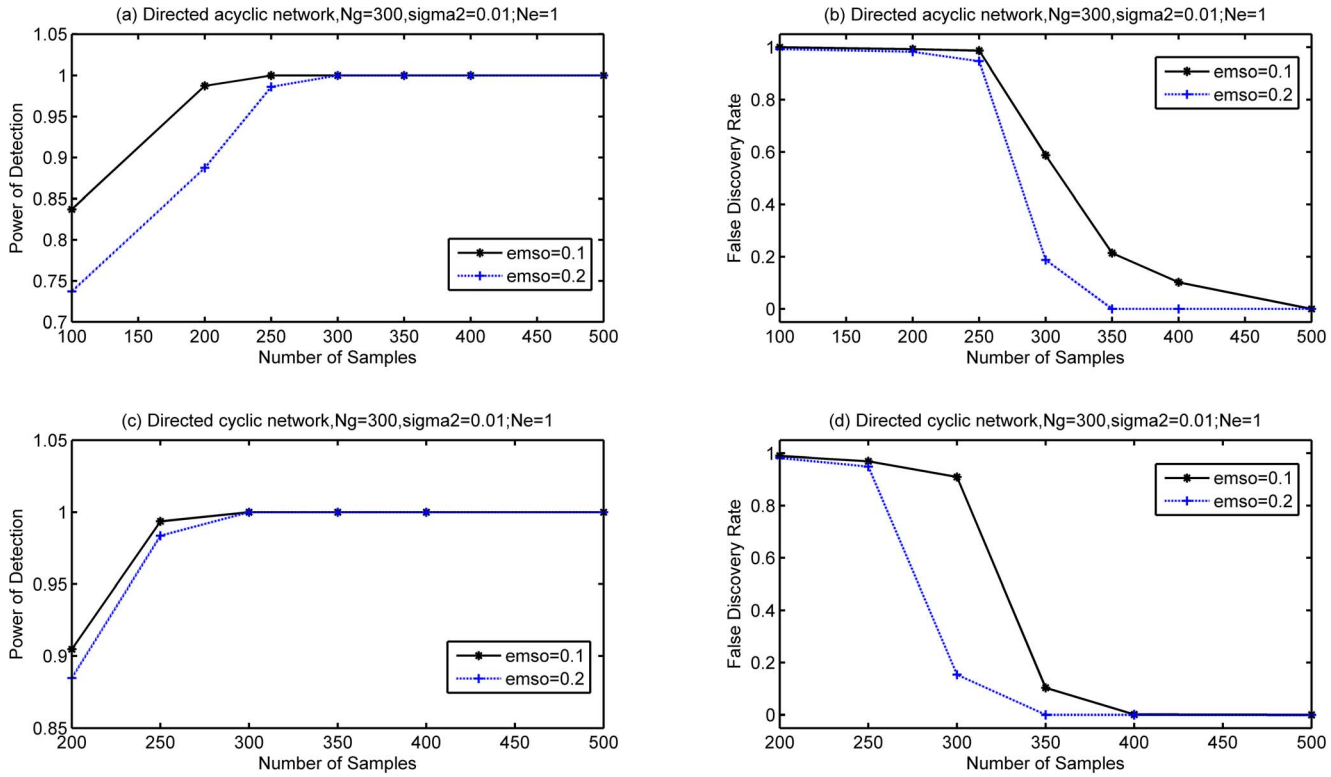

Figure 3. Performance of LRBI algorithms for various decision thresholds. Two network cases are simulated to find the impact of decision thresholds on PD and FDR. (a) and (b) are for directed acyclic networks, (c) and(d) are for directed cyclic networks; $m = 100, N_e = 3$, the variance of noise is 0.01. Thresholds are 0.05, 0.1, 0.2 (emso in figure).
doi:10.1371/journal.pone.0083263.g003

