Gene Network Modeling through Semi-Fixed Bayesian Network

Tie-Fei Liu [1] **Wing-Kin Sung**[2] **Ankush Mittal**[3]

Abstract. Gene networks describe functional pathways in a given cell or tissue, representing processes such as metabolism, gene expression regulation, protein or RNA transport. Thus, learning gene network is a crucial problem in the post genome era. Most existing works learn gene networks by assuming one gene provokes the expression of another gene directly leading to an over-simplified model. In this paper, we show that the gene regulation is a complex problem with many hidden variables. We propose a semi-fixed model to represent the gene network as a Bayesian network with hidden variables. In addition, an effective algorithm to learn the model is presented. Experiments on artificial and real-life dataset confirm the effectiveness of our approach.

1 introduction

The physiological functions of organisms are accomplished through the coordinated regulation of the expression of a large number of genes. The large number of regulation pathways comprise complex networks which is called gene network: gene ensembles functioning in a coordinated manner to provide vital functions, the fine regulation of physiological processes, and the responses to external stimuli[18]. The accurate reconstruction of gene networks, indicated by learning gene network, has many possible benefits and currently is the focus of much active research[7]. Yet, it is impractical to construct a detailed biochemical model of an organism containing hundreds or even tens of thousands of genes by analyzing each gene and determining all the binding and reaction constants one by one manually[4]. Therefore the methods for automatically reconstructing gene network by computing are needed.

Gene network can be reconstructed by analyzing the gene expression data, which are extracted from the microarray[1, 4, 5, 7, 11, 12, 16, 17, 19–21, 23–26]. Most works treat gene expression data as complete dataset and exclusively use this to learn the gene network. However, the regulation of genes consists of transcription, translation, splicing, posttranslational protein degradation, active membrane transport, and other processes[18]. Each regulation component should be taken into consideration in a correct description of the gene network. Currently, beside gene expression level(referring to the mRNA transcription), other types of data are still absence in the scale of genome or proteome. Treating all components as hidden variables is impractical since learning the structure with so many hidden variable is difficult.

Based on the biology knowledge, it is observed that the most important components in the regulation system is the interaction between $DNA \leftrightarrow Protein$. Therefore, the regulation system can be simplified as the interaction between genes and proteins, like $DNA \rightarrow Protein \rightarrow DNA$ while other intermediate step between $DNA \leftrightarrow Protein$ are to change the interaction strengths. That is, giving the gene expression levels alone, gene regulation is achieved by proteins, which is regarded as the agent of the regulation system (we shall describe the process in detail in section 2). Proteins play important roles in gene network. Therefore, when analyzing microarray gene expression data, proteins should be included as hidden variables although their expression levels are still difficult to measure in large scale. The following are the advantage of an approach modeling protein as hidden variables:

- The model will become more meaningful, more interpretable and more closer to real-life system[2, 8–10].
- In the model, proteins are decision-relevant. The network without considering hidden variable may omit some dependencies [2, 8]. Compared to protein, other regulation components are less important. A general simplification used by researchers is to integrate effect of protein into other parameters and adjust the regulation strength without treating the proteins as hidden variables.

Hidden variables introduce advantages as well as learning complexity in the network. Moreover, microarray gene expression datasets often have missing values. Thus, Bayesian network forms a natural choice with the advantage of allowing for principled methods for learning the causal relationships with incomplete data, both hidden variables and missing values[10]. Successful application of Bayesian network can be found in domain of learning with hidden variables for several applications [2, 8–10]. The most widely used method for structure learning is EM algorithm[10]. In E step, the algorithm calculates the score of each possible structure using the structure and parameters learnt from previous iteration. Selection of the structure and parameters which maximizes the score is done in M step. The procedure is repeated until convergence criteria are met. In our problem, such kind of learning is difficult since the algorithm needs to learn the relationship among the hidden variables and the observed variables. In addition it is difficult to predetermine the correct number of hidden variables. To determine optimum number of hidden variables introduces more complexity in learning algorithm.

