

Learning Causal Networks from Microarray Data

Nasir Ahsan[†]

Michael Bain^{†*}

John Potter[†]

Bruno Gaëta^{†‡}

Mark Temple[‡]

Ian Dawes[‡]

[†] School of Computer Science and Engineering

[‡] School of Biotechnology and Biomolecular Sciences

University of New South Wales,

Sydney, Australia 2052

* Email: mike@cse.unsw.edu.au

Abstract

We report on a new approach to modelling and identifying dependencies within a gene regulatory cycle. In particular, we aim to learn the structure of a causal network from gene expression microarray data. We model causality in two ways: by using conditional dependence assumptions to model the independence of different causes on a common effect; and by relying on time delays between cause and effect. Networks therefore incorporate both probabilistic and temporal aspects of regulation. We are thus able to deal with cyclic dependencies amongst genes, which is not possible in standard Bayesian networks. However, our model is kept deliberately simple to make it amenable for learning from microarray data, which typically contains a small number of samples for a large number of genes. We have developed a learning algorithm for this model which was implemented and experimentally validated against simulated data and on yeast cell cycle microarray time series data sets.

1 Introduction

With the complete genomic sequences of increasing numbers of organisms available there are unprecedented opportunities for the biological analysis of complex cellular processes. In this new era of cell biology there have been major advances in the ability to collect data on a genome-wide scale by the use of high-throughput technology such as gene expression microarrays. However this data requires analysis beyond the level of individual genes; we need to investigate networks of genes acting within highly regulated systems.

Owing to the complexity of the systems to be studied a wide range of methods to model networks and learn them from data have been studied (Endy & Brent 2001, de Jong 2002). However, in our work we are not aiming to model the full dynamical systems of the cell, but rather to learn key causal features from data.

In this paper we investigate the use of causal networks to model relations between genes as measured in microarray data. More specifically, we explore learning the structure of a genetic regulatory network in this setting. Our approach to causal networks is based on a form of dynamic Bayesian network in which causality is represented in two ways: by relying on conditional independence assumptions to model the independence of different causes on a common effect; and by relying on time delays between cause and effect. By incorporating time delays into our model,

we are able to deal with cyclic dependencies amongst genes. From a high-level perspective, we address the following problems:

1. How can we formalize the model of a causal network which combines probabilistic and temporal aspects, in particular, conditional independence and temporal precedence?
2. How can we learn such a model from observations of the variables over time in a robust and computationally efficient manner?

The paper is organised as follows. In Section 2 we introduce our representation for causal models, and in Section 3 we develop a learning algorithm for such models. In Section 4 we present results from experimental application of the approach to cell cycle time series microarray data.

2 A new probabilistic model

We introduce a probabilistic model that satisfies certain assumptions and addresses the problems of complexity discussed above. We then extend this probabilistic model to include time, and develop methods for estimating our probabilistic model from time series data, specifically microarray data. Note that in this paper our models contain only discrete random variables.

2.1 The \mathcal{F} -model

In the Bayesian network model an effect is conditionally independent of its ancestors given its immediate causes. This allows one to model an effect θ with immediate causes β_1, \dots, β_n as shown on the left in Figure 1. Unfortunately modelling an effect in this way requires the estimation of at least 2^n parameters, where n is the number of causes. Clearly the number of probabilities to be estimated increases exponentially in n .

However, if certain combinations of causes are known *a priori* to be unlikely they may be safely ignored in the interests of simplifying the problem. This suggests a way to reduce the number of distinct events that need to be modelled, effectively compressing the space of events for which probabilities need to be estimated. To do this we will use a function \mathcal{F} , called a *compression function*, to compress the joint probability distribution. In the standard Bayesian network framework the conditional probability for the dependence of an effect θ on its causes β_1, \dots, β_n is $P(\theta | \beta_1, \dots, \beta_n)$. Using the compression function this becomes $P(\theta | \mathcal{F}(\beta_1, \dots, \beta_n))$, as shown on the right in Figure 1.

If we assume that $P(\theta | \beta_1, \dots, \beta_n) \equiv P(\theta | \mathcal{F}(\beta_1, \dots, \beta_n))$, for some arbitrary \mathcal{F} , a joint probability distribution may be decomposed using \mathcal{F} -based conditional independence factors.