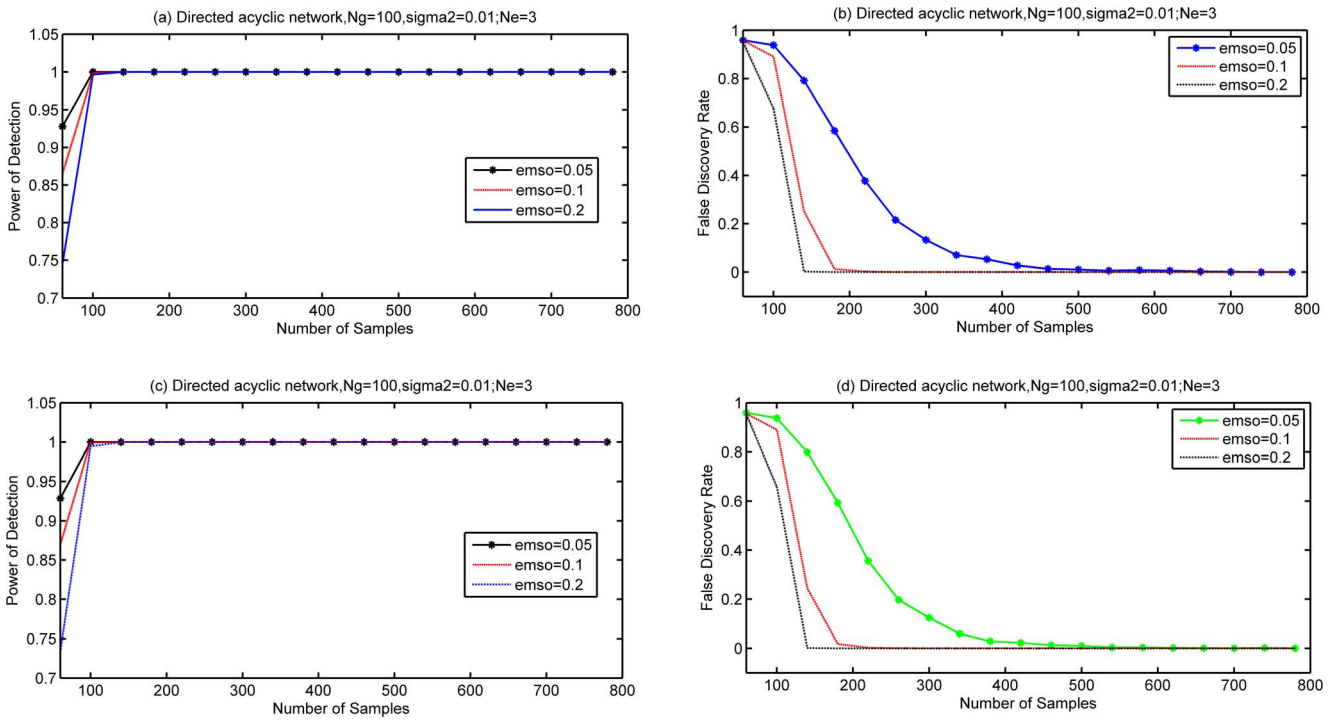

Figure 4. Performance of LRBI algorithms for various decision threshold with $m=300$. Two network cases are simulated to find the impact of decision thresholds on PD and FDR. (a) and (b) are for directed acyclic networks, (c) and(d) are for directed cyclic networks; $m=300$, $N_e=3$, the variance of noise is 0.01.Thresholds are 0.1 and 0.2.
doi:10.1371/journal.pone.0083263.g004

