Experiments on the Accuracy of Algorithms for Inferring the Structure of Genetic Regulatory Networks from Microarray Expression Levels[1]

Frank C. Wimberly, Institute for Human and Machine Cognition, University of West Florida

Thomas Heiman, School of Computational Sciences, George Mason University

Joseph Ramsey, Department of Philosophy, Carnegie Mellon University

Clark Glymour, Department of Philosophy, Carnegie Mellon University and Institute for Human and Machine Cognition, University of West Florida

Abstract

After reviewing theoretical reasons for doubting that statistical/machine learning methods can accurately infer gene regulatory networks from expression measures, we test 10 algorithms on simulated data from the sea urchin network, and on microarray data for yeast compared with recent experimental determinations of the regulatory network in the same yeast species. We find most algorithms are at chance for determining the existence of a regulatory connection between gene pairs, and the performance of better algorithms degrades as simulations become more realistic, in accord with theory.

1. Motivation. The development of microarray techniques for simultaneous measurements of concentrations of mRNA transcripts from thousands of genes has generated a number of proposed and actual applications of machine learning methods to infer networks—represented as Boolean or Bayes nets—of regulatory relations among genes. These proposals face several difficulties: (1) the number of measurements of each gene is typically much smaller than the number of genes under study, and the number of genes—or genes at time points in time series representations--effectively defines the number of variables; (2) microarray measurements have a small signal to noise ratio, (3) measurements are not of mRNA concentrations in individual cells, but from aggregates of thousands of cells, and, except when the probability distribution for individual cell expression levels is Gaussian, conditional independence relations that hold for probability distributions in individual units are typically not the same as those that hold in the probability distribution for cell aggregates (Danks and Glymour, 2002; Chu, *et al.*, in press)—experimentally established transcription dependencies in eukaryotic cells appear to be highly non-linear (Davidson, *et al.*, 2001); (4) the Boolean and Bayes net representations depend on acyclic graph representations, and cannot faithfully represent both the probability distributions for equilibrium distributions of feedback systems and the mechanisms that lead to an equilibrium—cyclic graphical representations can do both for equilibria from linear systems, and for non-linear systems under special assumptions, but the only available correct algorithm for obtaining such representations from equilibrium data has never been tested on gene expression data (Richardson, 1996); (5) statistical associations among measured expression levels for different genes may depend on variations in unrecorded regulator genes, or on extra-genetic factors not in the database; (6) summing variable values over many cell units reduces their variance, resulting in low correlations of transcript concentration due to regulatory interaction, which implies the need either for very large samples or very large expression differences to reliably distinguish zero from non-zero correlations; (7) discretization of continuous variables can alter the original conditional independence relations among variables; (8) when there are unrecorded sources of covariation, linear regression techniques overfit in the linear case, positing false connections even without correlated errors (Spirtes, et al., 2001), and feedback can produce statistical dependencies among measured variables similar to the effects of omitted common causes.

There are reports of successes at network inference with machine learning methods applied to both real-world and simulated expression data, but to our knowledge, no published simulation studies generate their data from experimentally established networks and treat measured values as aggregates of many individual cell values.

2. Algorithms. We consider the following algorithms: REVEAL (Liang, *et al.* 1998), BOOL2 (Akutsu, *et al.*, 2000), MRBN (Friedman, *et al.*, 1999), PC (Spirtes, *et al.*2001), CCD (Richardson, 1996), and algorithms described in Spirtes and Meek (1995), Arkin *et al.* (1997),

[1] Research supported by NASA grants NCC2-1295 and NCCI-1227. Our thanks to Peter Spirtes.

