Pacific Symposium on Biocomputing 4:112-123 (1999)

MODELING REGULATORY NETWORKS WITH WEIGHT MATRICES

D.C. WEAVER
Genomica Corporation, 4001 Discovery Rd., Suite 130
Boulder, CO 80303 (weaver@genomica.com)

C.T. WORKMAN, G.D. STORMO
University of Colorado, Molecular Cellular and Developmental Biology
Boulder, CO 80309 (*workmanc@colorado.edu*, *stormo@colorado.edu*)

Abstract

Systematic gene expression analyses provide comprehensive information about the transcriptional response to different environmental and developmental conditions. With enough gene expression data points, computational biologists may eventually generate predictive computer models of transcription regulation. Such models will require computational methodologies consistent with the behavior of known biological systems that remain tractable. We represent regulatory relationships between genes as linear coefficients or weights, with the "net" regulation influence on a gene's expression being the mathematical summation of the independent regulatory inputs. Test regulatory networks generated with this approach display stable and cyclically stable gene expression levels, consistent with known biological systems. We include variables to model the effect of environmental conditions on transcription regulation and observed various alterations in gene expression patterns in response to environmental input. Finally, we use a derivation of this model system to predict the regulatory network from simulated input/output data sets and find that it accurately predicts all components of the model, even with noisy expression data.

1 Introduction

The information derived from genome sequencing projects allow for systemic analyses of gene expression. As these expression analysis technologies mature, biologists will be presented with accumulated data sets detailing the transcriptional response of a cell, tissue, or organism to many environmental, genetic, and developmental stimuli. In addition to elucidating the cellular response to such stimuli, these experimental results provide an opportunity to understand the regulatory pathways that underlie the observed gene expression patterns. While our ability to predict such regulatory pathways will remain rudimentary with limited data, as more data points are collected, we will be able to define ever more accurate predictions of the transcriptional regulatory apparati.

Transcriptional regulation is conferred through the combinatorial action of gene products on sequence elements proximal to each gene's transcriptional start site. These "transcription factors" bind directly to DNA and influence gene expression by altering the binding or activity of the basal transcription machinery. Transcription factor activity is controlled in turn by other gene products via posttranslational mechanisms[1]. Thus, one can argue that the transcription of any gene is the result of integrating the cell's biochemical state rather than the action of any single gene product. Once an expressed gene is translated into a functional gene product, it affects the state of the cell and may directly or indirectly influence its own expression or the expression of other genes. In this way, the expression state of the cell is regulated; one set of expressed genes (*i.e.* one expression state) regulates the transcription of the cell's genes, leading to a new state, and so on.

Most previous attempts to model transcriptional regulatory networks simplify a gene's expression as being either completely on or completely off[2,3,4,5]. The response of a model gene to some set of expressed genes is dictated by a Boolean rules table (with rules like: "if gene A is expressed AND NOT (genes B OR C), then this gene is expressed). As the system progresses from one state (or timepoint) to the next, the input pattern of expressed genes is cross-referenced with the rules table to determine if the genes they control will be expressed at the next state or time step. Boolean networks converge to terminal states via a series of state transitions, where these different terminal states are analogous to terminal differentiation states in biology[4]. If the terminal condition of a regulatory network is a single unchanging state then it is termed a "point attractor" while if it is a series of states it is called a "dynamic attractor" or "limit cycle"[2].

While being good starting points to gain understanding about the behavior of large dynamic regulatory networks, these "Random Boolean Networks" depend on simplifying assumptions about biological systems. For example, by treating gene expression as either completely on or off, these systems ignore those genes that have different biological regulatory effects at expression levels intermediate between their basal and their maximal expression levels (*lin-14*[6], *lin-3*[7], and *bicoid*[8], for example). Furthermore, these networks cannot address those regulatory genes that influence the transcription of various genes to differing degrees. Finally, many of these regulatory networks are designed such that all genes have a fixed maximum number of regulatory inputs. In biology, some genes are known to have many regulatory inputs, while others are not known to have more than a few. While this may reflect our limited knowledge of the complexity of gene regulation, it seems likely that there will be variance in the number of regulatory inputs to many genes.