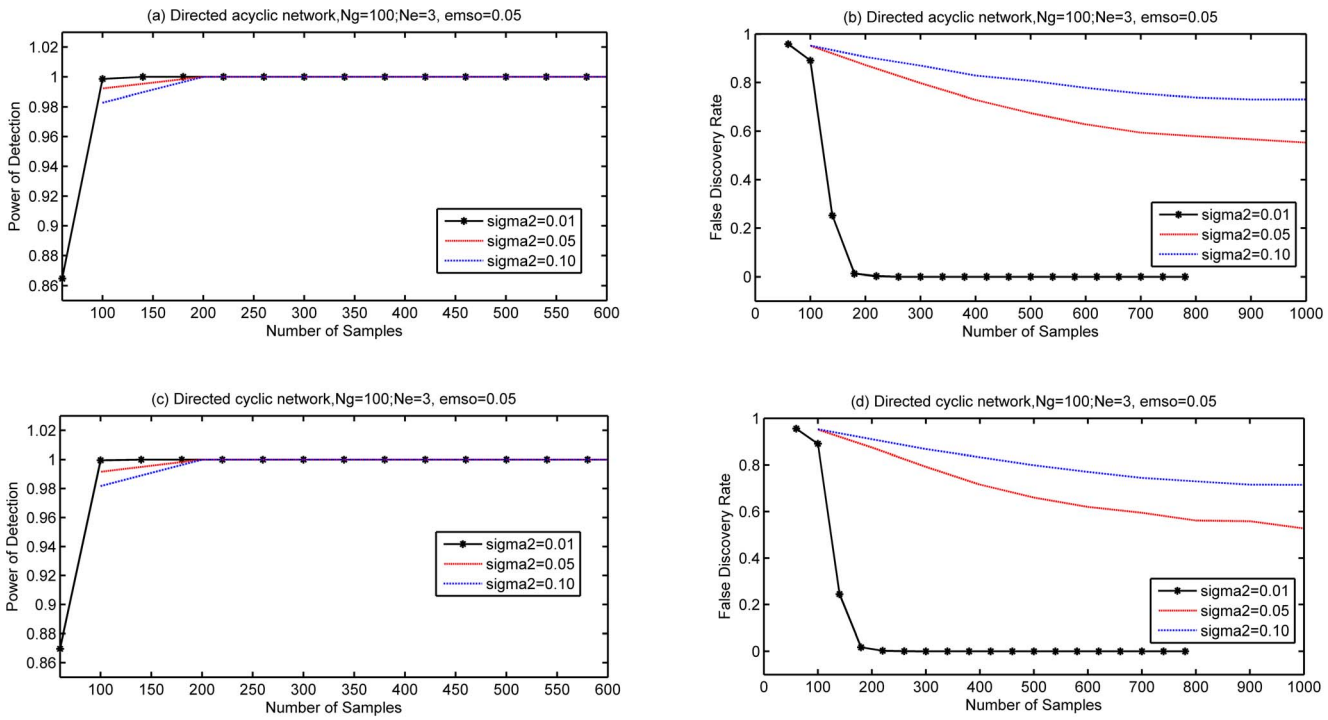

Figure 5. Performance of LRBI algorithm under various noise levels. Two network cases are simulated to find the impact of noise level on PD and FDR. (a) and (b) are for directed acyclic networks, (c) and (d) are for directed cyclic networks; $m=100$, Decision thresholds is 0.05. Three noise levels are simulated.
doi:10.1371/journal.pone.0083263.g005

Table 1. Some INErs of a network inferred by LRBI.

(i,j)	(1,8)	(1,17)	(1,26)	(1,28)	(2,17)	(2,19)	(3,1)	(3,5)	(3,23)	(3,28)	(4,2)	(4,12)
B(i,j)	0.9749	0.8016	−0.5050	−0.9123	−0.9925	0.9274	−0.7733	0.8633	0.5397	0.9918	0.8308	−0.6049
B′(i,j)	0.9301	0.7725	−0.4812	−0.8379	−0.9913	0.8959	−0.7452	0.8412	0.5183	0.9577	0.8081	−0.5974
INEr(i,j)	0.0448	0.0291	0.0238	0.0744	0.0012	0.0315	0.0281	0.0221	0.0214	0.0341	0.0227	0.0075

The network is an acyclic network with m = 30, Ne = 3, sigma2 = 0.01, decision threshold is 0.1.
doi:10.1371/journal.pone.0083263.t001

PRM7, SAN1, HIM1, YEL073C, SAP1, SNZ3, MST27, YHR054C, DAL7.

With 112 samples for these 35 genes, and the eQTLs data, we inferred the regulatory network as shown in Figure 6. It is noted that our algorithm doesn't need to assume the network is cyclic or acyclic. There are 145 edges in the inferred network. A total of 31 genes are regulators of at least one target, and 32 genes have at least one regulator. A total of 28 genes occur both as regulators and targets.

