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

Frank C. Wimberly[1], Thomas Heiman[2], Joseph Ramsey[3] and Clark Glymour[4],

Abstract

After reviewing theoretical reasons for doubting that machine learning methods can accurately infer gene regulatory networks from microarray data, 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. Our results agree with the theoretical arguments: most algorithms are at chance for determining the existence of a regulatory connection between gene pairs, and the algorithms that perform better than chance are nonetheless so error-prone as to be of little practical use in these applications.

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. Proposed networks encode both hypothetical causal relations and conditional independence relations among gene expressions. The measured, continuous-valued, expression levels are typically either projected to binary or other discrete values, or transformed to approximate marginal Gaussian distributions. In the latter case, the conditional independence relations are estimated from the second moments—the variance/covariance matrices—of the estimated joint distributions. Because the concentration measurements are essentially additive functions of mRNA concentrations in thousands of approximately independent values for individual cells, the joint distribution of the measured quantities should, by the Central Limit Theorem, be approximately Gaussian.

These proposals face several difficulties: (1) the number of measurements of each gene is invariably much, much smaller than the number of genes under study, and the number of genes effectively defines the number of variables; (2) microarray measurements have a small signal to noise ratio, and at least one standard type (cDNA arrays) appears to have systematic errors resulting in major anomalies in the variances; (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 linear and Gaussian, conditional independence relations that hold for probability distributions in individual units are not the same as

[1] Institute for Human and Machine Cognition, University of West Florida, wimberly3@earthlink.net.
[2] School of Computer Science, George Mason University, theiman@scs.gmu.edu.
[3] Department of Philosophy, Carnegie Mellon Univeristy, jdramsey@andrew.cmu.edu.
[4] Department of Philosophy, Carnegie Mellon University and Institute for Human and Machine Cognition, University of West Florida, cg09@andrew.cmu.edu.

those that hold in the probability distribution for cell aggregates (Danks and Glymour, 2001; 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 represent equilibrium distributions of feedback systems, but eukaryotic networks have feedback; (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) there is considerable evidence that regulatory relationships may be non-linear; and, (7) 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 for very large samples to reliably distinguish zero from non-zero correlations.

Despite these difficulties, which are insufficiently noted in the relevant machine learning literature, there are reports of successes with machine learning methods applied to both real-world and simulated data. But few of these reports compare regulatory networks obtained by machine learning methods with real data to networks independently established by experimental methods, and, 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, and none of the studies use the only available algorithm for equilibrium feedback systems. In the work reported here, we do all of the above.

2. Algorithms We consider two published algorithms for Boolean networks that assume all variables are binary: Reveal (Liang, *et al.* 1998) and Bool2 (Akutsu, *et al.*, 2000). We also consider an algorithm that builds Bayes nets from multivalued variables, the sparse network Bayesian algorithm MRBN (Friedman, *et al.*, 1999). We consider seven algorithms for continuous, Gaussian variables: PC (Spirtes, *et al.*) and CCD (Richardson, 1996) and algorithms described in Arkin *et al.* (1997), D'Haeseleer et al (1999), Weaver *et al.* (1999) and van Someren *et al.* (2000), as well as a genetic algorithm.

The Reveal and Bool2 algorithms were among the earliest proposed specifically for unraveling gene networks. The MRBN algorithm is one of the most discussed procedures in the computer science literature on gene regulation networks. The algorithm has been applied to yeast data with both finite-valued projections and also with the presumption of a Gaussian distribution, yielding very different results. The algorithm is, however, not clearly documented (and authors did not reply to our requests for further documentation); we have therefore used an implementation by Aaron Darling (see http://mrbn.dyndns.org/). The PC algorithm (Spirtes, *et al.*, 1993, 2001) is the first Bayes net search algorithm feasible for large, sparse networks. It is implemented for both discrete valued variables and for Gaussian variables; we have used only the latter version here. The CCD algorithm (Richardson, 1996) is to our knowledge the only correct algorithm ever published for feedback systems. We have used it with the assumption of Gaussian variables. Implementations of these algorithms are available at http://phil.cmu.edu/projects/tetrad. Implementations of the various algorithms described in Arkin *et al.* (1997), D'Haeseleer et al (1999), Weaver *et al.* (1999) and van Someren *et al.* (2000) can be found at the GENLAB web site http://www.genlab.tudelft.nl/ . We know of no proofs of correctness for any of the last four algorithms.