[1] Department of Computer Science,National University of Singapore & Institute of Bioengineering and Nanotechnology, Singapore. liutiefe@comp.nus.edu.sg

[2] Contact Author. Department of Computer Science,National University of Singapore, Singapore, 117543. ksung@comp.nus.edu.sg, Phone:(65)68746772, Fax:(65)67797465. This work is supported in part by the NUS Academic Research Grant R-252-000-120-112.

[3] Department of Electronics and Computer Engg., Indian Institute of Technology, Roorkee, India. ankumfec@iitr.ernet.in

[4] http://www.cs.unm.edu/∼patrik/networks/networks.html

We propose a system which models the gene network as a directed graph with hidden variables. In the model, the number of hidden variables is predefined by the biology knowledge and the relationships between hidden variables and observed variables are partially fixed. Also, we proposed an EM algorithm which can learn such networks efficiently.

2 Model Gene Network as a Semi-Fixed Network with Hidden Variables

In the gene regulation system, the regulation process can be described as: $g_i \rightarrow r_i \rightarrow p_i \rightarrow g_j$, where g_i and g_j are genes in which g_i regulates g_j, r_i is the mRNA(messenger RNA) generated by g_i and p_i is the protein translated by r_i. Here, $g_i \rightarrow r_i \rightarrow p_i$ is the central dogma which describes the gene expression, where a mRNA is a messenger to transform the genetic information from a gene to a protein. Thus, there are two main steps in the gene regulation procedure: the first step is gene expression and the second step is the regulation. The expression is to generate the protein to regulate the target gene. Therefore, we can simplify the gene regulation procedure as follows: $g_i \rightarrow p_i \rightarrow g_j$ where genes and proteins are the most important components in the regulation system and other components such as splicing and protein degradation are the intermediate components to adjust the expression and regulation strengths. Let $Pa(g_j) = \{g_1, ..., g_k\}$ denote a parent set of gene g_j such that each $g_i \in Pa(g_j)$ expresses gene g_j. An example network is shown in Figure 1(a). If a gene g_j is regulated a set of genes $Pa(g_j)$, all proteins $\{p_1, ..., p_k\}$ generated by $Pa(g_i)$ act in a combinative manner and it is not necessary to model the influence individually. Considering a single node representing the combined proteins influence as the directed regulator to regulate the target gene, the model can be further simplified as shown in Figure 1(b).

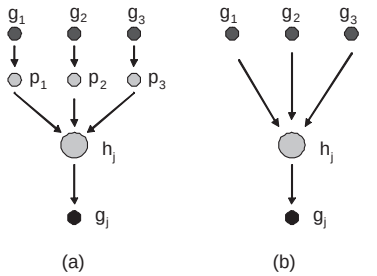

Figure 1. Simplified gene regulation system. (a) Gene expression system can be simplified as the interaction of genes and proteins. (b) The system can be further simplified since the combined proteins are the direct regulator to the target genes.

Based on the above discussion, for each gene g_j, we propose a hidden variable h_j, which is the combination of a set of proteins expressed from $Pa(g_j)$. Thereupon, $Pa(g_j)$ regulate h_j, then h_j regulates g_j. Based on the simplified regulation system, each gene is regulated by at most one hidden variable and the hidden variables are regulated by one or more than one genes. Such model means the interaction exists only between $Gene \leftrightarrow Protein$ and there is no edges from gene to gene or hidden variable to hidden variable. This model is termed as semi-fixed network and is formally defined as follows:

- A semi-fixed network $N =< V, E >$ where $V = O \cup H$ (O is a set of observed variables and H is a set of hidden variables) and E is the edge set.
- For each $e_k \in E$, $e_k = (o_i, h_j)$ or (h_j, o_i). Thus, N is a bipartite graph on two partitions O and H.
- For each $o_i \in O$, there is exactly one incoming edge (h_i, o_i) while there can be many outgoing edges.
- For each $h_i \in H$, it has exactly one outgoing edge (h_i, o_i) while there can be many incoming edges$\{(o_{i1}, h_i), (o_{i2}, h_i), ..., (o_{ik}, h_i)\}$. We define $Pa(h_i) = \{o_{i1}, o_{i2}, ..., o_{ik}\}$.