There were only 4 instances of reciprocal regulation (two genes act on each other) presented: $\mathrm{ORC5} \underset{-0.6901}{\overset{-0.5113}{\rightleftarrows}} \mathrm{SNF7}$, $\mathrm{CM13} \underset{-0.3897}{\overset{-0.4259}{\rightleftarrows}} \mathrm{YL14A}$, $\mathrm{YHL4} \underset{-0.4000}{\overset{-0.3952}{\rightleftarrows}} \mathrm{RDL1}$, $\mathrm{DAL7} \underset{0.4959}{\overset{0.3480}{\rightleftarrows}} \mathrm{HIM1}$.

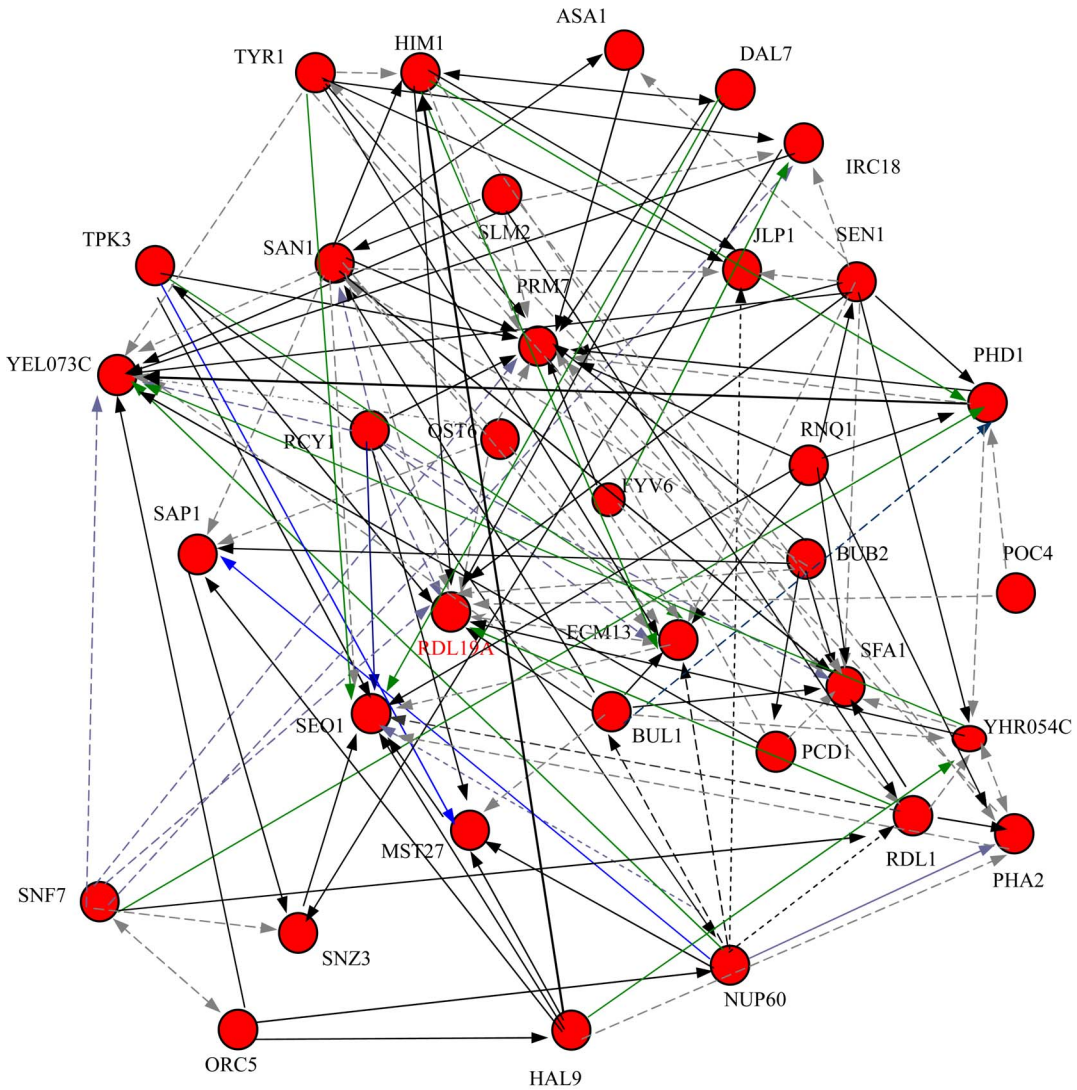

Figure 6. Regulatory network reconstruction for the 35 genes. These genes are filtered out by the methods as in [16]. In the figure, a solid line denotes that the interaction between two genes is positive regulatory, while a dotted line denotes a negative regulatory. Some color lines are used to make the figure clear. There are 145 regulator–target pairs, among which, 78 pairs are positive regulations, and 67 pairs are negative regulations.
doi:10.1371/journal.pone.0083263.g006

Among the 145 regulator–target pairs, there are 78 positive regulations, and 67 negative regulations. To verify the inference result, we used the Generate Regulation Matrix tool in the website of YEASTRACT [26] to create the gene regulatory network with the 35 selected yeast genes described above. In the network generated by the tool, there are only three regulatory relationship, HIM1 regulated by HAL9 [27], YEL073C regulated by PHD1 [27], and FYV6 regulated by PHD1 [28]. From the experimental yeast data, we deduced two out of the three regulatory relationships, HIM1 regulated by HAL9, YEL073C regulated by PHD1.

Conclusion and Discussion

We modeled the gene regulatory networks by using a LR model, and proposed a Bayesian method to complete the inference. We conducted a series of simulations to evaluate the performance of the proposed algorithm LRBI, and compared LRBI with another two algorithms, the AL-Based and the SML algorithms. LRBI had a significantly better performance than AL-based regarding to both PD and FDR. Compared to SML, LRBI showed a better performance in PD and slightly worse in FDR. This feature of LRBI makes more sense. Considering two cases, one is that we can find less false edges but loss more true edges, the other one is that we can find more, or even all true edges among genes, but with slightly more false edges, the latter one is more meaningful.

