BMC Bioinformatics

RESEARCH ARTICLE Open Access

Gene regulatory network inference using PLS-based methods

Shun Guo[1,2], Qingshan Jiang[2], Lifei Chen[3] and Donghui Guo[1*]

Abstract

Background: Inferring the topology of gene regulatory networks (GRNs) from microarray gene expression data has many potential applications, such as identifying candidate drug targets and providing valuable insights into the biological processes. It remains a challenge due to the fact that the data is noisy and high dimensional, and there exists a large number of potential interactions.

Results: We introduce an ensemble gene regulatory network inference method PLSNET, which decomposes the GRN inference problem with *p* genes into *p* subproblems and solves each of the subproblems by using Partial least squares (PLS) based feature selection algorithm. Then, a statistical technique is used to refine the predictions in our method. The proposed method was evaluated on the DREAM4 and DREAM5 benchmark datasets and achieved higher accuracy than the winners of those competitions and other state-of-the-art GRN inference methods.

Conclusions: Superior accuracy achieved on different benchmark datasets, including both *in silico* and *in vivo* networks, shows that PLSNET reaches state-of-the-art performance.

Keywords: Gene Regulatory Network inference, Gene expression data, Partial least squares (PLS), Ensemble

Background

Deciphering the structure of the gene regulatory networks (GRNs) [1] is crucial for bioinformatics, as it provide insight on the development, functioning and pathology of biological organisms. With the advent of high-throughput technologies such as next-generation sequencing, it has become relatively easy to measure chromatin state and gene expression genome-wide. Gene expression data obtained from high-throughput technologies correspond to the expression profiles of thousands of genes, which reflect gene expression levels for different replicates or experimental conditions (e.g., physicochemical, temporal and culture medium conditions). As a consequence, many methods have been proposed to solve the GRN reverse engineering problem by using gene expression data [2–5].

However, inferring the GRN from gene expression data remains a daunting task due to the large number of potential interactions, the small number of available measurements and the high dimensional, noisy data. Methods based on the statistical analysis of dependencies have been applied to the inference of GRNs, such as the method proposed in [6], which uses correlation coefficients to define the gene similarity metric for inferring the GRNs. One weakness of this method is that correlation coefficients fail to identify more complex statistical dependencies (e.g., non-linear ones) between genes. Thus, information theoretic measures have been proposed to capture more complex dependencies. In particular, these methods use mutual information (MI) between a pair of genes as a measure to infer networks [7]. As the existence of indirect interactions in relevance network, some refinements have been proposed to correct the predictions. For example, the CLR method [8] eliminates indirect influences based on the empirical distribution of all mutual information scores. The ARACNE method [9] was also designed to filter out indirect interactions by using the Data Processing Inequality. C3NET [10] and its extension BC3NET [11] correct the predictions based on estimates of mutual information values in conjunction with a maximization step. The ANOVerence method [12] includes meta-information of the microarray chips to guide the network inference process and uses η_2 score as an alternative measure to

* Correspondence: dhguo@xmu.edu.cn
[1] Department of Electronic Engineering, Xiamen University, Fujian 361005, China
Full list of author information is available at the end of the article

evaluate dependencies between genes, where η_2 score is a correlation coefficient derived using ANOVA.

The methods [13–15] based on probabilistic graphical models (e.g., Bayesian networks) have been widely used to infer GRNs. However, Bayesian networks do not allow the presence of feedback loops. Dynamic Bayesian networks [16, 17] are able to overcome the limitation while they can only handle time-series expression data. Moreover, learning the structure of a Bayesian network is a daunting task both from a computational and theoretical point of view [18]. Comparisons of existing GRN inference methods and detailed reviews can be referred in [4, 19].

Recently, some ensemble methods [18, 20–22] formalized the GRN inference problem as a feature selection problem and show interesting performance. The GENIE3 method [18], which is based on feature selection with ensembles of random forests, is recognized as state-of-the-art on some benchmarks [19]. As using random forests for feature selection is not well understood theoretically, the TIGRESS method [20] uses least angle regression (LARS) with stability selection combined to solve the GRN inference problem. The ENNET method [21] aggregates the features selected by an algorithm based on Gradient Boosting Machine. However, the ENNET method has high computational cost when it is applied on the high-dimensional data (i.e., the data with thousands of features). The NIMEFI method [22] explores the potential of several ensemble methods, such as GENIE3, Ensemble Support Vector Regression (E-SVR) and Ensemble Elastic Net [23] (E-EL), and combines the predictions of these methods under a general framework. However, NIMEFI has more adjustable parameters than other ensemble GRN inference methods, which increases the uncertainties of the model.

In this paper, we propose a new ensemble GRN inference method based on partial least squares (PLS). The method casts PLS-based feature selection algorithm into an ensemble setting by taking random potential regulatory genes. Then, we use a statistical technique to refine the predictions in our method by taking into account the impact of hub regulatory gene (i.e., a regulatory gene regulates many target genes). Various evaluations of techniques have been performed in the context of DREAM (Dialogue for Reverse Engineering Assessments and Methods) challenges [24], which aims to provide researchers with benchmark datasets to validate their work. Hence, we compare the performance of our method to several state-of-the-art methods in DREAM4 [25, 26] and DREAM5 [27] gene reconstruction challenge, and the results show our method performs competitively.

Methods

Problem definition

We focus on inferring the directed topology of GRNs using gene expression data in this paper. As input data, we consider gene expression measurements for p genes in n experimental conditions. The same as many ensemble methods (e.g., GENIE3, TIGRESS, ENNET and NIMEFI), we use a general framework for GRNs inference problem, which does not take the information of different experimental conditions (e.g., gene-knockouts, perturbations and even replicates) into account. The gene expression data D is defined as following:

$$D = [x_1, \dots, , x_p] \in R^{n \times p} \quad (1)$$

where x_i is a column vector of expression values of i-th gene in n experimental conditions.

GRN inference methods aim to make a prediction of the regulatory links between genes from gene expression data D. Most methods provide a ranking list of the potential regulatory links from the most to the less confident. Then, a network is automatically obtained by selecting a threshold value on this ranking. As it is beneficial to the end-user to explore the network at all sorts of threshold levels [22], we focus only the ranking task in this paper. It should be noted that the ranking is the standard prediction format of the DREAM challenges, where the challenges have been widely used to evaluate various GRN inference methods.