Compared to learning general network with hidden variables, learning the semi-fixed network is easy since the number of hidden variables is fixed and the relationship between hidden variables and observed variables are partially known.

3 Semi-Fixed Structure EM Learning Algorithm

In previous studies, the gene expression datasets are treated as completed datasets (i.e., without missing values and hidden variables). An important property is the decomposability in presence of full data, which means that the probability of a network can be expressed as the production of the probabilities of all independent sub-network:

$$P(X_1...X_n : N_o, D) = \prod_i P(X_i|Pa(X_i) : D) \quad (1)$$

where X_1 to X_n are observed variables and N_o denotes the network without hidden variables. D is a complete dataset.

This property reduces the learning difficulty: with score functions such as minimum description length(MDL) and Bayesian scoring metric, learning the structure of a network can be decomposed to learn each independent sub-network independently[10, 13]. Whereas, in the presence of incomplete data, the decomposability property is not valid and this makes the learning difficult[10]. However, if all parameters of all hidden variables are assigned, hidden variables are observed and thus in a sense, the incomplete dataset becomes "complete".

A useful property of semi-fixed network is that with all parameters been assigned to hidden variables(Note that we know the number of hidden variables and partial relationships of hidden variables and observed variables. It ensures that all hidden variables can be assigned parameters.), the network can be decomposed into independent subnetworks. Each subnetwork comprises of a gene g_j, a hidden variable h_j and a parent set of h_j ($Pa(h_j)$)(as show in Figure 1(b)). The probability is decomposed as:

$$P(g_1...g_n, h_1...h_n : N_{o,h}, D) = \prod_i P(g_j, h_j, Pa(h_j)) \quad (2)$$

$$= \prod_i P(h_i|Pa(h_i) : D) \prod_i P(g_i|h_i) \quad (3)$$

where g_1 to g_n are observed variables (genes), h_i to h_n are hidden variables and $N_{o,h}$ denotes the network with hidden variables. D is an incomplete dataset.

Based on the different decomposition method, the decomposition of Scores is also different (Bayesian score is used here [12]):

$$Score(N_o) = \sum_i Score(g_i|Pa(g_i)) \quad (4)$$

while

$$Score(N_{o,h}) = \sum_i Score(h_i|Pa(h_i)) + \sum_i Score(g_i|Hh_i) \quad (5)$$

The learning procedure tries to maximize the score. Different score functions lead to different network structures. That is, for g_i, $Pa(g_i)$ using score function given by equation 4 may be different from $Pa(H_i)$ using score function as given in eq. 4.

Using this property, we propose a modification to EM algorithm known as semi-fixed structure EM($SSEM$) algorithm to learn such network. The principle of the EM algorithm is: In each iteration, given an initial network structure N_{i-1} and a parameter set θ_{i-1} (which includes both the parameters of observed variables θ^o_{i-1} and hidden variables θ^h_{i-1}), the missing values and hidden variables to build a complete dataset are calculated. The next step is to find a better structure N_i and a better parameter set θ_i based on the new complete dataset.

In this paper, we apply the algorithm to Boolean state gene network model. The missing values and hidden variables can be filled up as follows:

> Given N and θ, for a missing value of a variable v_i, supposing the variable is regulated by $Pa(v_i)$, we compute the value of v_i by the probability distribution of v_i based on $Pr(v_i|Pa(v_i))$ given the value of $Pa(v_i)$ which is observed from the given dataset.