Definition 1 (\mathcal{F} -model) Let M denote a probability model over the random variable space $\Theta =$

$\{\theta_1, \dots, \theta_n\}$, and let \mathcal{F} be some arbitrary mapping over observations of Θ . An \mathcal{F} -model is the pair $\langle \Theta, G \rangle$, where G is a directed acyclic graph and each edge in G represents a dependence between a pair of variables, letting $\alpha \equiv \beta_1, \dots, \beta_n$, such that the set of immediate parents $\alpha_i \subset \Theta$ of $\theta_i \in \Theta$ in G render it independent of its other ancestors in G . The \mathcal{F} -model defines the following factorization of M :

$$P(\Theta) = \prod_{i=1}^n P(\theta_i | \mathcal{F}(\alpha_i))$$

where $P(\theta_i | \mathcal{F}(\alpha_i)) \equiv P(\theta_i | \alpha_i)$.

Effectively, for a set of random variables $\{\theta_i, \dots, \theta_j\}$, \mathcal{F} induces a new random variable $\mathcal{F}(\theta_i, \dots, \theta_j)$. Note that if our assumption above is valid with respect to the system under enquiry then the \mathcal{F} -model, by definition, is equivalent to a Bayesian network, assuming no cycles in the ancestor relation.

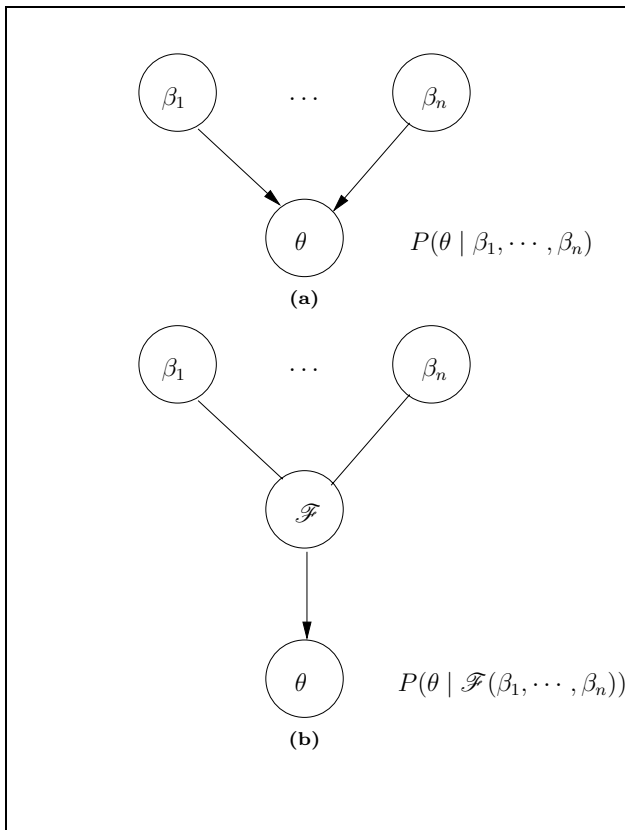


Figure 1: Modelling via (a) Bayes net dependencies and (b) \mathcal{F} based dependencies.

2.2 Temporal extension of the \mathcal{F} -model

In order to model a causal system we must account for time. Therefore we provide a temporal generalization of the \mathcal{F} -model that accounts for temporal precedence and contiguity in time (Hume 1999). We extend the \mathcal{F} -model, by replacing the set of random variables Θ with a corresponding random process $\Theta(t)$ and its history at any time.

Definition 2 (Discrete Time Random Process and History) $\Theta(t) = \{\theta_1(t), \dots, \theta_n(t)\}$ is a discrete time random process for $t \in \mathbb{Z}$. At time t the history of the random process is $\mathcal{H}(t) = \bigcup\{\theta_i(\tau) | i \in 1, \dots, n, \tau < t\}$.

A causal model for $\Theta(t)$ requires the parent set $\alpha(t)$ for some $\theta(t)$ to be drawn from the history

$\mathcal{H}(t)$. Each random process $\theta_i(t)$ has a finite set of parents $\beta_{i_1}, \dots, \beta_{i_m} \subseteq \Theta$ with associated time shifts $\delta_{i_1}, \dots, \delta_{i_m}$. Thus we may write the parent set $\alpha_i(t) = \{\beta_{i_1}(t - \delta_{i_1}), \dots, \beta_{i_m}(t - \delta_{i_m})\} \subseteq \mathcal{H}(t)$, allowing us to model causal relationships by the conditional probabilities $P(\theta_i(t) | \alpha_i(t))$.