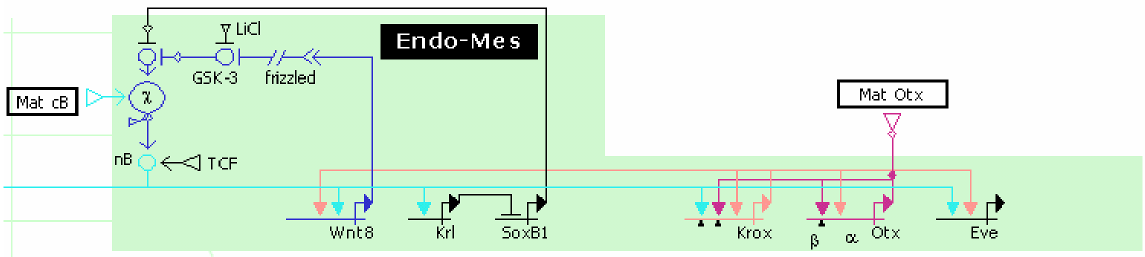

Figure 1: The "maternal and early interactions" portion of the regulatory network of the sea urchin embryo. See Davidson et al, 2001 for details.

D'Haeseleer et al (1999), Weaver *et al.* (1999) and van Someren *et al.* (2000). The version of MRBN we use was implemented by Aaron Darling (see http://mrbn.dyndns.org/). Other downloadable versions did not run, reimplementation was not possible from published accounts, and the authors did not respond to requests for clarification. PC and CCD were obtained from http:/www.phil.cmu.edu/projects/tetrad. The Meek/Spirtes algorithm was run for us by Peter Spirtes from an old implementation not currently publicly available. We implemented the REVEAL and BOOL2 algorithms from published descriptions. The remaining algorithms were obtained from http://genlab.tudelft.nl/info.

These algorithms include procedures that discretize variables to binary or ternary values (REVEAL, BOOL2, MBRN), procedures that treat variables as continuous, procedures that use optimization routines (BOOL2), regression procedures of various kinds (Weaver, Van Someren, Arkin, D'Haeseleer), constraint based searches (PC, CCD), Bayesian scoring searches (MBRN) and hybrid constraint/Bayesian searches (Spirtes/Meek). Clearly these are not all of the algorithms that have been or could be proposed for studying gene regulation. For example, we have not applied the FCI algorithm (Spirtes, *et al.,* 2001*),* nor have we included simulated annealing algorithms (Hartemink, 2001) or heuristic scoring procedures for Bayes nets with time indexed variables. We attempted to include a recent algorithm proposed and applied by Pe'er, et al. (2002), but the authors declined to provide their implementation.

3. Data. Data for this study were of the following kinds: Data generated in ten steps from a time series network modeling regulation in a fragment of the sea urchin genome; data similar to (1) but projected to binary values; data similar to (1) but projected to three values; data from microarray measurements of variations of expression levels over the cell cycle in yeast (Spellman *et al.*, 1998) compared with a recent experimental determination of a substantial fraction of the regulatory network in the same species (Lee, *et al.*, 2002).

3.1 Maternal Sea Urchin Network from NetBuilder

Davidson and his collaborators (Davidson, *et al.*, 2002) have worked for many years to elucidate the genetic network of the sea urchin embryo, resulting in experimental data for a network of some forty genes. Bolouri and his colleagues (Brown, 2002) have developed a simulator, NetBuilder, that implements realistic transfer functions relating gene inpts to their outputs (Appendix A; see http://strc.herts.ac.uk/bio/maria/NetBuilder/Tutorial/netbuilder_tutorial13.htm). One of the most extensive simulations is

Experiment 1: Non-Aggregated, each with 20 runs (samples) of the NetBuilder style simulator (S = 20).

	False Pos	Correct Pos	False Neg	Correct Neg	Total Errors	Mean Error rate
PC05	5.2/1.3	9.2	2.8/.6	3.8	8.0/2.2	0.38
CCD05	4.4/2.7	7.6	4.4/2.3	4.6	8.8/5.5	0.42
Meek/Spirtes	4.2/2.2	10.4	1.6/.5	4.8	5.8/4.2	0.28
Reveal	6.0/.1	8.2	3.8/.2	3.0	9.8/.4	0.47
Bool2	6.1/.6	7.9	4.1/.1	2.9	10.2/.3	0.49
MRBN	3.2/1	3.2	8.8/2.6	5.8	12/3.7	0.57
Arkin	1.4/.02	2.7	9.3./02	7.6	10.7/.01	0.51
Weaver	8.9/.01	11.9	0.1/0.0	0.1	9.0/.01	0.43
D'Haeseleer	5.4/.09	7.3	4.7/.14	3.6	10.1/.11	0.48

Experiment 2: Each experiment consists of 30 aggregated samples each of which is the average of 20 non-aggregated samples (S = 20, R = 30).