In order to infer the regulatory network from the expression data D, we compute a score w_{ij} for a potential edge directed from gene i to gene j, where the edge indicates that gene i regulates gene j on expression level and the score w_{ij} represents the strength that gene i associates (i.e., regulates) gene j.

Network inference with feature selection methods

Motivated by the success of ensemble methods based on feature selection (e.g., GENIE3 and TIGRESS), we decomposed the GRN inference problem with p genes into p subproblems, where each of these subproblems can be viewed as a problem of feature selection in statistics [18, 28]. More specifically, for each target gene, we wish to determine the subset of genes which directly influence the target gene from the expression level. Let D is the gene expression data defined in (1), the i-th gene is the target gene, and we define candidate regulators containing expression values in n experimental conditions as:

$$x^{-i} = [x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_p] \quad (2)$$

and the feature selection problem can be defined as:

$$x_i \;=\; f\left(x^{-i}\right) \;+\; \varepsilon, \forall i \in \{1, 2, \dots, p\} \tag{3}$$

where f is a regression function exploits the expression in x^{-i} of genes that are directly connected to gene i, and ε is some noise. Usually, f can be defined as:

$$f\left(x^{-i}\right) \;=\; \Sigma_j w_{ji} x_j,\; \forall j \in \{1, \dots, i-1, i \;+\; 1, \dots, p\} \tag{4}$$

where $w_{ji} \geq 0$ represents the strength that gene i associates (i.e., regulates) gene j. The rankings of the regulatory links of gene i is obtained by computing the w_{ji}. By aggregating the p individual gene rankings, we can get a global ranking of all regulatory links.

GRN inference with PLS-based ensemble methods

Recently, as PLS (Partial Least Squares) has been exploited by several authors to address the problem of feature selection for classification and showed interesting performance, such as TotalPLS [29] and KernelPLS [30], in this paper, we also use the PLS based method to solve the problem defined in (3). One difficulties of GRN inference problem is that we do not know how many candidate regulatory genes are sufficient to provide a good model for a target gene. For the purpose, we use PLS-based ensemble method. The basic idea is that the w_{ji} for gene i is computed by running PLS-based feature selection method many times, resampling the samples and selecting random K candidate regulatory genes at each run. We discuss and explore the effect of K values on the method performance in the Results Section.

Feature selection with PLS-based method

Let $X = [x_1, \dots, x_p] \in R^{n \times p}$ be a matrix that has been normalized to have a mean of zero and $Y = [y_1, \dots, y_n]^T \in R^{n \times 1}$ be a column vector that has been normalized to have a mean of zero. PLS aims to find a pair of projection directions w and u such that the projections $P = Xv$ (i.e., PLS components) and $Q = Yu$ can carry as much information on variation as possible in X and Y [31]. The projections P and Q can be obtained by solving the criterion function as:

$$\begin{cases} max\ J(v,\ u) \;=\; \dfrac{\left(v^T \Sigma_{\mathrm{XY}} u\right)^2}{v^T v \cdot u^T u} \\ \quad s.t.\ \ v^T v = u^T u = 1. \end{cases} \tag{5}$$

where $\Sigma_{XY} = cov^2(X, Y)$ is the covariance matrix for the vectors of X and Y.

The common solutions to PLS-based model include Non-linear Iterative PLS (NIPALS) [32] and Statistically Inspired Modification of PLS (SIMPLS) [33]. As SIMPLS is slightly superior to NIPALS and is computationally efficient, our analysis and calculation is based on SIMPLS in this paper. PLS components P are constructed to maximize the objective function based on the sample covariance between Y and $P = Xv$. Let m be the number of components, SIMPLS is able to calculate $v_1, v_2, \dots, v_m$ by solving the objective function as follow:

$$\begin{cases} maxJ(v) \;=\; cov^2(Xv_i,\ \ Y) \\ \quad s.t.\ \|v_i\| \;=\; 1; \\ v_i^T\left(X^T X\right) v_j \;=\; 0; \\ j \;=\; 1, \dots, i \;-\; 1. \end{cases} \tag{6}$$

The component $P_i = Xv_i$, which extracts from the SIMPLS calculation, represents as much variation information of X as possible. To explain the information of Y, the component should be associated with Y as much as possible. In order to analyze the explanation of variation of X to Y, the variable importance in projection (VIP) [34] is introduced to quantitatively denote the impact of i-th feature to Y.

Definition VIP: Let $r(\cdot,\cdot)$ be the correlation coefficient between two variables. The VIP is defined as:

$$VIP\left(x_j\right) \;=\; \sqrt{p\frac{\sum_{i=1}^{m} \psi(Y; t_i) {v_{ji}}^2}{\sum_{i=1}^{m} \psi(Y; t_i)}} \tag{7}$$

where $\psi(Y; t_i) = r^2(Y, P_i)$ is the explanation of variation of component P_i to Y, p is the number of features and v_{ji} is the weight of the j-th feature for the i-th component. The larger value of $VIP(x_j)$ is, the more explanatory power of x_j to Y.

The pseudo code of PLS-based feature selection is presented in Method 1.

```
Method 1. PLS-based Feature Selection (PLSFS)
Input: $X \in R^{n \times p}$, $Y \in R^{n \times 1}$, $m$
Output: feature weight $VIP$
Obtain the $\psi(Y; t_i)$ and $v_{ji}^2$ from the result of the function SIMPLS $(X, Y, m)$
For $k$ = 1 to p do
    Calculate each feature weight $VIP(k)$ in terms of Eq.(7)
End
Return $VIP$
```