We also make a *stationarity assumption*, i.e., we assume that the underlying causal relationships do not change over time. This is clearly not the case for cellular systems, but acts as a useful simplifying assumption. It follows that the parent sets $\alpha(t) = \langle \beta_1(t), \dots, \beta_m(t) \rangle$ and associated delays δ_i are independent of time t .

Hence the \mathcal{F} -model represents a stationary distribution by a graph where nodes are variables in Θ and edges are labelled with time shifts. Note that this graph may include cycles. However, since time shifts are positive, these cycles do not represent a cyclic probabilistic dependency, but merely that $\theta(t)$ depends on $\theta(t - \lambda)$, where λ is the period of a single cycle. Note that cycles are not permitted in the standard Bayesian network formalism which is restricted to acyclic graphs.

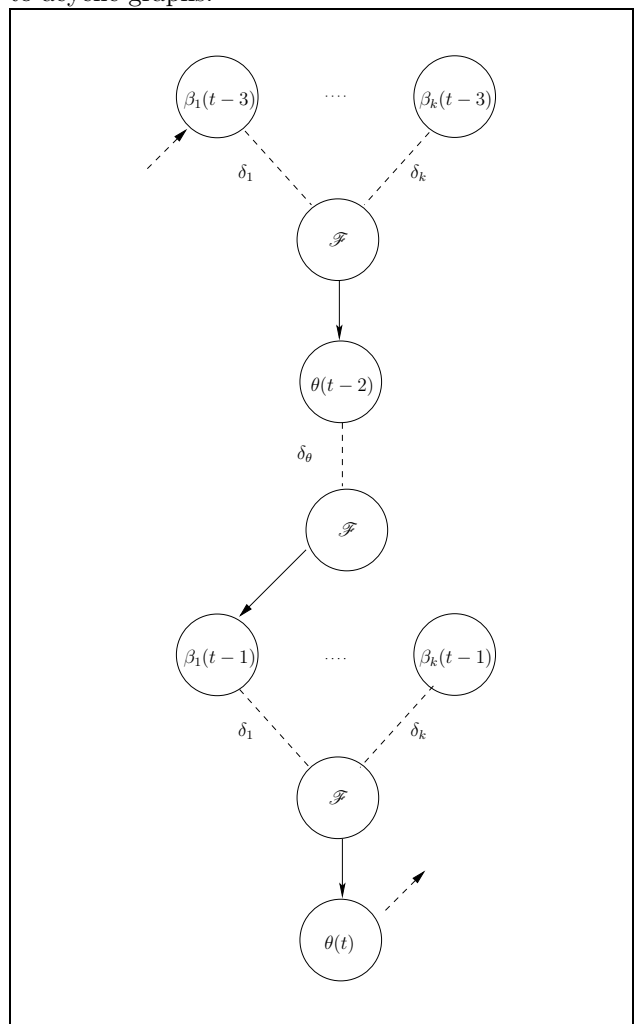


Figure 2: A temporal \mathcal{F} -model “unrolled” in time: each $\delta_i = 1$ time unit, and the cycle period = 2 time units.

In high dimensional spaces such as microarray data many independent processes may run concurrently and there may exist various causal chains, each with different cycle periods. Hence assuming a global cycle period on all variables of the system is too restrictive. Therefore we use a *relative cycle period* for each variable.

Definition 3 (Cycle period) Let $X(t) = \langle x_1, \dots, x_m \rangle$ be an expression vector and

$\lambda \in \{1, \dots, m\}$. Then the cycle period for $X(t)$ is the first off-zero peak correlation, i.e., the first λ for which

$$\arg \max_{\lambda \in \{1, \dots, m\}} \left(\sum_{t=1}^{m-\lambda} X(t) \cdot X(t+\lambda) \right) > 0$$

As mentioned in Section 2 we use discretized data. In (Friedman, Linial, Nachman & Pe'er 2000) a discretization was used where any log expression ratio outside a central band of ± 0.5 was taken as gene activity. However, this will tend to ignore low-amplitude gene expressions which could form part of regulatory interactions, e.g., certain transcription factors. Therefore we discretize each expression vector separately using a relative threshold: expression values greater (resp. less) than the threshold are mapped to 1 (resp. -1), otherwise they are mapped to zero. The relative threshold is set via a k^{th} order statistic, where k is a parameter to the algorithm.