	False Pos	Correct Pos	False Neg	Correct Neg	Total Errors	Error Rate
PC05	5.4/1.8	8.9	3.1/2.1	3.6	8.5/2.5	0.40
CCD05	4.9/.54	8.5	3.5/2.5	4.1	8.4/3.2	0.40
Meek	5.7/0.9	9.1	2.9/.8	3.3	8.6/1.2	0.41
Reveal	6.1/0	8.1	3.9/.07	2.9	10/.17	0.48
Bool2	6.3/.06	7.8	4.2/0	2.7	10.5/.1	0.50
MRBN	1.9/1	4.0	8.0/0.4	7.1	9.9/2.1	0.47
Arkin	1.4/.1	2.7	9.3/.05	7.6	10.7/.26	0.51
Weaver	9.0/0	11.9	0.1/0	0.0	9.1/0	0.43
D'Haeseleer	5.3/.07	7.0	5.0/.14	3.7	10.3/.34	0.49

Experiment 3: Each experiment uses 100 aggregated samples each of which is the average of 20 non-aggregated samples (S = 20, R = 100).

	False Pos	Correct Pos	False Neg	Correct Neg	Total Errors	Error Rate
PC05	5.7/1.8	9.3	2.7/1.8	3.3	8.4/3.6	0.40
CCD05	5.4/0.7	8.8	3.2	3.6/2.2	8.6/4	0.41
Meek	5.6/0.9	9.1	2.9/1.9	3.4/1.2	8.5/4.5	0.40
Reveal	6.1/0	8.2	3.8/0	2.9	9.9/0	0.47
Bool2	6.2/0	7.8	4.2/0	2.8	10.4/0	0.50
MRBN	2.2/1.3	3.7	8.3/1.1	6.8	10.5/1.4	0.50
Arkin	1.5/.04	2.7	9.3/.09	7.5	10.8/.2	0.51
Weaver	9.0/0	12.0	0.0/0	0.0	9.0/0	0.43
D'Haeseleer	5.5/.18	7.3	4.7/.17	3.5	10.2/.16	0.49

based on the sea urchin model developed by Davidson *et al.* In order to carry out extensive batch runs in which the parameters of the simulation could be systematically varied, we implemented a Java program with the logical and mathematical functionality described in the NetBuilder documentation, but with none of the visualization or user-interface features of NetBuilder. We tested this program by comparing the output with NetBuilder's results for the "maternal and early interactions" portion of the sea urchin network See Figure 1 for a diagram of the network; note that there are six genes (Wnt8, Krl, SoxB1, Krox, Otx, and Eve). The trace of values for our implementation is very close to the NetBuilder data[2] and, without noise, the two programs reach the same steady state after very few time steps (two or three). We computed the output value for each gene based on its inputs in the manner specified by NetBuilder and then multiplied the output by the value of a random Gaussian variable with mean 1 and variance 0.01. We did not include an additive error. Data were obtained for individual units and aggregated, with repeated runs. This network has several feedback loops, including at least 2 genes that auto-regulate.

To create a *non-aggregated* dataset we ran the Java implementation of the NetBuilder simulation of the maternal and early interactions network some number of times. For each such sample we recorded the simulated expression

[1] The values for components of the network (NetBuilder permits arithmetic and logical functions, among others, in addition to genes) were the same for our batch-mode software as in NetBuilder but in some cases they were shifted in time by one step. In a private communication, Maria Schilstra, the developer of NetBuilder, feels that this is because of a difference in the order of evaluation of the components and is immaterial.

Experiment 4: Non-aggregated. Sample size is 100 and transfer functions for the genes were replaced by linear functions.

