BIOINFORMATICS APPLICATIONS NOTE
Vol. 22 no. 23 2006, pages 2952–2954
doi:10.1093/bioinformatics/btl491

Gene expression

BNArray: an R package for constructing gene regulatory networks from microarray data by using Bayesian network

Xiaohui Chen[1], Ming Chen[1,2,*] and Kaida Ning[1]

[1]Department of Bioinformatics and [2]The National Key Laboratory of Plant Physiology and Biochemistry, College of Life Sciences, Zhejiang University, Hangzhou 310058, China

Received on June 14, 2006; revised on August 28, 2006; accepted on September 18, 2006
Advance Access publication September 27, 2006
Associate Editor: John Quackenbush

ABSTRACT

Summary: *BNArray* is a systemized tool developed in R. It facilitates the construction of gene regulatory networks from DNA microarray data by using Bayesian network. Significant sub-modules of regulatory networks with high confidence are reconstructed by using our extended sub-network mining algorithm of directed graphs. *BNArray* can handle microarray datasets with missing data. To evaluate the statistical features of generated Bayesian networks, re-sampling procedures are utilized to yield collections of candidate 1st-order network sets for mining dense coherent sub-networks.

Availability: The R package and the supplementary documentation are available at http://www.cls.zju.edu.cn/binfo/BNArray/.

Contact: mchen@zju.edu.cn

1 INTRODUCTION

The genome project has vastly increased our knowledge of the genomic sequences and their encoding products of human and many other model organisms. DNA microarray is one of the most powerful techniques developed to survey the transcriptional profile of the entire genome. It can measure the state of living cell in one experiment (Lander 1999). Detailed laboratory protocols of microarray experiments have been developed in the past few years. Still the computational issues of the generated data remain arguable when dealing with specific situations.

Recently, Bayesian network, a probabilistic graphical model representation (Heckerman, 1999), has been widely used to analyze expression data (Friedman *et al*., 1999). Compared with clustering analysis, Bayesian network has the advantage of uncovering conditional independency among genes, which provides a promising way to survey direct interaction of gene regulation. Moreover, by using statistical evaluation approaches, we can examine features of induced high score networks, e.g. the confidence of the existence of an edge (Friedman *et al*., 1999). Thus, highly confident features provide us a potential way to mine significant sub-networks from candidate Bayesian networks. CODENSE, an efficient and fast algorithm for identifying coherent network modules from across multiple network collections, is used to reconstruct 2nd-order graphs from each undirected 1st-order network set (Hu *et al*., 2005).

So far, there is no tool to implement a systemized process for sub-regulatory networks reconstruction from large scale DNA microarray data using Bayesian framework. We have developed an R package, named *BNArray*, that provides a flexible interface for conducting this analysis. Moreover, we extend the CODENSE algorithm to xCODENSE that mines coherent modules from collections of directed graphs.

2 MODULES

BNArray has four main function modules.

1. Imputing missing data in microarray experiments with Least Local Squares (LLSimpute) algorithm (Kim *et al*., 2005), such that we can input complete database for constructing Bayesian networks.
2. Constructing Bayesian networks for gene regulation. We utilize previously implemented R package *deal* for learning Bayesian networks with mixed variables (Bøttcher *et al*., 2003).
3. Re-sampling microarray dataset to produce more reliable data using Efron's Bootstrap, and then repeating procedure 2 to construct a collection of 1st-order Bayesian networks with high scores.
4. Reconstructing significant coherent regulatory sub-networks with our extended xCODENSE algorithm for directed graph from previously induced candidate Bayesian networks.

BNArray allows users to specify their own parameters and modify the open source code to meet their individual needs.

3 IMPLEMENTATION

BNArray package is implemented in R, an open source programming environment (RC Team, 2006).

3.1 Impute missing values

The current version of *BNArray*(1.0) allows using (LLSimpute) algorithm to estimate the missing values in target genes as the linear combination of their most k-similar neighbors chosen by the first k smallest Euclidean distance. For example, assuming that the target gene g_1 contains a missing value in the first position of its total $n = 5$ experiment measures, we choose k similar genes, which consist of complete measurements before imputing

[*]To whom correspondence should be addressed.

the missing value in target gene, then we construct matrix A, vectors b and w, and the missing value as follows:

$$\begin{pmatrix} \alpha & w^T \\ b & A \end{pmatrix} = \begin{pmatrix} \alpha & w_1 & w_2 & w_3 & w_4 \\ b_1 & A_{1,1} & A_{1,2} & A_{1,3} & A_{1,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_k & A_{k,1} & A_{k,2} & A_{k,3} & A_{k,4} \end{pmatrix},$$

where α is the missing value in g_1, $w^T \in R^{1 \times (n-1)}$ contains $n-1$ elements of g_1 whose first missing item is deleted, the elements of $b \in R^{k \times 1}$ are the first components of the k-nearest genes, and the rows of the matrix A contain k-nearest neighbor genes with their first values deleted. With the above definition, the least squares problem based on L_2-norm can be formulated as

$$\min_x \| A^T x - w \|_2$$

Then, the missing value α is estimated as linear combination of the vector b

$$\alpha = b^T x = b^T (A^T)^{\dagger} w,$$

where $(A^T)^{\dagger}$ is the pseudoinverse of A^T. This procedure is implemented in the LLSimpute function.