Finally, the compression function \mathcal{F} combines a set of variables (candidate causes) into a single variable. In application to microarray data the variables are time-shifted, discretized gene expression profiles representing candidate regulators of a selected target gene. For our implementation we used the following compression function:

Definition 4 Given a set of expression values $X \equiv \langle x_1, \dots, x_n \rangle$ for n genes at some point in time the compressed version of X is:

$$\mathcal{F}(x_1, \dots, x_n) = \begin{cases} +1, & \text{if } |X^+| \geq n - g \\ -1, & \text{if } |X^-| \leq n - g \\ 0, & \text{otherwise} \end{cases}$$

where $X^+ = \{x \in X \mid x = +1\}$, and $X^- = \{x \in X \mid x = -1\}$ and $0 \leq g < n/2$ is a slack parameter to allow some variation in + or -.

The slack parameter is to allow for noise. This compression function is applied to all columns in the set of candidate causes, giving a compressed variable. Note that as the number of variables n increases, the compressed values either remain at +1 or -1, or go to zero. This helps to avoid overfitting of the model: adding too many variables as causes will tend to reduce the compressed value to zero, which implies the model loses any dependency between the candidate causes and the effect. We only add genes as candidate causes whose expression profiles are closely similar to the target gene. Further details on the compression function \mathcal{F} are in Ahsan (2006).

3 A local learning algorithm

The key problem investigated here may be formalized as: Given a data set D , find an \mathcal{F} -model which adequately explains D . Since computational efficiency is a key issue for microarray data which has many variables we propose searching for *local neighborhoods* instead of searching for an entire network, as conditional independence may be efficiently computed.

For directed graphs this problem simplifies to searching for the immediate parents of a variable. Various definitions of an immediate parent have been used, e.g., (Pearl 1988, Margaritis & Thrun 2004, Koller & Sahami 1996). For us, however, immediate parents (a) temporally precede the target, (b) significantly correlate with the target, and (c) render the target independent of all its ancestors. Based on this definition we propose the following local learning algorithm.

Algorithm 1 learns causes of an effect, and is divided into two phases. Phase 1 filters out a set of

Algorithm 1 Learning Immediate Causes of an Effect: PIA

Input: Initial candidates Θ , Effect θ and Threshold τ for score

Output: Set of immediate causes for θ

```

1: PHASE 1: Selecting a Set of Candidate Causes for  $\theta$ 
2:  $\alpha \leftarrow \{\}$ 
3: for  $\beta \in \Theta$  do
4:   if  $I(\theta; \beta) > \tau$  then
5:      $\alpha \leftarrow \alpha \cup \{\beta\}$ 
6: PHASE 2: Learn local neighborhood of  $\theta$ 
7: Sort  $\alpha$  according to  $\succ_{\theta}$  //see text for details
8:  $\alpha' \leftarrow \{\}$ 
9: //Seek causes that increase the score given the current  $\alpha$ 
10: repeat
11:    $\beta \leftarrow \max(\alpha)$ 
12:   if  $I(\theta; \mathcal{F}(\alpha', \beta(t - \delta_{\theta\beta}))) - I(\theta; \mathcal{F}(\alpha')) > 0$  then
13:      $\alpha' \leftarrow \alpha' \cup \{\beta\}$ 
14:    $\alpha \leftarrow \alpha - \{\beta\}$ 
15: until  $\alpha = \emptyset$ 
16: return  $\alpha'$ 

```

plausible candidates based on a pairwise score between each candidate and the effect (lines 3-5). The score used was the mutual information $I(X; Y)$ between a pair of variables (Kullback & Leibler 1951).

In phase 2 the algorithm consists of two further stages. At line 7 the algorithm first sorts the filtered candidates α with respect to the following order: $\beta_1 \succ_{\theta} \beta_2$ iff either $\delta_{\theta\beta_1} < \delta_{\theta\beta_2}$, or $\delta_{\theta\beta_1} = \delta_{\theta\beta_2}$ and $I(\theta, \beta_1) > I(\theta, \beta_2)$. This ordering relies on computation of *delays* or time shifts δ_{XY} between variables X and Y . The time shift between a pair of variables is simply the time difference between the respective peaks in each time series. Sorting in this way implements criteria (a) and (b) of the definition of immediate parents.