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	1.2/1.1	6.9	5.1/1.4	7.8	6.3/2.7	0.30
CCD05	1.4/0.7	7.3	4.7/0.7	7.6	6.1/2.3	0.29
Meek	1.8/1.5	7.5	4.5/.95	7.2	6.3/3.1	0.30
Reveal	5.2/0	6.8	5.2/0	3.8	10.4/0	0.50
Bool2	5.3/0	7.6	4.4/0	2.7	9.7/0	0.46
MRBN	1.9/.5	3.4	8.6/1.4	7.1	10.5/1.2	0.50
Arkin	1.2/0	3.2	8.8/.03	7.8	10.0/.07	0.48
Weaver	4.5/.17	3.8	8.2/.1	4.5	12.7/.02	0.60
D'Haeseleer	8.0/.03	11.3	0.7/.02	1.0	8.7/.03	0.41

Experiment 5: Similar to experiment 5 but an aggregated dataset is used (S = 20, R = 100).

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	1.2/.84	7.4	4.6/.84	7.8	5.8/3.7	0.28
CCD05	1.7/0.9	7.5	4.5/1.6	7.3	6.2/3	0.30
Meek	2.0/0.9	7.5	4.5/0.7	7.0	6.5/1.6	0.31
Reveal	5.3/0	6.9	5.1/0	3.7	10.4/0	0.50
Bool2	5.3/0	7.6	4.0/0	3.7	9.7/0	0.46
MRBN	2.1/0.77	3.0	9.0/1.6	6.9	11.1/1.9	0.53
Arkin	1.2/.07	3.2	8.8/.03	7.8	10.0/.07	0.48
Weaver	5.6/.03	4.7	7.3/.02	3.4	12.9/.01	0.61
D'Haeseleer	7.8/.03	11.2	0.8/.03	1.2	8.6/.08	0.41

level for each of the six genes (Wnt8, Krl, etc.) at each of up to 10 time steps; we call the values recorded for each such simulation a *non-aggregated sample*. The algorithms assume a data matrix consisting of one column per variable and one row per sample. In this case each simulation corresponds to a row and each gene-time step corresponds to a column (variable). To construct an *aggregated* dataset we compute the mean for each column of a non-aggregated dataset and that number becomes the entry in the corresponding column of the aggregated dataset; the number of rows is determined by the number of non-aggregated datasets generated and averaged. We call the vector of values stored in such a row an *aggregated sample*. In the description of results below, let S = the number of simulations used to construct a non-aggregated dataset and let R = the number of non-aggregated datasets used to construct each aggregated sample in an aggregated dataset. In all of our experiments the sample sizes are comparatively small—reflecting the reality of microarray studies—and in most cases, the distributions are non-Gaussian, and the dependencies are non-linear. We approximated mean-zero normality by taking logs of all values in the data matrices (for both non-aggregated and aggregated datasets) and then subtracting the median of each column from all the values in that column.

By projecting based on the median value of each variable, we binarized the same data for tests of the Reveal and Bool2 algorithms. The MRBN algorithm implementation automatically projects real values to one of three values. The PC and CCD algorithms require multiple samples, each consisting of an entire time series. The binary algorithms require as input a dataset consisting of binary values for each of a set of genes at each of a number of time steps. Hence the same time series can be used for comparisons. Data are available at http://www.phil.cmu.edu/projects/genegroup.

3.2 Yeast Data Spellman, *et al.* (1998) report data on four experiments in which mRNA expression levels were measured in the course of the cell cycle with cells synchronized in different ways. Friedman, *et al.*, applied the MRBN algorithm to this data to obtain conjectured

Experiment 6. Results of testing the algorithms on the S. Cerevisiae data

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	5	3	26	32	31	0.47
CCD05	5	3	26	32	31	0.47
Reveal	16	13	16	21	32	0.48
Bool2	2	1	28	35	30	0.45
MRBN	18	6	23	19	41	0.62
Arkin	3	2	27	34	30	0.45
Weaver	12	18	11	25	23	0.35
D'Haeseleer	2	1	28	35	30	0.45

regulatory relations among the genes. Comparison experimental data are from Lee, *et al.* (2002), who applied immunoprecipitation techniques to experimentally estimate genes directly regulated by each of more than 100 known yeast regulators.