Connectionist models for gene regulation in the form of recurrent Hopfield[9] networks have been proposed by Mjolsness *et.al.*[10] and Reinitz and Sharp[11]. Thomas *et.al.*[12] describe regulatory networks as directed graphs or matrices of interactions without restrictions on connectivity. The continuous time networks of (10, 11) model interphase expression of a cell based on interaction weights that are free to take positive and negative real values. Networks of this type can be trained for the goal of function approximation with supervised training (data fitting) provided there are plenty of data points. Unfortunately, current sampling times of expression data are so large that continuous time models could only be based on theoretical data. This is the motivation for the discrete time model proposed here.

We describe herein an algorithm, TReMM (Transcription Regulation Modeled with Matrices), to modeling gene regulatory pathways with a linear weight matrix. Each gene can be expressed at any level from complete repression to maximal expression. Furthermore, the regulatory interactions between genes are allowed to take on any value along a continuum from highly activating to highly repressing as in the models of (10, 11). In addition, we show that this modeling system allows for the facile inclusion of environmental or state-specific variables and allows for the reverse engineering of regulatory networks with only 2-3 more data points than there are genes in the system. Finally, these models use and generate simulated data sets with the same units for expression that are coming out of expression studies. Thus, this model system may lend itself readily to application on real biological data, once enough expression data are available.

2. Methods

2.1 Conceptual definition of the TReMM modeling methodology

For computational tractability, we model transcription regulation as discrete state transitions, such that the expression levels of all genes are updated simultaneously. This assumption is convenient because expression data represent discrete "snapshots" of gene expression at various timepoints and environmental conditions. The expression state of a transcriptional regulatory network containing *n* genes is represented by a vector $\boldsymbol{u}(t)$ in *n*-dimensional space. Each element of $\boldsymbol{u}(t)$ corresponds to the expression of one gene at time or state *t*. Next, we model all the regulative interactions between the genes of our model with a weight matrix, *W*, where each row of *W* represents all the regulatory inputs for one gene. The net regulatory effect, of gene *j* on gene *i* at some state *t* is simply the expression level of j, $u_j(t)$, times its regulatory influence on i, $w_{i,j}$. The total regulatory input to i, $r_i(t)$, is derived by summing across all the genes in the system (Eq. 1).

$$\text{(Eq. 1)} \qquad r_i(t)=\sum_j w_{i,j}\, u_j(t)$$

A positive value for $w_{i,j}$ models gene *j* stimulating the expression of gene *i*. Similarly a negative value models repression, while a value of zero indicates that gene *j* does not influence the transcription of gene *i*. In this way, each gene in the organism can have multiple inputs, both positive and negative, of differing strength. Thus, given the input levels of all genes at time or state *t*, we can calculate the "net" regulation state of each gene, expressed as an *n*-dimensional vector $\boldsymbol{r}(t)$. This matrix formulation of regulation is similar to previously described methods[10,11]. By modeling regulatory interactions with a weight matrix we can use extant matrix mathematical approaches found in linear algebra and neural networks for subsequent analyses of the resultant models.

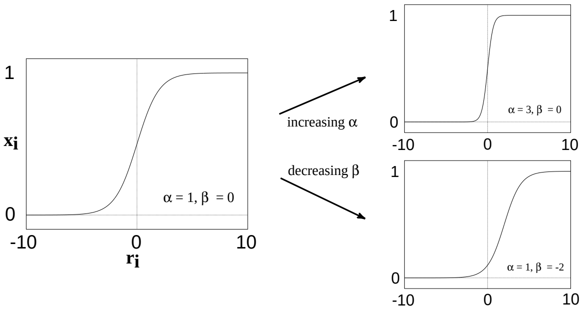

Figure 1. How the α and β constants adjust the dose-response function. The curve becomes more steeply sloped (becoming more like a step function) as α approaches infinity and it is shifted to the left when β is positive. Decreasing these constants has the opposite effect, making the curve more linear as α approaches 0 and shifting the curve to the right as β becomes negative.