Refining the inferred regulatory network

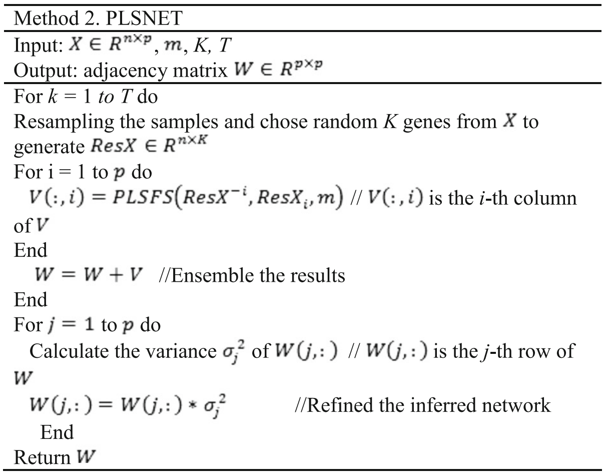
In our method, we use a statistical technique to refine the inferred regulatory network in our method. The final result is improved under the assumption that if a regulatory gene regulates many target genes (e.g., the regulatory gene is hub node), it is an important regulatory gene. Once the solution of the gene regulatory network inference is calculated, we can obtain an adjacency matrix W, where W_{ij} represents the strength that gene i

associates (i.e., regulates) gene *j*. Regulatory genes are scored based on their impacts on multiple target genes. An updated adjacency matrix *W* is given as:

$$W(i,:) \;=\; W(i,:) * \sigma_i^2, \forall i \in \{1, 2, ..., p\} \qquad (8)$$

where $W(i,:)$ is the *i*-th row of *W*, and σ_i^2 is a variance in the *i*-th row of *W*. It should be noted that each row of *W* is calculated in a subproblem of our method. Each row of *W* contains relative scores with respect to a different target gene. Therefore, if a regulatory gene regulates many target genes, the variance in a row of *W* corresponding to that regulatory gene is elevated.

The pseudo code of PLS-based ensemble method (PLSNET) is presented in Method 2.



```
Method 2. PLSNET
Input: $X \in R^{n \times p}$, $m$, K, T
Output: adjacency matrix $W \in R^{p \times p}$
For k = 1 to T do
    Resampling the samples and chose random K genes from $X$ to generate $ResX \in R^{n \times K}$
    For i = 1 to $p$ do
        $V(:,i) = PLSFS(ResX^{-i}, ResX_i, m)$ // $V(:,i)$ is the i-th column of $V$
    End
    $W = W + V$  //Ensemble the results
End
For $j = 1$ to $p$ do
    Calculate the variance $\sigma_j^2$ of $W(j,:)$ // $W(j,:)$ is the j-th row of $W$
    $W(j,:) = W(j,:) * \sigma_j^2$    //Refined the inferred network
End
Return $W$
```

Parameter settings

The main parameters of PLSNET are the number of components *m* and the number of candidate regulatory genes *K*. Parameter selection (i.e., the selection of the *m* variable) for the PLS model is a difficult task due to the fact that if *m* is too large, there will be over-fitting in the model and if *m* is too small, there will be under-fitting in the model. There are two widely used methods for PLS parameter tuning, specification and cross validation (CV). The drawback to CV is that it significantly increases the computation cost and the problem to a certain extent becomes even more difficult to handle. The specification method usually fixes the value of *m*, typically, the value is not larger than 5. Since we do not know how many candidate regulatory genes are sufficient to provide a good model for a target gene, the choice of *K* may not be trivial.

In this paper, we evaluated our method PLSNET on two popular benchmarks: DREAM4 multifactorial datasets and DREAM5 datasets. For DREAM4 multifactorial datasets, we use CV to set two main parameters of PLSNET, where *m* is chosen from $\{1, 2, \cdots, 5\}$ and *K* is chosen from $\{5, 10, \cdots, 100\}$. And we choose the parameter setting ($m = 5$, $K = 30$) as default values. As the size of DREAM5 datasets is much larger than that of DREAM4 multifactorial datasets, it is difficult to utilize CV to choose the parameters due to the fact that it would significantly increase the computation cost. Instead, we utilize the specification method to set $m = 5$. And following the suggestion of GENIE3 [18], we set $K = \sqrt{p}$ as default value for DREAM5 datasets.

Computational complexity

As shown in Method 2, there are two main parts in PLSNET, including calculating the score of each edge and refining the inferred network. Consider $N \times P$ *matrix X* and $N \times 1$ *matrix Y*, SIMPLS is $O(mNP)$ complex. Here, *m* is the number of components, *N* is number of samples and *P* is the number of genes. Another part of PLSFS (i.e., *VIP*) is also $O(mNP)$ complex. Hence, the computational complexity of PLSFS is $O(mNP)$ and we calculate the score of each edge in an $O(mTKNP)$ time, where *K* is the number of candidate regulatory genes and *T* is the number of iterations. PLSNET's complex is thus on the order of $O(mTKNP + P^2)$. In practice, the dominating part of the sum is *mTKNP* and the value of *m* is not larger than 5, we therefore report a final computational complexity of PLSNET as $O(TKNP)$. We compare our method with other inference methods in Table 1. It should be noted that the calculation of the mutual information matrix is not included for information-theoretic methods (i.e., CLR and ARACNE).

Results

In recent years, the problem of evaluating performance of the inference methods on adequate benchmarks has

Table 1 The computational complexity of different GRN inference methods

Method	Complexity
GENIE3	$O(TKPN \log N), T = 1000,\ K = \sqrt{P}.$
TIGRESS	$O(TKPN), T = 1000,$ *K = number of regulatory genes.*
CLR	$O(P^2)$
ARACNE	$O(P^3)$
NIMEFI	$O(TKPN \log N), T = 1000,\ K = \sqrt{P}$
PLSNET	$O(TKPN),\ T = 1000, K = \sqrt{P}.$

The computational complexity of PLSNET and other GRN inference methods with respect to the number of genes *P*, the number of iterations *T* and the number of samples *N* been widely investigated [24, 35]. The most popular benchmarks, such as *S. cerevisiae* [36], *E. coli* [37] and artificially simulated in silico networks [24, 38–40], are derived from well-studied in *vivo* networks of model organisms. One weakness of *in vivo* benchmark networks is that no matter how well the model organism is studied, experimentally confirmed pathways can never be assumed complete [21]. As such networks are assembled from known transcriptional interactions with strong experimental support, the gold standard networks are expected to have few false positives. Given a gene expression data matrix, a GRN inference method outputs a ranked list of putative regulatory interactions. Taking the top L predictions in this list, we can compare them to known regulations (i.e., the gold standard networks) to evaluate the performance of the GRN inference method.

In this paper, we used several popular benchmark GRNs to evaluate the accuracy of our proposed method and compare it with the other inference methods. The datasets we used in our experiments are from DREAM challenges and the details of the datasets are summarized in Table 2. The first three networks come from the DREAM5 challenge. Network 1 (*in-silico*) is a simulated network with simulated expression data, while the other two expression datasets are real expression data collected for *E. coli* (Network 3) and *S. cerevisiae* (Network 4). It should be noted that we do not use Network 2 of DREAM5 in our experiments for the reason that there is no verified interaction provided for this dataset. In order to assess the ability of our method to predict directionality, we used the five DREAM4 size 100 Multifactorial Networks in our experiments, where the regulatory genes are not known in advance for these networks.