Here is an example:

Example: The variable v_1 has parents v_2 and v_3. As shown in Table 1 (a), there is a missing value in an instance: $\{v_1, v_2, v_3\} = \{(), 1, 1\}$ where () indicate a missing value. Suppose $Pr(v_i = 0|v_2 = 1, v_3 = 1) = 0.4$ and $Pr(v_i = 1|v_2 = 1, v_3 = 1) = 0.6$, we fill up the instance by $\{v_1, v_2, v_3\} = \{(0.4, 0.6), 1, 1\}$. It means, the filled value adds 0.4 count to the case $v_1 = 0$ and 0.6 count to the case $v_1 = 1$. As shown in Table (b), there are 1.4 cases for which $v_1 = 0$ and 2.6 cases for which $v_1 = 1$. In other words, $Pr(v_1 = 0) = 1.4/4 = 0.35$ and $Pr(v_1 = 1) = 2.6/4 = 0.65$.

v_1	v_2	v_3
	1	1
1	0	1
1	1	1
0	1	1

(a)

v_1	v_2	v_3
(0.4, 0.6)	1	1
1	0	1
1	1	1
0	1	1

(b)

Table 1. An example of filling missing values.

The values of hidden variables are treated as missing values too and computed in similar fashion. Thus, hidden variables have the same value set as observed variables. Given the filled dataset, the learning problem is the same as that of learning structure from a complete dataset. The general method to learn the structure from a complete dataset is to decompose the network into independent subnetworks. Then, learn the structure of each subnetwork independently. Since we fix partial structure, we decomposed it as the independent subnetworks, each of which has a target gene, a hidden variable and a parent set of the hidden variable (similar to Figure 1 (b)). The main objective is to find the optimal parents for each hidden variable. Gene network is a sparse network [21]. Learning parents from a big number of candidates is difficult task. The often used trick is to measure the dependencies of candidates to target variable and to choose the best k genes as candidate parents. Then, we search the parents from the candidate parents. Friedman et al.[13] proposed to calculate the dependency by KL-divergence(as Equation 6):

$$MI(X, Y|M) = \sum_{X,Y} \hat{P}(X, Y) \log \frac{\hat{P}(X, Y)}{P_M(X, Y)} \quad (6)$$

where $MI(X, Y|M)$ is the mutual information of X and Y with respect to the network M and $P_M(X, Y)$ is the estimated joint probability of X and Y given M.

However, we cannot observe the probability distribution $P(g_j, h_i)$ to measure the dependency of g_j and h_i. We propose an alternative way to measure the dependency: in each subnetwork, the target gene is the only descendant of the hidden variable. The probabilities of the hidden variable is passed to the target gene. Therefore, $MI(g_j, h_i|M)$ can be reflect by $MI(g_j, g_i|M)$. Thus, the MI can be calculated as following:

$$MI(g_j, g_i|M) = \sum_{g_j,g_i} \hat{P}(g_j, g_i) \log \frac{\hat{P}(g_j, g_i)}{P_M(g_j, g_i)} \quad (7)$$

where $\hat{P}(g_j, g_i)$ is the observed probability distribution of genes g_j and g_i while $P_M(g_j, g_i)$ is the estimated probability distribution of gene g_j and g_i in network M. Performing inference on joint probabilities in a big dataset is quite time intensive. Thus, we approximate the joint probability of g_i and g_j as follows:

- If $g_j \in Pa(g_i)$, then $g_j \rightarrow h_i \rightarrow g_i$. $P_M(g_j, g_i) = P_M(g_i|g_j)P_M(g_j) \approx P_M(g_i|h_i)P_M(h_i|g_j)P_M(g_j)$. Note that in this case, $P_M(g_j, g_i) \neq P_M(g_i, g_j)$.
- Otherwise, g_i and g_j are conditionally independent and $P_M(g_j, g_i)$ can be approximated as $P_M(g_j)P_M(g_i)$.