The algorithm then iteratively processes each cause according to the order \succ_{θ} in a greedy manner (lines 10-15). At each step the current candidate cause β is added to the set of immediate causes of θ if this results in an increase in the mutual information score between θ and the updated compressed variable $\mathcal{F}(\alpha', \beta(t - \delta_{\theta\beta}))$. This test implements criterion (c) of the definition of immediate parents. The notation $\beta(t - \delta_{\theta\beta})$ denotes the application of a variable-specific time shift before computing the score. Note that our selection procedure is biased towards variables with shorter time delays; the rationale for this is that longer time delays can be caused by a chain of causes in the network, whereas shorter ones cannot.

The temporal \mathcal{F} -model is based on the assumption that the time shift for any cycle in an \mathcal{F} -model is a multiple of the period λ , and that any path with the time shift greater than λ must include a cycle. However, target variables may vary in λ , and therefore simply piecing all local neighborhoods will not result in a true \mathcal{F} -model. Currently, no constraint has been enforced on the local learning algorithm PIA which would allow one to induce a global structure in a modular manner. However, one may induce \mathcal{F} -models by grouping target variables with similar cycle periods, and post-pruning any edges that conflict with cyclicity assumptions made in Section 2.2. This approach, called the Piece-wise Network Induction Algorithm (PWNIA), was used for the experiments in the next section.

4 Experimental results

A major issue in current research on learning genetic regulatory networks from genome-wide data is that there are no reference “ground truth” models. However, we attempted to validate our approach in two ways: by reconstruction experiments using simulated data; and on real data by using selected sets of genes and evaluating features of learned network structures against known properties of the cellular system.

4.1 Simulated data

We conducted simulation experiments by implementing a simple model of cell-cycle gene expression. Input to the simulator was in the form of a directed graph, modelling a genetic regulatory network, plus properties of each edge in the graph, representing a gene interaction, such as cycle length, time delays, etc. For each edge $A \rightarrow B$ in the graph, the dependence of B on A was modelled as a probabilistic, time-delayed function of A , with added Gaussian noise. Variables with no parents (root nodes) were modelled as sine functions with added Gaussian noise. The simulator was implemented to enable manipulation of a number of parameters, such as the amount of added noise, the sampling frequency of data, etc.

Graphs were generated randomly using the preferential attachment model of Albert and Barabasi (1999) implemented as part of the Python random graph library NetworkX¹. The edges of these undirected graphs were directed arbitrarily, avoiding cycles.

For each randomly-generated graph and a set of associated probabilistic dependencies, the simulator was used to generate multiple time series data sets to which the PWNIA algorithm was applied to learn a network. Learned networks were then compared with the originals to evaluate the reconstruction. Features of the reconstructed networks were evaluated using a version of the approach in Friedman et al. (1999) adapted for time series data.

We investigated the effect on learning of (a) adding varying amounts of noise, (b) varying the number of time points, (c) varying network density, and (d) varying parameters of the learning algorithm. Precision and recall measures were recorded for recovery of order and Markov relations (Friedman, Goldszmidt & Wyner 1999). Order relations are true for gene-gene interactions $A \rightarrow B$ where A is an ancestor of B . However, Markov relations are more stringent, containing only the subset of genes that make a gene probabilistically independent of all other genes in the network. Note that precision and recall curves were generated as opposed to the use of either sensitivity/specificity or ROC analysis due to the difficulty of generating true negatives for order and, particularly, Markov relations in our experimental setup.

In terms of precision; which reflects the extent to which predicted gene-to-gene relationships made by our learning algorithm were correct, our results were similar for both order and Markov relations over experimental conditions (a) to (d). However, for recall; the extent to which actual relationships were predicted by our algorithm, we found that order relations were much easier to reconstruct than Markov relations.

For example, condition (a), adding noise to the generation of simulated data, led to reductions in precision (from 0.9 to 0.6 for order relations and from 0.85 to 0.65 for Markov relations) and greater reductions in recall (from 1.0 to 0.7 for order relations and from 0.45 to 0.25 for Markov relations), all measures are rounded mean values for 30 data sets. Since recall for order relations was higher than for Markov relations, it follows that it should be easier on real data to

discover general causal ordering relations than more detailed gene-to-gene regulatory interactions. This formed the basis for the experiments on real microarray data discussed below.