The proposed algorithm was accurate, and the gap between the inferred and the original parameters was less than 5% (even 2%) in most case. The proposed algorithm was also very effective. We inferred the GRN of the 35 yeast genes in a short time (1.2 seconds in a laptop), while for the SML algorithm, a program error occurred after about 52 minutes' run with the same 35 yeast genes data set. LRBI also had the benefit that the dependency of the performance on the estimates of initial parameters is not strong. For simplicity, we just assign some constants to these parameters in simulations and case studies. Therefore, the LR model and the LRBI algorithm can provide an effective way of exploiting both gene expression and perturbation data to infer GRN.

The reason our method seemed to perform better was that, LRBI fully exploited the structure of the SEM, and transformed it into a linear regression model without information loss, while AL-Based only partly exploited the structure of the SEM and used the adaptive Lasso to infer the networks, so LRBI was more effective. LRBI used the Bayesian method, while SML essentially used the maximum likelihood method to infer the GRN, therefore SML was not efficient and sensitive to data. However, there are many other methods for linear regression problems, such as hierarchical Bayesian, variational approximation, and so on. These methods can potentially improve the inference accuracy of GRN with the linear regression model proposed by this paper.

However, the FDR of LRBI is considerably high when the noise level is large, and another issue is the ability of dealing with large-scale gene networks. Thus, a future work is to decrease FDR in high-noise context, and apply new strategies to handle large-size gene networks.

Supporting Information

Text S1 Bayesian Inference for the Linear Regression Model.
(PDF)

Software S1 Software package implementing the LRBI algorithm.
(RAR)

Acknowledgments

We would like to thank Logsdon, B.A, for his valuable suggestions and providing the filtered yeast gene expression data.

Author Contributions

Conceived and designed the experiments: ZJD. Performed the experiments: ZJD. Analyzed the data: ZJD CY. Wrote the paper: ZJD TCS.