The detail of the iterative algorithm is as follows:

- In iteration i, give N_{i-1}, θ_{i-1}, the original incomplete dataset D^0 and a decomposable score function $Score(N:\theta, D)$ where D is a complete dataset.

- In E step, the missing values and the values of hidden variables of D^0 are filled up based on the inference of N_{i-1} and θ_{i-1}. Then, we obtain a complete dataset D_{i-1}. The structure N_i is learnt based on D_{i-1} by maximizing $Score(N_i : \theta_{i-1}, D_{i-1})$. Because of the decomposability, we learn the parents for each gene g_i independently:

(1) find k genes with best MI score with respect to g_i to build the candidate parent set CPS.

(2) find parents $PS \in CPS$ which maximum $Score(SS_i : \theta_{i-1}, D_{i-1})$, where SS_i is a substructure $PS \rightarrow h_i \rightarrow g_i$.

- repeat the procedure until converging.

- In M step, θ_i is learnt based on N_i which maximum $Score(N_i : \theta_i, D_{i-1})$.

- Repeat the procedure until converging or the predefined iteration number is reached.

In the above procedure, the network structure and parameters are refined iteratively.

4 Experimental results

The model and the algorithm are implemented in MATLAB and BNT[5]. We present experimental results on artificial and real life gene expression datasets.

4.1 Experiment on artificial datasets

D_{11} and D_{12} are generated by simulating the gene expression with hidden variables. D_{11} contains 5 variables and 6 edges (as shown in Figure 2(b) while D_{12} contains 7 variables and 8 edges (the structure is not shown).

We compare the learning result with traditional learning algorithm $K2$ (a Bayesian network learning algorithm[6]) and structural EM(SEM) (an algorithm in learning Bayesian network with hidden variables[10]). K2 represents the learning algorithm treating the gene expression dataset as complete dataset. SEM adds some hidden variables but does not define the number of hidden variables nor any prior information of the relationships of hidden variables and observed variables[6]. $K2$ and SEM recovered less than 30% correct edges while $SSEM$ recovered more than 75% correct edges in both datasets. The comparison for artificial dataset is shown in Table 2 and Figure 2.

It can be seen from the table that SSEM got more correct edges compared to K2 and SEM.

4.2 Experiment on real-life experiment

The microarray gene expression data *S. Cerevisiae* genome contains 76 gene expression measurements[22]. The expression levels were discretized to discrete values -1, 0 and 1 following the discretization policy of [12]. A value is discretized to -1 if it is smaller than -0.5 and 1 if it is bigger than 0.5, otherwise 0. The real-life gene network in our work is a Yeast transcriptional cell cycle subnetwork published in [14], which includes 15 genes and 21 edges. The regulation relationships for the database are verified by YPD (Yeast Proteome Database)[7][15] and [14], as shown in Figure 3(a). Since there are some time delays existing in the regulation system, the task is to find the edges from consecutive k time slices(k indicates the maximum time delay). For the purpose of this data, an approach based on $k-DBN$ [3] is implemented for the comparison purpose. The comparison is shown in Table 3 and Figure 2(Only the original structure and the structure learnt by SSEM are presented.).

Algorithm	D_{11}		D_{12}	
	TE	CE	TE	CE
$K2$	2	0	4	1
SEM	3	1	5	2
$SSEM$	5	3	9	7

Table 2. TE indicates total learnt edges which is the number of edges the algorithm learnt from the dataset while CE indicates the correct edges that are learnt.