Table 2 Datasets

Network	# Genes	# Regulatory genes	#Samples	# Verified interactions
DREAM5 Network 1 (in-silico)	1643	195	805	4012
DREAM5 Network 3 (E. coli)	4511	334	805	2066
DREAM5 Network 4 (S. cerevisiae)	5950	333	536	3940
DREAM4 Multifactorial Network 1	100	100	100	176
DREAM4 Multifactorial Network 2	100	100	100	249
DREAM4 Multifactorial Network 3	100	100	100	195
DREAM4 Multifactorial Network 4	100	100	100	211
DREAM4 Multifactorial Network 5	100	100	100	193

In fact, DREAM4 and DREAM5 datasets have been widely used for several GRNs inference methods to evaluate the performance recently. For example, the authors of TIGRESS [20] compared the performance of some GRNs inference methods on DREAM4 Multifactorial Networks and DREAM 5 Networks in 2012. In the same year, the authors of ANOVerence [12] presented the results of several GRNs inference methods performed on DREAM5 Networks. In 2014, the performance comparisons of many GRNs inference methods on DREAM4 Multifactorial Networks and DREAM 5 Networks were shown in NIMEFI [22].

We evaluated the accuracy of the methods using the Overall Score metric proposed by the authors of DREAM challenges [24], as shown in the following:

$$Overall\ Score \ = \ -\frac{1}{2}log_{10}(P_{AUPR}\cdot P_{AUROC}) \tag{9}$$

where *P_AUPR* and *P_AUROC* are respectively the geometric means of p-values taken over the networks from DREAM challenges, relating to an area under the precision-recall curve (AUPR) and an area under the receiver operating characteristic curve (AUROC). The probability densities of DREAM Network data which are used to calculate the p-values and the respective gold standard networks are provided on DREAM web site.

Performance evaluation

We compare the performance of our method PLSNET with five of the most prominent GRN inference methods, GENIE3 [18], TIGRESS [20], CLR [8], ARACNE [9] and NIMEFI [22], that are widely used in the literature. Moreover, the top three performers in each of DREAM challenges as listed on the DREAM web site are also selected for comparison. We use the Matlab implementations of GENIE3 and TIGRESS, while ARACNE and CLR are run in the *minet* R package [41]. NIMEFI is implemented using the R package available for download at http://bioinformatics.intec.ugent.be/. The Matlab code of PLSNET is included in Additional file 1. We keep default parameter values for each of these methods and set the number of iterations $T = 1000$ for ensemble methods (i.e., GENIE3, TIGRESS, NIMEFI and PLSNET).

Performance on the DREAM4 multifactorial datasets

The goal of the *In Silico Size 100 Multifactorial* challenge of DREAM4 was to infer five networks from Multifactorial perturbation data, where each of them contained 100 genes and 100 samples. Multifactorial perturbation data are defined as gene expression profiles resulting from slight perturbations of all genes

Table 3 Performance comparisons of different GRN inference methods on the DREAM4 networks, challenge size 100 Multifactorial

Method	Network 1		Network2		Network 3		Network 4		Network 5		Overall Score
	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	
GENIE3	**0.161**	0.750	0.154	0.734	0.234	0.776	0.211	0.800	0.200	0.795	38.033
TIGRESS	0.158	0.747	0.161	0.703	0.233	0.761	0.225	0.774	0.233	0.754	36.590
CLR	0.143	0.701	0.117	0.695	0.174	0.744	0.181	0.753	0.175	0.723	29.112
ARACNE	0.122	0.605	0.102	0.603	0.201	0.691	0159	0.713	0.167	0.661	23.478
NIMEFI	0.157	**0.758**	0.157	0.731	**0.248**	0.776	0.225	0.806	**0.241**	**0.801**	40.762
PLSNET	0.118	0.713	**0.290**	**0.828**	0.202	**0.794**	**0.228**	**0.819**	0.206	0.786	**46.046**
Winner of the Challenge											
GENIE3	0.154	0.745	0.155	0.733	0.231	0.775	0.208	0.791	0.197	0.798	37.428
2nd	0.108	0.739	0.147	0.694	0.185	0.748	0.161	0.736	0.111	0.745	28.165
3rd	0.140	0.658	0.098	0.626	0.215	0.717	0.201	0.693	0.194	0.719	27.053

The best results for each column are in bold. Numbers in the "Winner of competition" part of the table correspond to the best methods participating in the challenge as listed on the DREAM web site

simultaneously. The topology of these benchmark networks were derived from the transcriptional regulatory system of *S. cerevisiae* and *E. coli*.

Each DREAM4 Multifactorial Network data is a 100 × 100 matrix, where each column represents a gene and each row represents a different experimental condition (i.e., perturbation). The values in the matrix are the expression values of the genes on the respectively experimental conditions. In our experiments, all compared GRNs inference methods used these matrices as the input data and the results are shown in Table 3.

Table 3 lists the results of PLSNET with default parameter setting ($m = 5$, $K = 30$) compared with those of other GRN inference methods on the DREAM4 multifactorial datasets. Without further optimization of the parameters on these networks, PLSNET achieves the best Overall Score. And PLSNET shows particularly strong performance on Network2 and Network4, improving over other GRN inference methods in terms of AUPR and AUROC.

Influence of parameters

In this section, we provide more details about the influence of the parameters of compared methods on