Having derived the net regulatory state of each gene, we model the response of each gene to that regulatory input. The transcription response of gene *i* to $r_i(t)$ is calculated with a dose-response or "squashing" function.

$$\textit{(Eq. 2)} \quad x_i(t+1) = \frac{1}{1 + e^{-(\alpha_i r_i(t) + \beta_i)}}$$

where $r_i(t)$ is the net regulatory state of gene *i*, and α_i and β_i are two gene specific constants that define the shape of the dose-response curve for gene *i*. This assumes that each gene has a static dose-dependent response to activating and repressing regulatory influences. The α constant can be any positive real-number value and defines the slope of the curve at its inflection point (50% maximal expression). Genes with a large corresponding α_i will shift rapidly from near zero expression to near maximal expression when the activating inputs surpass some gene specific threshold, while those with a small α_i will have a nearly linear response to over the biologically relevant range of regulatory inputs. The β_i constant can be any real number and defines the curve's y-intercept, where the positive and negative regulatory inputs are equal. This point corresponds conceptually to the gene's basal level of expression. Positive β_i represents genes with high basal levels of transcription, while negative β_i represent genes with low levels of basal transcription. When modeling a regulatory network, the net regulatory state of each gene is input into an appropriate squashing function with its gene-specific constants the output of which is $x_i(t+1)$, the relative expression level for that gene at time or state *t*+1.

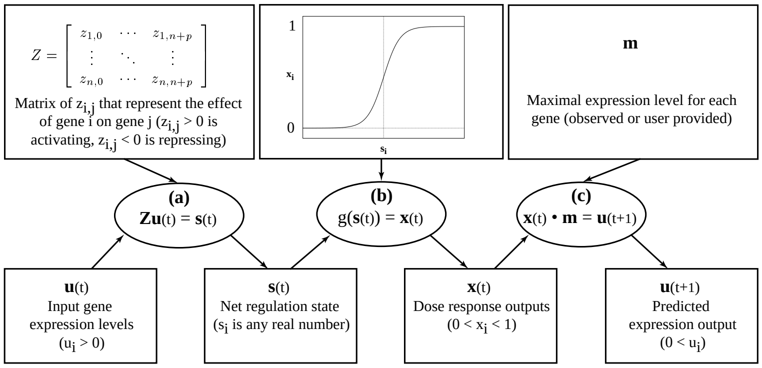
$Z = \begin{bmatrix} z_{1,0} & \cdots & z_{1,n+p} \\ \vdots & \ddots & \vdots \\ z_{n,0} & \cdots & z_{n,n+p} \end{bmatrix}$
Matrix of z_{i,j} that represent the effect of gene i on gene j (z_{i,j} > 0 is activating, z_{i,j} < 0 is repressing)
1
x_i
0
s_i
m
Maximal expression level for each gene (observed or user provided)
(a)
Zu(t) = s(t)
(b)
g(s(t)) = x(t)
(c)
x(t) • m = u(t+1)
u(t)
Input gene expression levels
(u_i > 0)
s(t)
Net regulation state
(s_i is any real number)
x(t)
Dose response outputs
(0 < x_i < 1)
u(t+1)
Predicted expression output
(0 < u_i)

Figure 2. A flow chart detailing the steps of this modeling method. (a) A vector, ***u(t)***, representing the input expression levels of all the genes (and bias term $u_0(t)$=1) in the regulatory network is mapped to a new vector, ***s(t)***, by weight matrix *Z*. Vector ***s(t)*** represents the net regulation state of all genes. (b) The relative expression response of each gene is calculated by inputting each element of ***s(t)*** into a gene specific dose-response function (Eq. 2). A relative expression level of 0 represents complete repression, while 1 represents maximal expression. (c) The relative expression levels are converted to "real" expression levels by multiplying by the empirically determined maximal expression level for each gene.