Clearly these are not all of the algorithms that have been or could be proposed for studying gene regulation. We have not applied the FCI algorithm (Spirtes, *et al.*), which is correct for systems that may have latent variables, because it is very slow; our simulated data do not include latent variables. Nor have we included simulated annealing algorithms (Hartemink, 2001).

3. Data Data in this study were of the following kinds:

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

In all simulated cases algorithms were studied with both data from individual units (simulated cells) and data in which values of each variable were summed over 30 or 100 units. We give details only of the aggregated data.

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 inputs to their outputs (see Appendix A). One of the most extensive simulations is 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 are 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. The choice of the "maternal and early interactions" subnetwork was indicated by the manageable size of the network and the fact that the only exogenous inputs were a small number of maternal factors. See Figure 1 for a diagram of the network; note that there are six genes (Wnt8, Krl, SoxB1, Krox, Otx, and Eve) and several feedback loops. The trace of values for our implementation is very close to the NetBuilder data[5] and, without noise, the two programs reach the same steady state after very few time steps (two or three). In order to introduce stochasticity into the simulations, 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

[5] 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.

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. For a discussion of the nonlinear transfer functions associated with the genes in the NetBuilder simulator see http://strc.herts.ac.uk/bio/maria/NetBuilder/Tutorial/netbuilder_tutorial13.htm

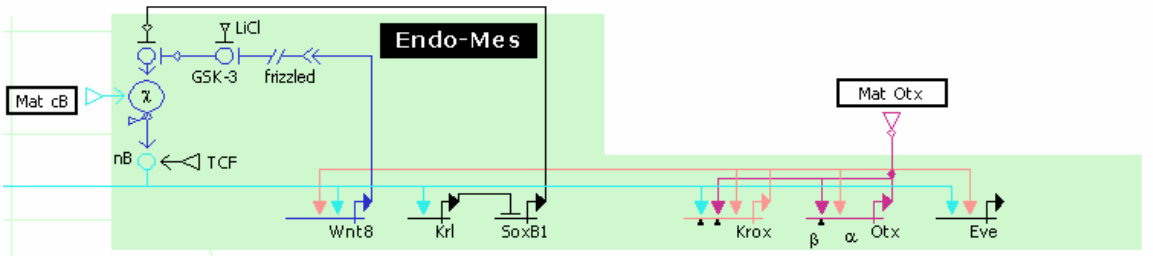

Figure 1: The "maternal and early interactions" portion of the regulatory network of the sea urchin embryo. See Davidson et al, 2001 for details.

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—typically 20 or 100. For each such sample we recorded the simulated expression 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*. To redundantly clarify: each row of an aggregated dataset is an aggregated sample and is the mean of the rows of a non-aggregated dataset. 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. Hence an aggregated dataset will be characterized by two integers, S and R.

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 values for a set of gene/time-step pairs. 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. For comparisons, in all cases we compute the average number of errors

(false positives, false negatives and their total) across the set of estimated models. In the results section we report the outcome of a number of experiments using NetBuilder style simulations. In all cases, directions of edges are ignored and errors are reported only for adjacencies. All of the simulated data are available at http://www.phil.cmu.edu/projects/genegroup.

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

Some caveats are appropriate. Although the network structure of a fragment of the sea urchin gene has been established experimentally, the definite mathematical form of the dependencies of expression level concentrations on regulator concentrations in the sea urchin has to our knowledge only been published for a single gene, Endo16. In our simulations, we have used the nonlinear dependencies described on the NetBuilder Tutorial web page referenced above. The comparison of machine learning results applied to the Brown lab study of the yeast cell cycle with the regulatory network found in Lee, *et al.* cannot expect a perfect match, because the cells in the two studies were not maintained or measured in the same conditions. One would nonetheless expect a significant fraction of regulatory relations to be invariant to sustain "housekeeping" physiological processes.

4. Results

4.1 Comparison of Continuous and Binary Algorithms for Simulated Data We performed several experiments using the NetBuilder style simulator as the source of data. These included both non-aggregated and aggregated datasets with S = 20 and with R = 30 and 100. We binarized these datasets and applied the Reveal and Bool2 algorithms, and, separately, we applied the MRBN algorithm. With the MRBN algorithm the S samples were concatenated to create the input; this resulted in a longer time series of expression vectors. The PC and CCD algorithms require a user specified significance level; we have tested the algorithms on a range of values of the parameters from .05 to .3. We do not tabulate results of the genetic algorithm, which tended to produce no edges in any tests.