Real microarray data contains typically only a few time points, since each time point represents a separate microarray experiment, which is time-consuming and relatively expensive to carry out. With simulated data, however, the number of data points measured can be varied. We found that under experimental conditions (b) and (c), increasing the number of data points per cycle, led to increases in both recall and precision, except for precision on reconstruction of the denser networks. This was as expected, since a chance correlation is more likely because more variables have shorter time shifts and hence appear near the top of the order \succ_{θ} , leading to possible inclusion as immediate causes. Experimental condition (d), varying the similarity threshold above which genes are considered potential parents, showed an inverse relationship for both order and Markov relations, i.e., recall fell while precision rose as the threshold was increased. This suggests that the threshold may be acting to control the number of false positives, although at the expense of coverage. If so this could be a useful property of the algorithm and should be investigated further.

An additional parameter, not investigated due to lack of time, that may be useful in reducing the effect of noise on the algorithm is the *slack parameter*. This has the effect of relaxing the strict requirement of direct or inverted similarity between a gene and its candidate regulators. As part of future work this should be investigated in combination with the similarity threshold, as the former may compensate for the reduction in recall due to the application of the latter. Lastly, we have implemented a fixed form of compression function \mathcal{F} . This may not be the most appropriate for learning genetic regulatory interactions, and it would be interesting to investigate the possibility of learning the compression function from data.

The simulation experiments are described in more detail in Ahsan (2006). We concluded from these results on simulated data that the algorithm shows promise for the discovery of gene relationships, although it is likely to be susceptible to noise and low numbers of instances.

4.2 Yeast cell cycle data

Since it is not currently possible to evaluate network learning attempts against a “real” genetic regulatory network, even in well-studied organisms such as the budding yeast *Saccharomyces cerevisiae*, on real data we took the approach of examining key *temporal* features. At this stage we are only attempting to validate our approach rather than generate biologically useful knowledge. Therefore, following the results discussed above on learning order relations on simulated data we investigated the extent to which the edges in a learned network reflect known temporal features of the domain.

Our starting point was the seminal experimental work on the budding yeast cell cycle by Spellman et al. (1998). In this work the cell cycle was arrested by different means and on release the cells went through one to two synchronised cell cycle iterations, during which microarray gene expression measurements were taken at regular intervals. Using a Fourier analysis-based scoring function, calibrated to known cell-cycle regulated genes, 800 genes were determined as being cell-cycle regulated in terms of their gene expression. In addition, a temporal ordering was applied to sort these 800 genes into one of five cell cycle *phases*, G1, S, S/G2, G2/M or M/G1. This provides a benchmark, similar to the order relations from our simulated data, against which to evaluate our algorithm.