Because this relative expression level is a value between 0 and 1, with 0 representing complete transcriptional repression and 1 representing maximal expression, we must convert these relative levels into “real” units of expression. In addition, we want to allow the genes in our models to have different levels of maximal expression. To this end, we multiply the calculated relative gene expression level, x_i, by the maximal expression level for gene *i*, m_i, to get the “real” expression output for *i*, $u_i(t+1)$. In our simulations, m_i was randomly assigned values in a predetermined range set in each experiment. When applied to “real” biological data, m_i will have to be empirically determined (from the maximal observed expression level, for example) or defined for each gene .

By borrowing a page from recurrent neural networks[12], we can incorporate the α and β constants into the weight matrix and simplify our system of equations. We begin by replacing the original weight matrix *W* with a new matrix *Z* such that $z_{i,j}=\alpha_i w_{i,j}$. In addition, we can define a new column of weights in *Z*, such that $z_{i,0}=\beta_i$ and a new input value $u_0(t)$=1. Thus, the vector of net regulation states, ***r(t)***, becomes a new vector ***s(t)*** such that

$$(Eq.\ 3) \qquad s_i(t)=\sum_{j=0}^{n} u_j(t) z_{i,j}$$

and

$$(Eq.\ 4) \qquad x_i(t+1)=\frac{1}{1+e^{-s_i(t)}}$$

When these changes are compiled, we can formulate a new single equation that summarizes the whole model system:

$$(Eq.\ 5) \qquad u_i(t+1) = m_i x_i(t+1) = \frac{m_i}{1 + e^{-\Sigma z_{i,j} u_j(t)}}$$

2.2 Analysis of the behavior of this modeling methodology

We implemented TReMM in MATLAB 5.0 to test the validity of these models and to investigate their behavior. We generated random model regulatory networks ranging in size from 10 genes to 200 genes through the following 4 steps. (1) A maximal expression level was randomly assigned to each gene within a preset range that varied from experiment to experiment. (2) The α and β constants were set with a statistically normal distribution around a set base value. The specific base value and the breadth of the distribution varied from experiment to experiment. (3) Weight matrices were calculated with parameters defining the average percent of non-zero weights throughout the matrix, the maximum allowed weight absolute value, and the minimum allowed weight absolute value. Each gene was required to have at least one positive and one negative input, though different numbers of positive and negative inputs were allowed. The weight maximum and minimum were set for each gene such that the maximal activation or repression input to a gene would not exceed a set multiplicative factor (usually 2x) of that required to give maximal expression (set in (1)) or complete repression (*i.e.* $x_I<10^{-10}$). (4) Finally, the α's and β's were incorporated as described above to give the final weight matrix, *Z.* Step 3 assumes that no gene will have much more total potential regulative input than the amount of regulative input required to achieve the maximal transcriptional rate. Without this limitation on the weights (or with a large multiplier), the genes frequently display boolean-like behavior, oscillating between maximal expression and complete repression. Whether these assumptions are consistent with biological behavior and what constitutes a "good" linear simulation of biological regulation are questions beyond the scope of this work (for discussion see Thomas[12], Mjosness[10], and S. Kauffman[5]). For this work, we take these assumptions as a starting point for analyzing our algorithms.

We ran simulated time course experiments with these randomly generated models where the output from one state transition was used as the input vector for the next. These model networks were allowed to iterate until they reached a stable terminal state. In all models examined, the regulatory networks converged to an unchanging condition of gene expression or cyclical set of gene expression states, not unlike those observed in Boolean networks[13]. The number of time steps required for the network to reach a terminal state depended on the initial expression state of the network, the number of genes in the system, and the overall percentage of non-zero values in the weight matrix. Figure 3 displays an example of the behaviors observed from a single model system. Not surprisingly, larger networks required more time steps to arrive at a terminal state, regardless of the percent of non-zero weights in the system.