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. In experiments with simulated data we show only the results with aggregation. Results with non-aggregated data are similar. Each value in the following tables represents the average number of false positives, false negatives and total errors across 10 replications of the experiment for each algorithm.

Experiment 1: Aggregated data where each of 30 aggregated samples is the average of 20 non-aggregated samples (S = 20, R = 30). The values in the table represent the mean across 10 replications of the experiment.

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	5.4	8.9	3.1	3.6	8.5	0.40
PC10	6.6	10.5	1.5	2.4	8.1	0.39
PC15	7.6	11.3	0.7	1.4	8.3	0.40
PC20	7.5	11.6	0.4	1.5	7.9	0.38
PC30	8.1	12.0	0.0	0.9	8.1	0.39
CCD05	4.9	8.5	3.5	4.1	8.4	0.40
CCD10	6.9	10.3	1.7	2.1	8.6	0.41
CCD15	7.8	11.4	0.6	1.2	8.4	0.40
CCD20	8.0	11.6	0.4	1.0	8.4	0.40
CCD30	8.1	11.9	0.1	0.9	8.2	0.39
Reveal	6.1	8.1	3.9	2.9	10.0	0.48
Bool2	6.3	7.8	4.2	2.7	10.5	0.50
MRBN	1.9	4.0	8.0	7.1	9.9	0.47
Arkin	1.4	2.7	9.3	7.6	10.7	0.51
Weaver	9.0	11.9	0.1	0.0	9.1	0.43
Someren	5.3	7.0	5.0	3.7	10.3	0.49
D'Haeseleer	5.3	7.0	5.0	3.7	10.3	0.49

Experiment 2: Identical to experiment 3 except that there are 100 aggregated samples each of which is the average of 20 non-aggregated samples (S = 20, R = 100). Again there are 10 replications.

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	5.7	9.3	2.7	3.3	8.4	0.40
PC10	7.1	11.0	1.0	1.9	8.1	0.39
PC15	7.8	11.6	0.4	1.2	8.2	0.39
PC20	8.0	11.8	0.2	1.0	8.2	0.39
PC30	8.7	12.0	0.0	0.3	8.7	0.41
CCD05	5.4	8.8	3.2	3.6	8.6	0.41
CCD10	7.5	11.0	1.0	1.5	8.5	0.40
CCD15	8.3	11.7	0.3	0.7	8.6	0.41
CCD20	8.4	11.9	0.1	0.6	8.5	0.40
CCD30	8.7	11.9	0.1	0.3	8.8	0.42
Reveal	6.1	8.2	3.8	2.9	9.9	0.47
Bool2	6.2	7.8	4.2	2.8	10.4	0.50
MRBN	2.2	3.7	8.3	6.8	10.5	0.50
Arkin	1.5	2.7	9.3	7.5	10.8	0.51
Weaver	9.0	12.0	0.0	0.0	9.0	0.43
Someren	5.5	7.3	4.7	3.5	10.2	0.49
D'Haeseleer	5.5	7.3	4.7	3.5	10.2	0.49

Experiment 3: This is similar to experiment 2, except that linear transfer functions are used (S = 20, R = 100).

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	1.2	7.4	4.6	7.8	5.8	0.28
PC10	2.4	8.4	3.6	6.6	6.0	0.29
PC15	4.5	8.9	3.1	4.5	7.6	0.36
PC20	5.8	9.5	2.5	3.2	8.3	0.40
PC30	7.0	10.3	1.7	2.0	8.7	0.41
CCD05	1.7	7.5	4.5	7.3	6.2	0.30
CCD10	3.2	8.5	3.5	5.8	6.7	0.32
CCD15	5.1	9.3	2.7	3.9	7.8	0.37
CCD20	5.7	10.3	1.7	3.3	7.4	0.35
CCD30	7.1	11.2	0.8	1.9	7.9	0.38
Reveal	5.3	6.9	5.1	3.7	10.4	0.50
Bool2	5.3	7.6	4.4	3.7	9.7	0.46
MRBN	2.1	3.0	9.0	6.9	11.1	0.53
Arkin	1.2	3.2	8.8	7.8	10.0	0.48
Weaver	5.6	4.7	7.3	3.4	12.9	0.61
Someren	7.8	11.2	0.8	1.2	8.6	0.41
D'Haeseleer	7.8	11.2	0.8	1.2	8.6	0.41