The objective is as follows: given a yeast cell-

¹<https://networkx.lanl.gov>

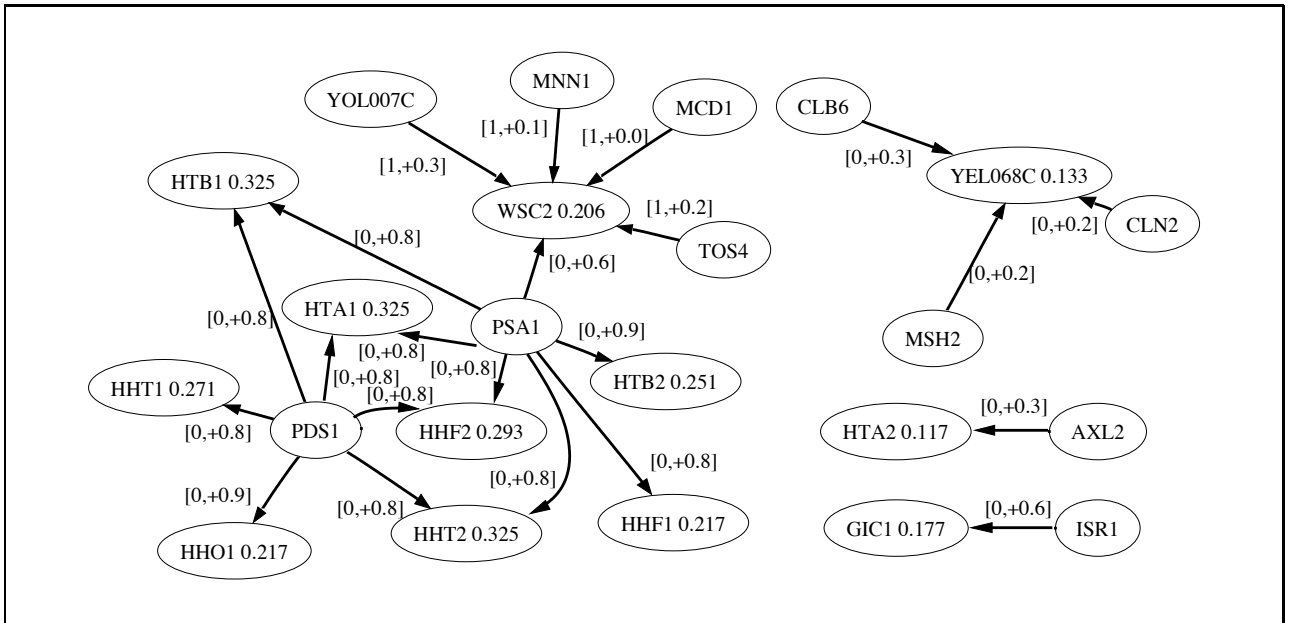


Figure 3: A network learned from the top-scoring 135 genes on the alpha cell cycle data set. Each node with an in-arrow is labelled by its mutual information with the compressed parent set. Each edge from parent to target represents a G1 to S phase interaction and is labelled with 2 numbers: time shift and pairwise correlation coefficient (see text for details).

cycle microarray time series data set, run the network learning algorithm and compare the edges thus obtained to the known temporal ordering in terms of their *phase coherence*. Phase coherence is defined by a set of rules stating the biological plausibility of each possible phase labelling of edges in a network graph. For example, if the cause and effect were both in the same phase, this would be acceptable, whereas if the cause and effect were separated by a large number of phases, this is unlikely to be a biologically valid causal relationship.

The rules were:

Cause phase	Possible effect phases
G1	S, S/G2 or G2/M
S	S/G2, G2/M or M/G1
S/G2	G2/M or M/G1
G2/M	M/G1 (current cycle) or G1 (next)
M/G1	G1 (next cycle) or S (next cycle)

This defines 12 out of a possible 25 cause-effect phase relations. Allowing the 5 same-phase relations gives a total of 17 permitted by these rules. Notice that the last two phases in the cell cycle, G2/M and M/G1, have causal relations that cross the cell-division boundary.

To test the phase coherence of the 0 learned by our algorithms we performed a number of network learning experiments using three yeast cell-cycle time series data sets from the work of Spellman et al. (1998). Since our algorithm requires microarray data containing reasonable cyclic expression profiles, we restricted attention to the 800 genes determined to be cell cycle regulated. We ranked these in decreasing order of their aggregate score – known as the “CDC score” – generated by Spellman et al. (1998). The higher the CDC score, the clearer the periodic “signal” in the microarray data. This score is calibrated to genes known to be cell-cycle regulated, and ranges from 15.990 (maximum) to 1.314 (minimum). We set two thresholds on this score: ≥ 5.0 and ≥ 3.0 . There are 135 genes above the first threshold and 297 genes above the second. This gave us the basis to construct three sets of genes, in decreasing order of “cyclicality”, of size 135, 297 and 800.

We ran our network learning algorithm with default parameter settings for the three gene sets on each of the three microarray data sets known as *alpha*, *cdc15* and *cdc28*. (We did not use the *elutriation* data set since it does not comprise two complete cell cycles and our algorithm currently requires ≥ 2 cycles.) The set of edges from each learned network was filtered to select only those having a Pearson pairwise correlation above a certain correlation threshold (results shown are for $r \geq 0.7$). Nodes in each set of “well-correlating” edges was then labelled their respective phase. This phase labelling was then evaluated for coherence against the above rules.

Data		135	297	800
alpha	corr.	0.75 (239)	0.73 (508)	0.63 (1154)
	legal	0.91 (180)	0.90 (369)	0.85 (726)
cdc15	corr.	0.85 (217)	0.78 (498)	0.56 (1592)
	legal	0.91 (184)	0.89 (387)	0.83 (888)
cdc28	corr.	0.71 (224)	0.70 (490)	0.59 (1383)
	legal	0.91 (154)	0.89 (344)	0.87 (816)

Table 1: Results from phase coherence tests on edges from learned networks on 3 cell-cycle regulated data sets. Shown are the proportions (totals) of well-correlating edges and phase-coherent edges for 3 gene sets of increasing size and reducing overall quality (see text for details).

The results are summarised in Table 1. For each data set there are two rows, labelled “corr.” and “legal”. The “corr.” row contains the proportion (total in brackets) of edges above the correlation threshold. The “legal” row contains the proportion (total in brackets) of edges that are phase coherent, i.e., are in accord with the rules above. The three columns refer to the three different-sized gene sets.