References

1. Davidson E, Levin M (2005) Gene regulatory networks. Proceedings of the National Academy of Sciences of the United States of America 102: 4935–4935.
2. Davidson EH, Erwin DH (2006) Gene regulatory networks and the evolution of animal body plans. Science Signalling 311: 796.
3. Olson EN (2006) Gene regulatory networks in the evolution and development of the heart. Science Signalling 313: 1922.
4. Friedman N (2004) Inferring cellular networks using probabilistic graphical models. Science Signalling 303: 799.
5. Toh H, Horimoto K (2002) Inference of a genetic network by a combined approach of cluster analysis and graphical Gaussian modeling. Bioinformatics 18: 287–297.
6. Werhli AV, Grzegorczyk M, Husmeier D (2006) Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical Gaussian models and Bayesian networks. Bioinformatics 22: 2523–2531.
7. Kramer N, Schafer J, Boulesteix AL (2009) Regularized estimation of large-scale gene association networks using graphical Gaussian models. Bmc Bioinformatics 10.
8. Li RH, Tsaih SW, Shockley K, Stylianou IM, Wergedal J, et al. (2006) Structural model analysis of multiple quantitative traits. Plos Genetics 2: 1046–1057.
9. Liu B, de la Fuente A, Hoeschele I (2008) Gene network inference via structural equation modeling in genetical genomics experiments. Genetics 178: 1763–1776.
10. Schafer J, Strimmer K (2005) An empirical Bayes approach to inferring large-scale gene association networks. Bioinformatics 21: 754–764.
11. Husmeier D (2003) Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks. Bioinformatics 19: 2271–2282.
12. Watkinson J, Wang X, Zheng T, Anastassiou D (2008) Identification of gene interactions associated with disease from gene expression data using synergy networks. BMC systems biology 2: 10.
13. Anastassiou D (2007) Computational analysis of the synergy among multiple interacting genes. Molecular systems biology 3.
14. Lee S-Y (2007) Structural equation modeling: A Bayesian approach: Wiley.
15. Xiong M, Li J, Fang X (2004) Identification of genetic networks. Genetics 166: 1037–1052.
16. Logsdon BA, Mezey J (2010) Gene Expression Network Reconstruction by Convex Feature Selection when Incorporating Genetic Perturbations. Plos Computational Biology 6.
17. Cai X, Bazerque JA, Giannakis GB (2011) Gene network inference via sparse structural equation modeling with genetic perturbations. Genomic Signal Processing and Statistics (GENSIPS), IEEE International Workshop on. pp. 66–69.
18. Cai X, Bazerque JA, Giannakis GB (2013) Inference of gene regulatory networks with sparse structural equation models exploiting genetic perturbations. PLoS computational biology 9: e1003068.
19. DeGroot MH (1970) Optimal Statistical Decisions. New York: McGraw-Hill.
20. Broemeling LD (1985) Bayesian Analysis of Linear Models. New York: Marcel Dekker Inc.
21. Zheng XY, Liu ZH, Liu YC (2003) A PC algorithm for dynamic actuated traffic control system. 2003 Ieee Intelligent Transportation Systems Proceedings, Vols 1 & 2: 856–860.
22. Li JN, Wang ZJ (2009) Controlling the False Discovery Rate of the Association/Causality Structure Learned with the PC Algorithm. Journal of Machine Learning Research 10: 475–514.
23. Neto EC, Keller MP, Attie AD, Yandell BS (2010) Causal Graphical Models in Systems Genetics: A Unified Framework for Joint Inference of Causal Network and Genetic Architecture for Correlated Phenotypes. Annals of Applied Statistics 4: 320–339.
24. Chaibub Neto E, Ferrara CT, Attie AD, Yandell BS (2008) Inferring causal phenotype networks from segregating populations. Genetics 179: 1089–1100.

25. Stranger BE, Forrest MS, Dunning M, Ingle CE, Beazley C, et al. (2007) Relative impact of nucleotide and copy number variation on gene expression phenotypes. Science 315: 848–853.
26. Yeastract website. Available: http://www.yeastract.com/index.php. Accessed 2013 Sept 9.
27. Lee TI, Rinaldi NJ, Robert F, Odom DT, Bar-Joseph Z, et al. (2002) Transcriptional regulatory networks in Saccharomyces cerevisiae. Science Signaling 298: 799.
28. Borneman AR, Leigh-Bell JA, Yu H, Bertone P, Gerstein M, et al. (2006) Target hub proteins serve as master regulators of development in yeast. Genes & development 20: 435–448.