4.2 Testing the Algorithms with Actual Microarray Data. The Spellman *et al.* (1998) and Lee *et al.* (2002) studies provide, respectively, microarray measurements of a time series of synchronized cells and an elucidation of the network of regulators for *Saccharomyces cerevisiae*. This offers us the opportunity to test the algorithms on actual datasets. The microarray data available at the website supporting the Lee *et al.* paper, http://staffa.wi.mit.edu/cgi-bin/young_public/navframe.cgi?s=17&f=downloaddata, represents results of five experiments. To increase sample size, we concatenated the data from the experiments so they appeared to be from one experiment. We applied the PC, CCD, Reveal, Bool2 and MRBN algorithms to this data. The analysis was restricted to the 11 genes that appear in the diagram published by Lee *et al.* on their website http://staffa.wi.mit.edu/cgi-bin/young_public/navframe.cgi?s=17&f=structures
Note that there are 66 possible regulatory relationships, ignoring direction of regulation, including autoregulation. The error rates in the results tables were estimated by dividing the sum of the false negatives and false positives by 66.

The results appear below:

	False Pos	Correct Pos	False Neg	Correct Neg	Total errors	Error rate
PC05	5	3	26	32	31	0.47
PC10	5	3	26	32	31	0.47
PC15	6	3	26	31	32	0.48
PC20	7	3	26	30	33	0.50
PC30	9	4	25	28	34	0.52
CCD05	5	3	26	32	31	0.47
CCD10	6	3	26	31	32	0.48
CCD15	7	3	26	30	33	0.50
CCD20	7	3	26	30	33	0.50
CCD30	9	4	25	28	34	0.52
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
Someren	2	1	28	35	30	0.45
D'Haeseleer	2	1	28	35	30	0.45

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 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).

5. Discussion. Reveal and Bool2 perform at chance level in every experiment: they are equivalent to flipping a coin. MRBN detects very few edges, so few that it, too, is equivalent in most experiments to flipping a coin. The Arkin, D'Haseleer and Sorensen algorithms are at chance in all of the simulation studies. The Weaver algorithm does slightly better than chance in two of the simulation studies, worse than chance with linear transfer functions, and better than chance with the yeast data. With low significance levels, the linear algorithms, PC and CCD, do better than chance in the simulation tests, but not much better, and they are at chance on the yeast test. In confirmation that part of the difficulty for all algorithms is the non-linearity of the regulation dependencies, the PC and CCD algorithms considerably improve with linear transfer functions. The similarity of performances on the aggregated and non-aggregated data is due to the fact that the algorithms perform very poorly on the disaggregated data: there is very little to make worse by aggregation. The variation in sample size between 30 and 100 cases makes no difference in accuracy, suggesting either that sample size is not an issue, or that much larger samples would be required for accurate results. 100 repetitions is already an unrealistically large number of repetitions for microarray studies.

These results confirm the theoretical arguments against the reliability of machine learning algorithms for estimating gene regulation networks from microarray

measurements of expression levels. Other approaches, taking advantage of immunoprecipitation, tagging of binding sites and regulatory proteins, binding site sequence homologies, and evolutionary preservation of regulatory mechanisms, are proving more fruitful.

6. References

Tatsuya Akutsu, Satoru Miyano, and Satoru 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.

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

Eric H. Davidson, Jonathan P. Rast, Paola Oliveri, Andrew Ransick, Cristina Calestani, Chiou-Hwa Yuh, Takuya Minokawa, Gabriele Amore, Veronica Hinman, César Arenas-Mena, Ochan Otim, C. Titus Brown, Carolina B. Livi, Pei Yun Lee, Roger Revilla, Alistair G. Rust, Zheng jun Pan, Maria J. Schilstra, Peter J. C. Clarke, Maria I. Arnone, Lee Rowen, R. Andrew Cameron, David R. McClay, Leroy Hood, and Hamid 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, 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.

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

Thomas Richardson: 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, (2000), Prediction Causation and Search, 2nd Ed., MIT Press.
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), p. 355-366, La Jolla, California, August.
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. There are six genes in the maternal and early interactions region of the regulatory network of the sea urchin embryo. They are: Wnt8, Krl, SoxB1, Krox, Otx and Eve. All of these are 'All And' genes except Krox and Otx which are 'All Or' genes. These are genes in which all inputs are combined using an And or Or function. 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 maternal and early interactions network are listed below:

- 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.

For much more detail on NetBuilder including a tutorial, theoretical discussions, and the endomesoderm (sea urchin) model see the NetBuilder website at http://strc.herts.ac.uk/bio/maria/NetBuilder/.