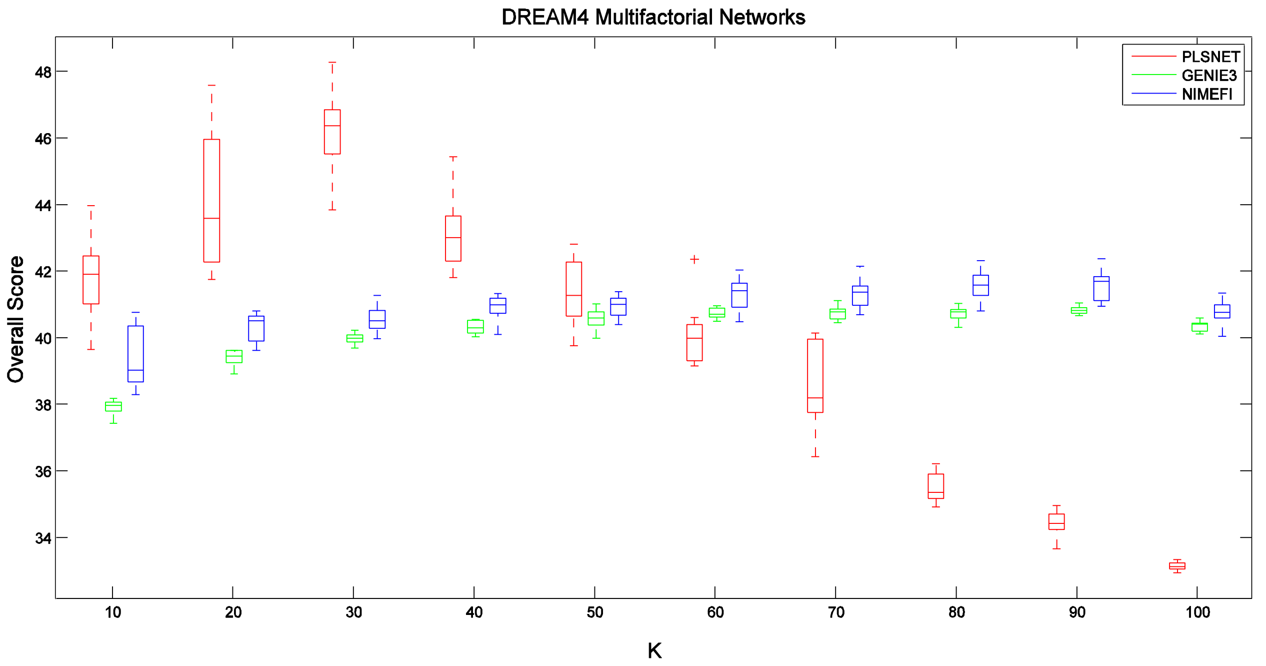


Fig. 1 Boxplots of Overall Score on DREAM4 Multifactorial Networks with respect to the number of candidate regulatory genes

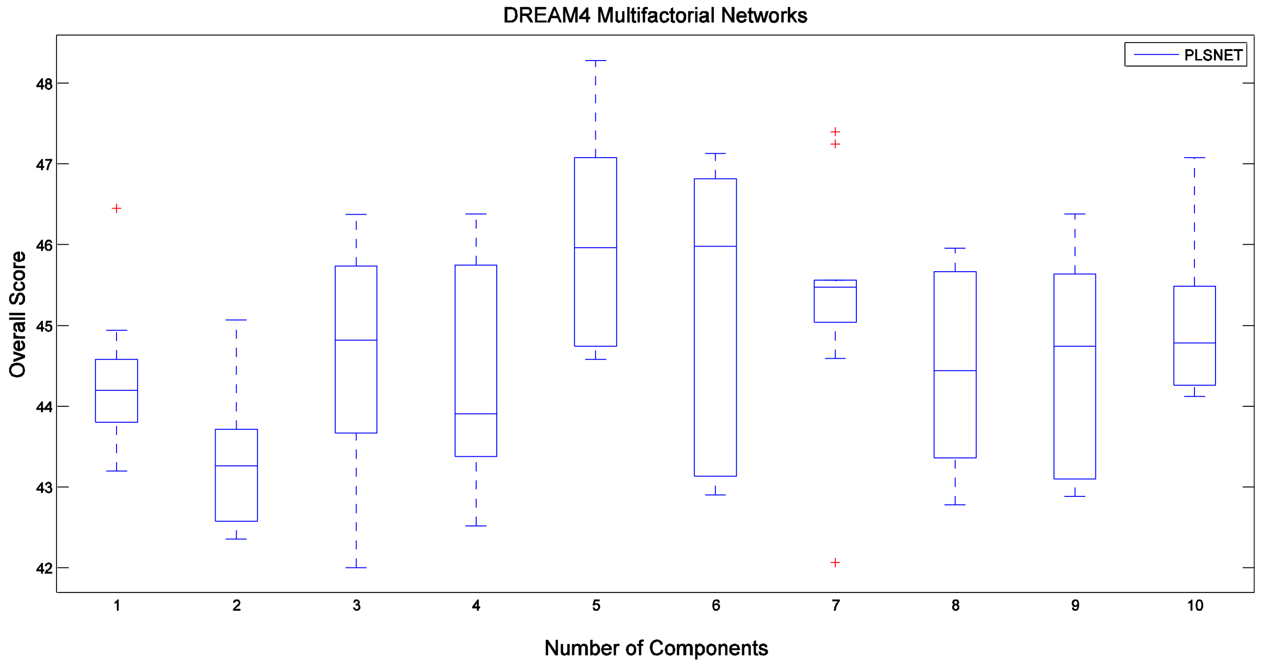


Fig. 2 Boxplots of Overall Score on DREAM4 Multifactorial Networks with respect to the parameter of PLSNET

performance, taking five DREAM4 Multifactorial Networks as benchmark datasets.

Figure 1 summarizes the Overall Score of three compared methods (PLSNET, GENIE3 and NIMEFI) for different number of candidate regulatory genes K on the DREAM4 multifactorial datasets. As seen in Fig. 1, the range of K values leading to the best performance is narrow with our proposed method, and therefore it is difficult to find an appropriate value of K as default value in advance.