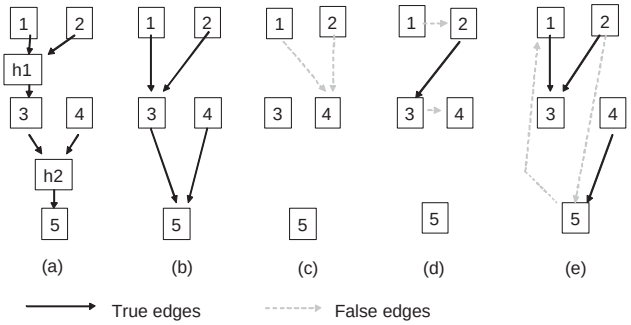

Figure 2. Leaning performance comparison in D_{11} dataset. (a) Original structure with hidden variables. (b) Structure with removing hidden variables. (c) structure learnt by K2. there is no correct edge. (b) structure learnt by SEM. There is only 1 correct edge. (d) structure learnt by SSEM. 3 correct edges plus 2 false edges.

Algorithm	Yeast gene subnetwork	
	TE	CE
$K2$	41	2
SEM	16	2
$k-DBN$	31	6
$SSEM$	33	14

Table 3. TE(Total learnt edges) indicates the number of edges the algorithm learnt from the dataset and CE(the correct edges) indicates the number of correct learnt edge.

The effectiveness of semi-fixed structure EM learning algorithm can be seen as compared to $k-DBN$ algorithm as $SSEM$ takes advantage of its semi-fixed structure and effective learning algorithm.

[5] Bayes network toolbox, http://www.ai.mit.edu/∼murphyk

[6] In the experiments, several numbers of hidden variables are predefined for SEM and the one with best performance is selected as the result.

[7] http://www.proteome.com/YPDhome.html

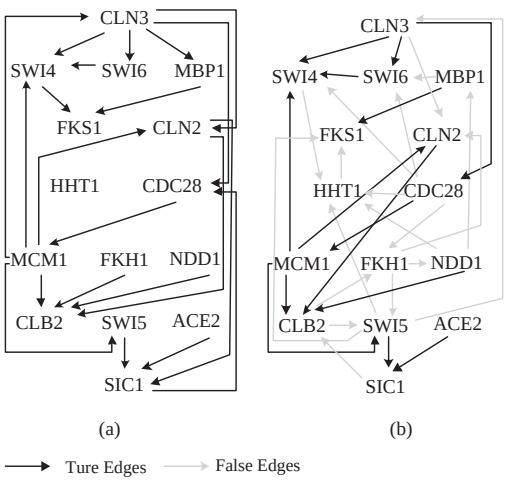

Figure 3. Leaning performance of SSEM in real-life gene network. (a) The original subnetwork structure. (b) The structure learnt by $SSEM$. 14 edges are learnt.

5 Conclusion

The semi-fixed hidden variable model introduces hidden variables to model the important components of gene network, i.e., proteins. The model makes the network decomposable and parts of the network are fixed using biological knowledge. Compared to the current work on gene network, we integrate the biological knowledge to build the semi-fixed hidden variable model which is effective and reflects the real life system. An effective learning algorithm is presented in the paper suitable to learn such partially fixed network. The main disadvantage of the model is that it needs more computing resource to control the hidden variables and requires more iterations to converge.

REFERENCES

[1] T. Akutsu, S. Miyano, and S. Kuhara, 'Identification of Genetic Getworks from a Small Number of Gene Expression Patterns Under the Boolean Network Model', in *PSB*, pp. 17–28, (1999).

[2] G.H. Barbara, A. Jameson, and F. Witting, 'Learning Bayesian Networks with Hidden Hariables for User Modeling', in *Proceedings of the IJCAI 99 Workshop "Learning About Users"*,, (1999).

[3] X. Boyen, N. Friedman, and D. Koller, 'Discovering the Hidden Structure of Complex Dynamic Systems', in *UAI*, (1999).

[4] T. Chen, V. Filkov, and S.S. Skiena, 'Identifying Gene Regulatory Networks from Exprimental Data', in *RECOMB*, (1999).

[5] T. Chen, H.L. He, and G.M. Church, 'Modeling Gene Expression with Differential Equations', (1999).

[6] G.F. Cooper and E. Herskovits, 'A Bayesian Bethod for The Induction of Probabilistic Networks from Data', *Machine Learning*, **9**, 309–347, (1992).