The results show that the algorithm is constructing network graphs containing a majority of well-correlated edges. In addition, the phase-coherence of the edge sets remains high even on the largest data set (800) which produces networks with many low-correlating edges. Note that the pairwise correlation

is not the necessarily a good measure of functional relatedness, since it ignores multi-gene dependencies, unlike the mutual information score used by our algorithm. However, we use it here as a useful heuristic, since it could easily form a pre-processing step for our algorithm. Note also that the algorithm is *not* given the phase information – phase coherence is used solely to assess performance.

We are continuing to work on examining the phase coherence of this approach. For example, we can isolate genes from different phases to use as candidate cause and effect sets to initialise our algorithm. The edge sets can then illustrate how well temporal ordering is being recovered during learning. Figure 3 shows one network, learned by setting the candidate causes to G1-phase genes in the top-scoring 135 and the candidate effects to S-phase genes from the same set. So far we have not compared the performance of this algorithm with alternative approaches, but this should be undertaken as part of future work. We believe that ideas such as phase coherence may provide useful methods to compare performance of such algorithms in the absence of a “gold standard”.

5 Conclusions

In this paper we have developed a novel framework for modelling and learning causal gene-to-gene interactions from microarray data. Our approach extends the standard Bayesian network formalism to allow temporal relations and enable cyclic dependencies which is a critical requirement for the representation of biological regulation. Some key simplifying features of the approach are that gene expression is discretized, and each gene has a set of immediate causes, with an associated time shift, on its gene expression. Furthermore, the effect of multiple causes is combined into a single variable, also discretized, via a novel compression function.

The dependence of the gene on its compressed parent, which precedes it in the temporal ordering, is modelled with a simple discrete probability distribution. Our learning algorithm focuses on identifying the immediate causes of a given gene. The compression function limits the introduction of new parents through its behavior in losing information with more parents, and fixes the number of parameters to be learnt (avoiding over-fitting).

Experimental validation is promising, based on simulated and real data. On real data we demonstrated that the networks learned are largely consistent with the temporal ordering properties of the cell cycle, reinforcing some results from experiments on simulated data.

The extent to which learned interactions are spurious (false positives) is not known. This is a key area for future work. We believe that combining microarray data with data from other sources, such as protein-protein interactions, has potential for improving this aspect of our approach. For example, some critical aspects of function are clearly absent from microarray data. An example is the gene CDC28, which is central to cell-cycle regulation, but assumed to be expressed in excess and therefore will not be learned as part of any interaction from microarray data alone, although it is well-represented in protein interaction databases.

References

Ahsan, N. (2006), Learning Causal Networks from Gene Expression Data (Submitted), Master’s thesis, University of New South Wales.

Barabasi, A. L. & Albert, R. (1999), ‘Emergence of scaling in random networks’, *Science* **286**(5439), 509 – 512.

de Jong, H. (2002), ‘Modeling and Simulation of Genetic Regulatory Systems: A Literature Review’, *Journal of Computational Biology* **1**(9), 67–103.

Endy, D. & Brent, R. (2001), ‘Modelling cellular behaviour’, *Nature* **409**, 391–395.

Friedman, N., Goldszmidt, M. & Wyner, A. (1999), Data analysis with bayesian networks: A bootstrap approach, in ‘Proceedings of UAI99’, pp. 196–205.

Friedman, N., Linial, M., Nachman, I. & Pe’er, D. (2000), Using Bayesian networks to analyze expression data, in ‘RECOMB’, pp. 127–135.

Hume, D. (1999), *Enquiry concerning Human Understanding*, Oxford/New York: Oxford University Press.

Koller, D. & Sahami, M. (1996), Toward optimal feature selection, in ‘International Conference on Machine Learning’, pp. 284–292.

Kullback, S. & Leibler, R. A. (1951), ‘On information and sufficiency’, *Ann. Math. Statist.* pp. 22:79–86.

Margaritis, D. & Thrun, S. (2004), ‘Bayesian network induction via local neighborhoods’, *Advances in Neural Information Processing Systems 12*.

Pearl, J. (1988), *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*, Morgan Kaufmann.

Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botstein, D. & Futcher, B. (1998), ‘Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast *Saccharomyces cerevisiae* by Microarray Hybridization’, *Molecular Biology of the Cell* **9**, 3273–3297.