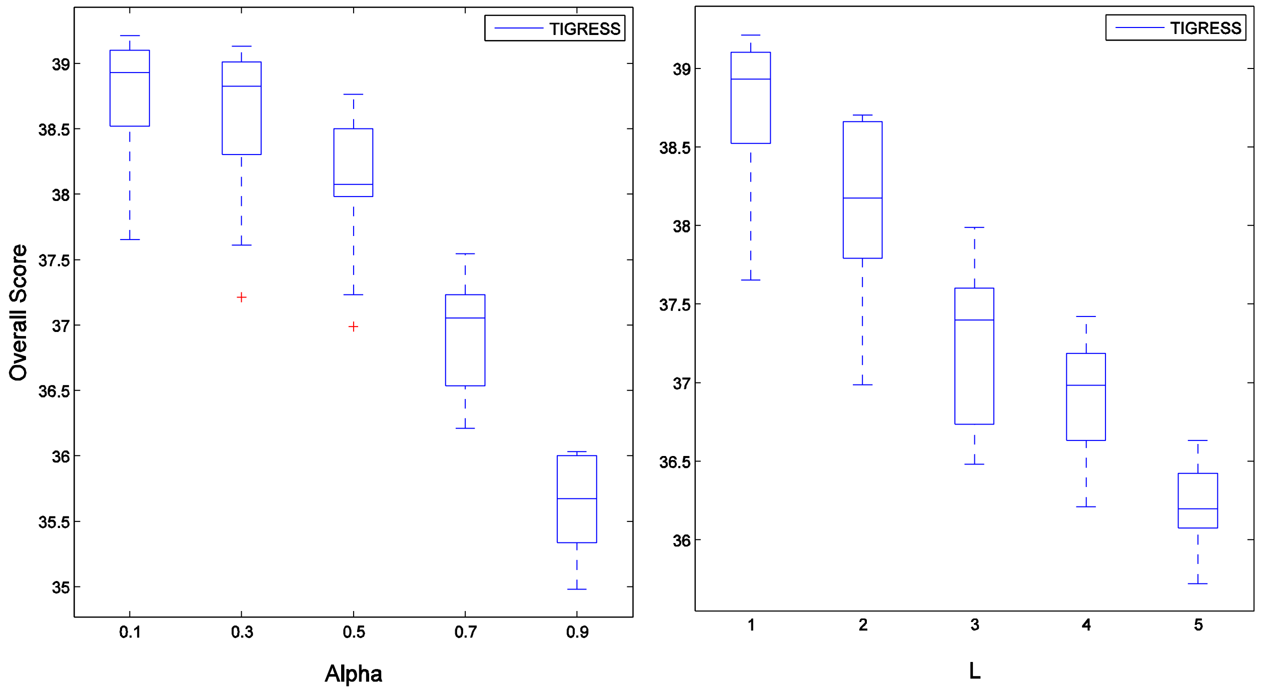


Fig. 3 Boxplots of Overall Score on DREAM4 Multifactorial Networks with respect to the parameters of TIGRESS

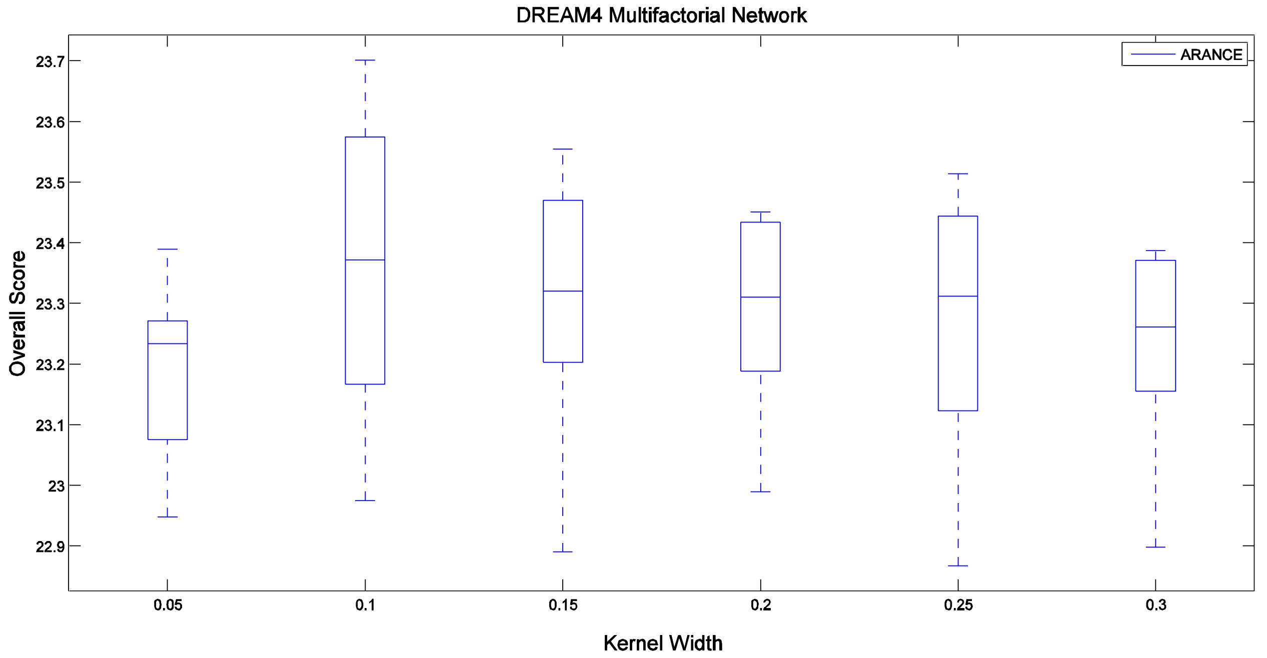


Fig. 4 Boxplots of Overall Score on DREAM4 Multifactorial Networks with respect to the parameter of ARACNE

Figure 2 shows the Overall Score of our method for different number of components m with $K = 30$ on the DREAM4 multifactorial datasets. We observe in Fig. 2 that the Overall Score is not very sensitive to the choice of the number of components, and therefore one may practically more easily tune it for optimal performance.

Figure 3 shows the influence of two main parameters (α and L) of TIGRESS on the Overall Score using DREAM4 Multifactorial datasets, where $\alpha \in [0, 1]$ controls the random re-weighting in each stability selection run and L is the number of LARS (Least Angle Regression) steps. The Overall Score of ARACNE for different kernel widths on DREAM4 Multifactorial Networks is shown in Fig. 4.

Performance on the DREAM5 datasets

The three DREAM5 datasets were structured with respect to different model organisms, and were different in size. The expression data of the only one network (Network1) were simulated *in silico*, while two other sets of expression data were measured in real experiments *in vivo*. As in all DREAM challenges, *in silico* expression data were simulated using an open-source GeneNetWeaver simulator [25]. The gold standard networks of DREAM5 were mainly obtained from two sources: Gene Ontology (GO) annotations [42] and RegulonDB database [36].