[7] M. Cumiskey, J. Levine, and D. Armstrong. Gene Network Reconstruction Using a Distributed GA with a Backprop Local Search.

[8] G. Elidan, N. Lotner, N. Friedman, and D. Koller, 'Discovering Hidden Variables: A Structure-Based Approach', in *NIPS*, pp. 479–485, (2000).

[9] N. Friedman, 'Learning Belief Networks in The Presence of Missing Values and Hidden Variables', in *Proc. 14th International Conference on Machine Learning*, pp. 125–133. Morgan Kaufmann, (1997).

[10] N. Friedman, 'The Bayesian Structure EM Algorithm', in *UAI*, (1998).

[11] N. Friedman, 'Inferring Cellular Networks Using Probabilistic Graphical Models', *Science*, **33**, 799–805, (2004).

[12] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, 'Using Bayesian Networks to Analyze Expression Data', in *RECOMB*, pp. 127–135, (2000).

[13] N. Friedman, I. Nachman, and K. Peer, 'Learning Bayesian Network Structure from Massive Datasets: the "Sparse Candidate" Algorithm', pp. 206–215, (1999).

[14] B. Futcher, 'Transcriptional Regulatory Networks and Yeast Cell Cycle', *Current Opinion in Cell Biology*, **14**, 676–683, (2002).

[15] P.E. Hodges, A.H. McKee, B.P. Davis, W.E. Payne, and J.I. Garrels, 'The Yeast Proteome Database(YPD): a Model for The Organization and Presentation of Genome-Wide Functional Data', *Nucleic Acids Research*, **27**(1), 69–73, (1999).

[16] S. Imoto, T. Goto, and S. Miyano, 'Estimation of Genetic Networks and Functional Structures Between Genes by Using Bayesian Networks and Nonparametric Regression', in *PSB*, (2002).

[17] S. Imoto, T. Higuchi, T. Goto, K. Tashiro, S. Kuhara, and S. Miyano, 'Combining Microarrays and Biological Knowkedge for Estimating Gene Networks via Bayesian Network', in *CSB*, (2003).

[18] F.A. Kolpakov, E.A. Ananko, G.B. Kolesov, and N.A. Kolchanov, 'Genenet: a Gene Network Database and its Automated Visualization', *Bioinformatics*, **14**, 529–537, (1998).

[19] K. Murphy and S. Mian, 'Modelling Gene Expression Data Using Dynamic Bayesian Networks', *Technical Report*, (1999).

[20] P.D'Haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, 'Linear Modeling of mRNA Expression Levels During CNS Development and Injury', in *PSB*, pp. 41–52, (1999).

[21] E.P.V. Someren, L.F.A. Wessels, and M.J.T. Reinders, 'Linear Modeling of Genetic Networks from Experimental Data', in *ISMB*, pp. 355–366, (2000).

[22] P.T. Spellman, G. Sherlock, and B. Futcher, 'Comprehensive Identification of Cell Cycle-Regulated Genes of the Yeast Saccharomyces Cerevisiae by Microarray Hybridization', *Molecular Biology of the Cell*, **9**, 3273–3297, (1998).

[23] A. Wagner, 'Estimating Coarse Gene Network Structure from Large-Scalse Gene Perturbation Data', in *Intl. Workshop on Genome Informatics*, (2001).

[24] S. Watanabe, 'Algorithms for Inference of Genetic Networks AIGNET', in *Intl. Workshop on Genome Informatics*, (1998).

[25] L.F.A. Wessels, E.P. Van Someren, and M.J.T. Reinders, 'A Comparison of Genetic Network Models', in *PSB*, volume 6, pp. 508–519, (2001).

[26] M. Yaki, D. Tominaga, M. Okamoto, S. Watanabe, and Y. Eguchi, 'Development of a System for the Inference of Large Scale Genetic Networks', in *PSB*, pp. 446–58, (2001).