Note that if a gene misses too many values across the total experiments, LLSimpute cannot estimate a coefficient for these missing data. FinalImpute prodedure imputes too bad data to a specified value, which means the expressionlevels cannot be detected in these experiments.

3.2 Construct Bayesian networks

Differentially expressed genes to be modeled within the domain of Bayesian networks are selected. For example, we select a subset of $m = 5$ genes with $n = 5$ experiments as the following matrix:

YAL022C	YAL040C	YAL053W	YAL067C	YAR003W
−0.22	3.43	0.34	−0.15	0.41
0.86	2.75	0.90	−1.25	0.25
−0.25	0.36	0.58	1.15	0.62
0.89	0.72	−0.92	0.07	0.29
−0.36	1.04	0.21	0.01	0.30

We use the Gaussian-inverse Gamma distribution for continuous variables, i.e. genes in our context, as conjugate local priors given the configuration of the discrete parents (In the current version, we recognize the gene expression measurements are all continuous from a multivariate normal distribution). Under this framework and the assumption of parameter independence, an initial Bayesian network structure is learned from the training data and a user-specified prior network.

From this initial network, greedy search algorithm with random restarts is performed to get the highest score posterior network to avoid local maxima. Finally, we obtain an optimized Bayesian network that maximizes the Bayes factor, i.e. the highest scoring network using heuristic search of the network space in a specified domain. The obtained network structure encodes the conditional independence relationship among the genes in the domain. In fact, there will exist an edge between two directly interacted genes. If they are in the same v-structures of the Bayesian networks with the same skeletons, then one gene is the regulator of the other. Otherwise, due to the structural equivalence, they function as a binding complex. Note that because we only consider the regulation on RNA level, post-transcriptional and protein level regulation are not included in current discussion.

3.3 Bootstrap Bayesian networks

To fully use our limited experiment data as far as possible, we then generate several best reasonable networks from microarray data perturbed by using Efron's non-parametric bootstrap approach with replacement. This provides a computationally effective approach to estimate the confidence levels on features of generated networks: is the existence of an edge between two genes warranted? i.e is the regulatory or binding relationship between two genes highly confident? By selecting edges whose confidence levels exceed the pre-defined threshold, we obtain a set of highly confident edges whose encoding relationships are believable. Furthermore, the bootstrap-produced 1st-order regulatory networks are used to construct 2nd-order graphs, which are a representation of the meta-information of the induced collections of 1st-order Bayesian networks. This function is implemented in the BootstrapBN procedure.

3.4 Reconstruct significant coherent regulatory modules

We extend CODENSE algorithm to directed graph. xCODENSE internally calls function HCS for mining highly connected subgraphs (Hartuv *et al*., 2000), a clustering algorithm based on graph connectivity.

We can derive frequent subgraphs from previously selected highly-confident-edges, which provide potential networks for real regulatory relationship among genes. The biological networks, however, often function as 'network modules', which means the edges in the module are present or absent simultaneously. Therefore, from the candidate 1st-order Bayesian networks, we aim at mining the dense coherent sub-networks. For this purpose, 2nd-order graph is retrieved, as a node in 2nd graph represents an edge in 1st network and an edge in 2nd graph denotes a co-occurrence of two connected edges in 1st network above a user-defined threshold. For each coherent sub-network, we believe the feature and the co-occurrence of all edges in these modules are highly confident by defining a connectivity threshold. This process is implemented in the xCODENSE function. For the comparison of parameters' choice, we develop a more detailed discussion in an example in the Supplementary materials.

The downloadable package and its detailed information of usage are available at http://www.cls.zju.edu.cn/binfo/BNArray/.

4 CONCLUSION

BNArray facilitates the analysis of large amounts of microarray data, which differs from classic clustering methods. It provides an implementation of reconstructing regulatory sub-networks in R programming environment. *BNArray* can systematically model DNA microarray data with missing values with Bayesian framework. Further, it employs statistics evaluation of candidate high scoring Bayesian networks and collects them as a network set.

Finally, directed dense coherent significant sub-networks are reconstructed from the network set.

ACKNOWLEDGEMENTS

The authors thank the reviewers for suggestions on how to improve the manuscript. This work was supported in part by the NSFC (No. 30500106), the 973 Program (No. 2005CB120900), and by the DEZP (No. 20051038).

Conflict of Interest: none declared.

REFERENCES

Bøttcher,S.G. and Dethlefsen,C. (2003) Deal: a package for learning Bayesian Networks. *J. Stat. Software*, **8**, 1–40.

Friedman,N. *et al*. (1999) Data analysis with Bayesian networks: a bootstrap approach. In *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, 206–215.

Friedman,N. *et al*. (2000) Using Bayesian networks to analyze expression data. *J. Comput. Biol*., **7**, 601–620.

Hartuv,E. and Shamir,R. (2000) Clustering algorithm based on graph connectivity. *Information processing letters archive*, **76**, 175–181.

Heckerman,D. (1999) A tutorial on learning with Bayesian networks. In Jordan,M. (ed.), *Learning in Graphical Models*. MIT Press, Cambridge, MA.

Hu,H. *et al*. (2005) Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, **21** (suppl), i213–i221.

Kim,H. *et al*. (2005) Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics*, **21**, 187–198.

Lander,E.S. (1999) Array of hope. *Nature*, **21**, 3–4.

R Development Core Team (2006) R: A language and environment for statistical computing. *R Foundation Statistical Computing,* Vienna, Austria.