Each DREAM5 Network data contain three files: network chip features, network transcription factors and network expression data. The file of network chip features records the details of each experimental condition in network expression data, which contain time series, perturbations and even replicates. However, as mentioned in Section 2.1, we do not use the information for inferring GRNs. And the methods compared in our experiments do not use the information as well. The file of network transcription factors records the genes that have been verified to be regulatory genes. Typically, the number of regulatory genes is used as a parameter for GRNs inference methods to construct the

Table 4 Performance comparisons of different GRN inference methods on the DREAM5 networks

Method	Network 1		Network 3		Network 4		Overall Score
	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	
GENIE3	0.291	0.814	0.094	0.618	0.021	0.517	40.313
TIGRESS	**0.302**	0.783	0.070	0.596	0.020	0.517	31.112
CLR	0.254	0.771	0.075	0.591	0.020	0.516	19.387
ARACNE	0.187	0.763	0.069	0.572	0.018	0.504	9.24
NIMEFI	0.298	0.817	0.101	0.625	0.022	0.518	46.015
PLSNET	0.270	**0.862**	0.065	0.577	**0.023**	**0.519**	**48.269**
Winner of the Challenge							
GENIE3	0.291	0.815	0.093	0.617	0.021	0.518	40.279
ANOVerence	0.245	0.780	**0.119**	**0.671**	0.022	**0.519**	34.023
TIGRESS	0.301	0.782	0.069	0.595	0.020	0.517	31.099

The best results for each column are in bold. Numbers in the “Winner of competition” part of the table correspond to the best methods participating in the challenge as listed on the DREAM web site model. The file of network expression data contains a $n \times p$ matrix, where n represents the number of experimental conditions and p is the number of genes, and the values in the matrix are the expression values of the genes on the respectively experimental conditions. In our experiments, all compared methods used these matrices as the input data and the results are given in Table 4.

Table 4 summarizes the results of PLSNET with default parameter setting ($m = 5,\ K = \sqrt{p}$) compared with those of other GRN inference methods on the DREAM5 datasets. As seen in Table 4, PLSNET achieves the best Overall Score, as well as the best individual AUROC scores for Network 1 and Network 4. ANOVerence achieved the best performance on the *E. coli* network (Network 2), as it does include meta-information of the microarray chips to guide the network inference process.

Since the number of regulatory genes on DREAM5 datasets is much larger than that of on DREAM4 datasets, it is more difficult to set the number of candidate regulatory genes K. In our experiments, we set $K = \sqrt{p}$ and observed that our method perform well in this setting. However, it should be noted that better results could be obtained if K is set to other values.

Obviously, all GRN inference methods achieved better scores for an *in silico* network (Network 1) than for other two *in vivo* networks. One main reason for a poor performance of the inference methods for *in vivo* networks may be that experimentally confirmed pathways, and the gold standards derived from them, cannot be assumed completely. On the other hand, *in silico* datasets provide enough information to confidently reverse-engineer their underlying structure.

CPU time

In our experiments, ARACNE, CLR and NIMEFI were implemented using the R package, while GENIE3, TIGRESS and our method PLSNET were run in Matlab. As PLSNET is an ensemble method, we focus on the running time of ensemble methods rather than other GRN inference methods. On the other hand, ensemble methods usually achieve better results than other GRN inference methods.

Table 5 Comparisons of running times of different GRN inference methods

Method	CPU time (in seconds)			
	DREAM4 (the average of 5 networks)	DREAM5 Network 1	DREAM4 Network 3	DREAM4 Network 4
GENIE3	47.73	3.51E + 4	1.36E + 5	1.17E + 5
TIGRESS	160.41	3.06E + 4	9.08E + 4	7.02E + 4
PLSNET	136.71	4.22E + 3	1.66E + 4	2.09E + 4

Table 5 gives an overview of the running times of some of the GRN inference methods. These measurements were conducted using Matlab (R2010a edition), an Intel Core (TM) i5-3317U, clocked at 1.70 GHz, 4.00 GB of RAM memory and a 64-bit Microsoft Windows 7 operating system. Note that we do not include NIMEFI for comparison due to the fact that NIMEFI is a method using multiple ensembles of GRN inference methods, including GENIE3, Ensemble Elastic Net and Ensemble Support Vector Regression.

As can be seen from the table, in terms of computational efficiency, PLSNET performs best on DREAM5 networks and performs the second best on DREAM4 networks. GENIE3 performs best on DREAM4 networks as the size of the datasets is small. However, GENIE3 is more time consuming than PLSNET when it is implemented on the big datasets.

Conclusions

In this paper, we presented PLSNET, a new ensemble method for GRN inference. PLSNET expresses the GRN inference problem as a feature selection problem, and solves it with the PLS-based feature selection method combined with a statistical technique for refining the predictions. The influence of PLSNET parameters was clarified in this paper, and we showed that further improvement may result from finer parameter tuning.

Different from other ensemble methods, such as GENIE3 and TIGRESS, PLSNET aggregates the features selected by PLS-based method. Moreover, considering that if a regulatory gene regulates many target genes (e.g., a regulatory gene is a hub node), it indicates an important regulator gene; we use a statistical technique to refine the inferred network in our method.

We evaluated our proposed method on the DREAM4 multifactorial and DREAM5 benchmarks and achieved higher accuracy than other state-of-the-art methods. Furthermore, among ensemble GRN inference methods, our method is computationally efficient.

Additional file

Additional file 1: Code of PLSNET. The Matlab code of our proposed method (ZIP 4 kb)

Abbreviations

ARACNE: Algorithm for the Reconstruction of Accurate Cellular NEtworks; AUPR: An area under the precision-recall curve; AUROC: An area under the receiver operating characteristic curve; CLR: Context likelihood of relatedness; DREAM: Dialogue for reverse engineering assessments and methods; GENIE3: Gene network inference with ensemble of trees; GRN: Gene regulatory network; NIMEFI: Network inference using multiple ensemble feature importance algorithms; PLS: Partial least squares; PLSNET: PLS-based gene NETwork inference method; TIGRESS: Trustful Inference of Gene REgulation using Stability Selection

Acknowledgments
We would like to acknowledgment three reviewers for helpful suggestions.

Funding
This research was supported by Research Fund for National Natural Science Foundation of China (General Program) under Grant No. 61274133 and Shenzhen Technology Development Foundation Grant No. CXZZ20150813155917544.

Availability of data and materials
The datasets of DREAM4 are available on http://www.synapse.org/#!Synapse:syn3049712/files/.
The datasets of DREAM5 are available on http://www.synapse.org/#!Synapse:syn2787209/files/.

Authors' contributions
SG designed the method, conducted the experiments, and wrote the manuscript. QJ supervised the project. LC wrote the manuscript. DG gave the ideas and supervised the project. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable.

Ethics approval and consent to participate
Not applicable.

Author details
[1]Department of Electronic Engineering, Xiamen University, Fujian 361005, China. [2]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518000, China. [3]School of Mathematics and Computer Science, Fujian Normal University, Fujian 350117, China.