4. Results

4.1 Simulated Data In all cases, directions of edges are ignored and results are reported only for adjacencies. An adjacency is judged present between two genes in an experiment if and only if it is present between those genes for *any* two times. Many other counting procedures are possible within each experiment (e.g., majority rule; restriction to sequential time steps) that would reduce false positives and increase false negatives. There are 12 edges in the true graph for the maternal and early interactions portion of the sea urchin embryo network. There are 21 pairs of the six genes since a gene can auto-regulate. PC, CCD and Spirtes/Meek take a significance level as input--we give results for .05; results for other significance levels up to 0.3 are similar, with lower significance levels slightly better in most experiments. Someren and D'Haeseleer results were essentially identical and we show only the latter. Each of the numerical results shown below is the average over 10 replications of the experiment. Variances over 10 replications are given on the right hand side of the backslash. Note that random assignment of edges for pairs of genes would result in 10.5 expected errors and an error rate of 0.5. Simply saying "yes" to each possible adjacency would result in an error rate of 0.43.

4.2 Testing the Algorithms with Actual Microarray Data. The Spellman *et al.* (1998) studies of *Saccharomyces cerevisiae* report four experiments involving different numbers of time measures, at different time delays, for cell populations with very different metabolic rates. See http://staffa.wi.mit.edu/cgi-bin/young_public/navframe.cgi?s=17&f=downloaddata. Lee *et al.* (2002) used immunoprecipitation techniques to identify genes directly regulated by more than 100 regulators; their study includes 11 genes implicated in the cell cycle. Friedman, et al. analyzed the Spellman data but their results are not searchable and so we cannot make direct comparisons with their results. Because the four experiments used samples in different metabolic conditions, it is not sensible to use the Spellman data as repeated samples of the same time series, and we therefore simply concatenated the data so they appeared to be from one experiment; this introduces 3 false breaks in time series. Our analysis was restricted to the 11 cell cycle genes that appear in the diagram published by Lee *et al.* We applied the PC and CCD algorithms to the data as though it were equilibrium data, using the 11 genes as variables, implicitly violating i.i.d. sampling assumptions of these algorithms. There are 66 possible regulatory relationships, ignoring direction of regulation, including autoregulation.

5. Discussion. No confirmation of an algorithm for obtaining regulatory structure from expression data can be rationally justified by results with simulated data unless the data generating model is non-linear, with feedback, and the variable values are aggregated over simulated individual cells. Selective comparisons with independently wet-laboratory results do not suffice either. Among the tests reported here, experiments 2 and 3 provide the most realistic simulations, and the best tests, of the algorithms considered. Even so, in several respects the inference problems posed by our simulated experiments 2 and 3 are easier than with real data: the correct time sampling frequency is known; all replications are with the same simulated metabolism and the same time sampling; there are no missing values; the variables are aggregated over only 20 units (the larger the number of units of aggregation, the smaller the correlation among the aggregated variables).

The implementations of Reveal and Bool2 limited them to three regulators per gene. For those yeast genes actually with three or fewer regulators in the Lee, *et al.*, model, the results for these algorithms were almost always at chance, indicating the restriction to three regulators was inessential to their performance. One run of Bool2 was attempted for the yeast data, which allowed up to four regulators; the program ran for about 8 hours (over 200 times as long as the three regulator case) and returned the null model (no estimated regulatory relationships). On the simulation data, the REVEAL and BOOL2, Weaver and D'Haeseleer algorithms proved useless; the remaining algorithms proved to be of some slight utility.

Considering both positive and negative errors, as deployed here the REVEAL, BOOL2, MRBN and Arkin algorithms performed essentially at chance in all experiments: they are equivalent to flipping a coin to decide adjacencies. For non-linear simulated data, Weaver's algorithm is equivalent to saying "yes" to every adjacency; for linear data one would do better to use an inverted Weaver algorithm: say "yes" when it says "no." The D'Haeseleer algorithm is better than chance only for linear data, where it approximates saying "yes" in almost all cases. The PC and CCD are a little better than chance in all experiments and considerably better than chance with linear data. The Meek/Spirtes hybrid algorithm nearly dominates for total error rates on simulated data, and shows the theoretically expected increase in false positives with aggregated non-linear data. None of the algorithms improved with sample size increases up to 100. If we consider only the ratio of correctly predicted positives to predicted positives, and the most realistic simulations (Experiments 2 and 3), the PC, CCD, Meek/Spirtes, MRBN and Arkin algorithms all do slightly better than merely saying "yes" in all cases—varying in experiment 4 from .62 to .64 as against the constant "yes" ratio of .57 . The MRBN and Arkin algorithms purchase the slight improvement at the cost of missing most of the true positives.

Regrettably, these results tend to confirm the theoretical arguments against the reliability of machine learning algorithms for estimating gene regulation networks from microarray measurements of expression levels. The Meek/Spirtes algorithm, which does notably better than chance or constant "yes" responses on non-linear, non-aggregated data in experiment 1, falls to the constant "yes" error rate when the variables are aggregated. The linear regression procedures overfit with data from a linear feedback system. It is conceivable that other counting principles would decrease false positives for PC, CCD and Spirtes/Meek and perhaps other algorithms in experiments 2 and 3, rendering them more useful, but we have not explored the possibilities. It would be preferable to have each algorithm run by its authors on common, well-specified, realistic simulation data from structures kept secret from those executing the algorithms and with explicit, pre-specified principle or principles for counting errors, but such cooperative tests seem unlikely while relevant authors make neither their algorithms nor implementations publicly available.

Experimental techniques that take advantage of immunoprecipitation, tagging of binding sites and regulatory proteins, binding site sequence homologies, and evolutionary preservation of regulatory mechanisms, are proving more fruitful. We may hope that machine learning techniques that are biased by such extra information may prove useful. The Meek/Spirtes algorithm, for example, can be easily and flexibly biased by prior information about the likely presence or absence of particular regulatory connections.

6. References

T. Akutsu, S. Miyano, and S. Kuhara, (2000). Algorithms for Inferring Qualitative Models of Biological Networks, *Pacific Symposium on Biocomputing* 5:290-301.

A. Arkin, P. Shen, and J. Ross, (1997). A Test Case of Correlation Metric Construction of a Reaction Pathway from Measurements, *Science* 277:1275-1279.

C. T. Brown, A. G. Rust, P. J. C. Clarke, Z. Pan, M. J. Schilstra, T. D. Buysscher, G. Griffin, B. J. Wold, R. A. Cameron, E. H. Davidson and H. Bolouri, (2002). New Computational Approaches for Analysis of Cis-regulatory Networks, *Developmental Biology* 246, 86-102.

T. Chu, C. Glymour, R. Scheines and P. Spirtes, (2002). A Note on a Statistical Problem for Inference to Gene Regulation from Microarray Data, *Bioinformatics*, in press.

D. Danks and C. Glymour, (2002). Linearity Properties of Bayes Nets with Binary Variables, *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, Seattle.

E. Davidson, J. Rast, P. Oliveri, A. Ransick, C. Calestani, C.Yuh, T. Minokawa, G. Amore, V. Hinman, C. Arenas-Mena, O.Otim, C. Brown, C. Livi, P. Lee, R. Revilla, A. Rust, Z. Pan, M. Schilstra, P. Clarke, M. Arnone, L. Rowen, R. Cameron, D. McClay, L. Hood, and H. Bolouri, (2002). A Genomic Regulatory Network for Development, *Science* 295: 1669-1678.

P. D'haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, (1999). Linear Modeling of mRNA Expression Levels During CNS Development and Injury, *Pacific Symposium on Biocomputing* '99, pp. 41-52.

N. Friedman, D. Pe'er, I. Nachman, (1999). Learning Bayesian Network Structure from Massive Datasets: The 'Sparse Candidate' Algorithm, UAI 1999.