Received: 12 April 2016 Accepted: 1 December 2016
Published online: 28 December 2016

References
1. Bolouri H. Computational modeling of gene regulatory networks: a primer. London: Imperial College Press; 2008.
2. Gardner TS, Faith JJ. Reverse-engineering transcription control networks. Phys Life Rev. 2005;2(1):65–88.
3. Bansal M, Belcastro V, Ambesi-Impiombato A, et al. How to infer gene networks from expression profiles. Mol Syst Biol. 2007;3(1):78.
4. Markowetz F, Spang R. Inferring cellular networks–a review. BMC Bioinf. 2007;8 Suppl 6:S5.
5. Lee WP, Tzou WS. Computational methods for discovering gene networks from expression data. Brief Bioinform. 2009;10(4):408–23.
6. Eisen MB, Spellman PT, Brown PO, et al. Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci. 1998;95(25):14863–8.
7. Butte AJ, Kohane IS. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. Pac Symp Biocomput. 2000;5:418–29.
8. Faith JJ, Hayete B, Thaden JT, et al. Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS Biol. 2007;5(1):e8.
9. Margolin AA, Wang K, Lim WK, et al. Reverse engineering cellular networks. Nat Protoc. 2006;1(2):662–71.
10. Altay G, Emmert-Streib F. Inferring the conservative causal core of gene regulatory networks. BMC Syst Biol. 2010;4(1):132.
11. de Matos SR, Emmert-Streib F. Bagging statistical network inference from large-scale gene expression data. PLoS One. 2012;7(3):e33624.
12. Küffner R, Petri T, Tavakkolkhah P, et al. Inferring gene regulatory networks by ANOVA. Bioinformatics. 2012;28(10):1376–82.
13. Friedman N. Inferring cellular networks using probabilistic graphical models. Science. 2004;303(5659):799–805.
14. Friedman N, Linial M, Nachman I, et al. Using Bayesian networks to analyze expression data. J Comput Biol. 2000;7(3-4):601–20.
15. Auliac C, Frouin V, Gidrol X, et al. Evolutionary approaches for the reverse-engineering of gene regulatory networks: A study on a biologically realistic dataset. BMC Bioinf. 2008;9(1):91.
16. Yu J, Smith VA, Wang P, et al. Advances to Bayesian network inference for generating causal networks from observational biological data. Bioinformatics. 2004;20(18):3594–603.
17. Perrin BE, Ralaivola L, Mazurie A, et al. Gene networks inference using dynamic Bayesian networks. Bioinformatics. 2003;19 suppl 2:ii138–48.
18. Irrthum A, Wehenkel L, Geurts P. Inferring regulatory networks from expression data using tree-based methods. PLoS One. 2010;5(9):e12776.
19. Marbach D, Prill RJ, Schaffter T, et al. Revealing strengths and weaknesses of methods for gene network inference. Proc Natl Acad Sci. 2010;107(14):6286–91.
20. Haury AC, Mordelet F, Vera-Licona P, et al. TIGRESS: trustful inference of gene regulation using stability selection. BMC Syst Biol. 2012;6(1):145.
21. Sławek J, Arodź T. ENNET: inferring large gene regulatory networks from expression data using gradient boosting. BMC Syst Biol. 2013;7(1):106.
22. Ruyssinck J, Geurts P, Dhaene T, et al. Nimefi: gene regulatory network inference using multiple ensemble feature importance algorithms. PLoS One. 2014;9(3):e92709.
23. Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc Series B Stat Methodology. 2005;67(2):301–20.
24. Marbach D, Costello JC, Küffner R, et al. Wisdom of crowds for robust gene network inference. Nat Methods. 2012;9(8):796–804.
25. Marbach D, Schaffter T, Mattiussi C, et al. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. J Comput Biol. 2009;16(2):229–39.
26. The DREAM4 In Silico network challenge. http://www.synapse.org/#!Synapse:syn3049712/files/.
27. The DREAM5 network challenge. http://www.synapse.org/#!Synapse:syn2787209/files/.
28. Meinshausen N, Bühlmann P. High-dimensional graphs and variable selection with the lasso, The annals of statistics. 2006. p. 1436–62.
29. You W, Yang Z, Yuan M, et al. Totalpls: Local dimension reduction for multicategory microarray data. Human-Machine Systems, IEEE Transactions on. 2014;44(1):125–38.
30. Sun S, Peng Q, Shakoor A. A kernel-based multivariate feature selection method for microarray data classification [J]. PLoS One. 2014;9(7):e102541.
31. Barker M, Rayens W. Partial least squares for discrimination. J Chemometr. 2003;17(3):166–73.
32. Wold H, Lyttkens E. Nonlinear iterative partial least squares (NIPALS) estimation procedures. Bull Int Stat Inst. 1969;43(1).
33. De Jong S. SIMPLS: an alternative approach to partial least squares regression. Chemom Intel Lab Syst. 1993;18(3):251–63.
34. Wold S, Johansson E, Cocchi M. PLS—partial least squares projections to latent structures. 3D QSAR in drug design. 1993;1:523–50.
35. Schaffter T, Marbach D, Floreano D. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics. 2011;27(16):2263–70.
36. Gama-Castro S, Salgado H, Peralta-Gil M, et al. RegulonDB version 7.0: transcriptional regulation of Escherichia coli K-12 integrated within genetic sensory response units (Gensor Units). Nucleic Acids Res. 2011;39 suppl 1:D98–D105.
37. Kim SY, Imoto S, Miyano S. Inferring gene networks from time series microarray data using dynamic Bayesian networks. Brief Bioinform. 2003;4(3):228–35.
38. Di Camillo B, Toffolo G, Cobelli C. A gene network simulator to assess reverse engineering algorithms. Ann N Y Acad Sci. 2009;1158(1):125–42.
39. Van den Bulcke T, Van Leemput K, Naudts B, et al. SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. BMC Bioinf. 2006;7(1):43.
40. Mendes P, Sha W, Ye K. Artificial gene networks for objective comparison of analysis algorithms. Bioinformatics. 2003;19 suppl 2:ii122–9.
41. Meyer PE, Lafitte F, Bontempi G. minet: AR/Bioconductor package for inferring large transcriptional networks using mutual information. BMC Bioinf. 2008;9(1):461.
42. Ashburner M, Ball CA, Blake JA, et al. Gene Ontology: tool for the unification of biology. Nat Genet. 2000;25(1):25–9.