A. Hartemink, (2001). Principled Search for Gene Regulation. Ph.D Thesis, Harvard University.

T. Lee, N. Rinaldi, F. Robert, D. Odom, Z. Bar-Joseph, G. Gerber, N. Hannett, C. Harbison, C. Thompson, I. Simon, J. Zeitlinger, E. Jennings, H. Murray, D. B. Gordon, B. Ren, J. Wyrick, J. Tagne, T. Volkert, E. Fraenkel, D. Gifford, R. Young, (2002). Transcriptional Regulatory Networks in *Saccharomyces cerevisiae*, *Science* 298:799-804.

S.Liang, S.Fuhrman, R. Somogyi (1998). Reveal, a General Reverse Engineering Algorithm for Inference of Genetic Network Architectures, *Pacific Symposium on Biocomputing* 3:18-29.

D. Pe'er, A. Regev, A. Tanay (2002). MinReg: Inferring an Active Regulator Set, In *Proc. Tenth International Conference on Intelligent Systems for Molecular Biology (ISMB).*

T. Richardson (1996). A Discovery Algorithm for Directed Cyclic Graphs. UAI 1996: 454-461.

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

P. Spirtes, C. Glymour, R. Scheines, (2001). *Causation, Prediction and Search*, 2nd Ed., MIT Press.

P. Spirtes and C. Meek, (1995). Learning Bayesian Networks with Discrete Variables from Data, in *Proceedings of The First International Conference on Knowledge Discovery and Data Mining*, ed. by Usama M. Fayyad and Ramasamy Uthurusamy, AAI Press, pp. 294-299, 1995.

E.P. van Someren, L.F.A. Wessels and M.J.T. Reinders, (2000). Linear Modeling of Genetic Networks from Experimental Data, *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB00*), 355-366, La Jolla, California.

D. C. Weaver, C.T. Workman, G.D. Stormo, (1999). Modeling Regulatory Networks with Weight Matrices, *Pacific Symposium on Biocomputing* 4:112-123.

7. Appendix A: Transfer Functions in the NetBuilder Model of the Sea Urchin Embryo. The six genes considered here are: Wnt8, Krl, SoxB1, Krox, Otx and Eve. All of these are 'All And' genes except Krox and Otx which are 'All Or' genes. For all genes their inputs are transformed as x/(x+1) for a positive link (represented by an arrowhead in Figure 1) or as 1 – x/(x+1) for a negated link (represented by a segment perpendicular to the link). Note that the transformed inputs lie between 0 and 1. An And function multiplies the inputs and an Or function is the continuous generalization of a Boolean "or" so that $y_{or} = x_1 + x_2*(1 - x_1)$. When there are more than two inputs the form of the Or function is $y_{or} = x_1 + x_2*(1 - x_1) + x3(1 - (x1 + x2(1 - x1))) + \ldots$ etc. All components of a NetBuilder model may also have a factor F and a power P associated with them. Then $y = F*x^P$ where x is the input and y is the output. For the genes in Figure 1, P is 1.0 in every case and F is 1.0 except for genes Eve, Otx and Krox which have F = 100.0. Other components of the network are: χ is an And function with P = 1.0 and F = 1.0. Its inputs are Mat cB and pre-χ. post-χ is an Or function which has inputs χ and χ–switch (an exogenous input with constant value 1.0) It is not labeled in Figure 1. Mat cB, LiCl, TCF and Mat Otx are exogenous inputs with constant values 1.0, 0.0 and 1.0 and 1.0 respectively. nBmod is function whose effect is to multiply its input by 10.0. It is not labeled in Figure 1. GSK-3 is an And function whose inputs are Wnt8 and LiCl. pre-χ is an And function whose inputs are GSK-3mod and SoxB1mod. GSK-3mod is a single-valued function that multiplies its input (GSK-3) by 10.0. It is not labeled in Figure 1. SoxB1mod is a single-valued function that multiplies its input (SoxB1) by 10.0. It is not labeled in Figure 1. Frizzled is not in the model.