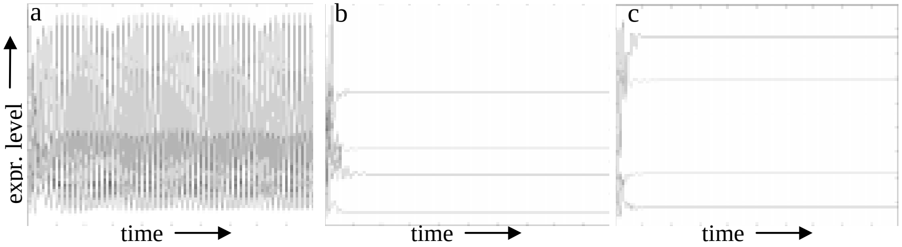

Figure 3. Three different terminal expression patterns from a single model system. These graphs plot the expression levels of 4 genes out of a 40 gene regulatory network. The weight matrix was the same in each simulated timecourse, but the initial starting conditions were different. In the terminal state, (a) the genes cycled through a set series of expression states (*i.e.,* a dynamic attractor), or were expressed at a constant levels (b and c).

2.3 Inclusion of environmental variables in a model system

Modeling transcriptional regulatory networks in this manner facilitates introduction of environmental variables into the modeling scheme. The experimental values of *p* environmental variables can be added to an input vector of *n* genes to generate a complete genetic and environmental input vector of size *n+p*. The regulative effect of these environmental inputs are modeled by adding *p* weights to each row of the weight matrix, expanding it from size *n* x *n*, to size *n* x *(n+p)*. These weights represent the activating and repressing influence of environmental factors on each gene's expression as conveyed through the cell's regulatory apparatus.

Random regulatory networks were constructed including up to 5 environmental variables and simulated timecourse experiments performed (though these scripts can track any number of environmental variables). In these experiments, the input values representing the environmental conditions were altered independently over the course of 100 timesteps.

Five different general behaviors were observed in response to these simulated environmental changes. Figure 4 shows a sampling of 5 genes from different 40 gene regulatory networks. (1) Some genes displayed a roughly linear dose-response to changing simulated environmental conditions, incrementing or decrementing smoothly to some stable "basal" level of expression (Fig.4, 1). (2) Other genes underwent an abrupt expression level transition, jumping rapidly from one relatively stable expression level to another (Fig.4, 2). (3) Genes expressed periodically under the initial environmental conditions transitioned to stable expression levels (Fig.4, 3). (4) Conversely, genes stably expressed initially transitioned to highly periodic expression (Fig.4, 4). (5) Finally, many genes were unaffected by changes to the environmental conditions (Fig.4, 5). Because the definition of this modeling scheme precludes testing all starting conditions, we do not know the frequency of each of these responses, or the commonalties amongst each model response.

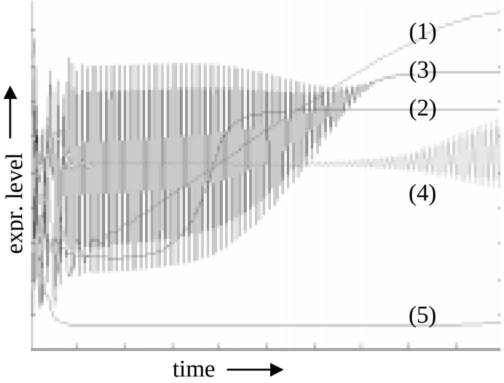

Figure 4. A sample of gene expression responses to environmental inputs. These five plots were compiled from 5 different 40-gene regulatory networks that included 1 environmental variable each. Genes displayed (1) linear response, (2) abrupt expression level transition, (3) transition from periodic to stable expression, (4) transition from stable to periodic expression, or (5) no response to changing simulated environmental conditions.

2.4 Reverse Engineering of Genetic pathways from expression data points

Another goal of this work is to predict the genetic pathways that underlie observed gene expression data. Given only input/output data sets we wanted to identify values for the weight matrix elements that define the regulatory network that relate an observed input to its corresponding output. The hope is that if our modeling scheme is a reasonable approximation of true biological networks, we may use it to predict genetic pathways from experimentally derived expression data.

Because our models treat the regulation of each gene as an independent event, the problem simplifies to calculating the weight matrix row for one gene at a time. Solving for these values on real biological data sets requires that we have a training data set with which we predict the weight matrix and a test data set consisting of at least 1 data point. This employs the assumption that there is a single weight matrix that describes all regulatory relationships in a biological system.

We divide each element of each output data point by the maximal expression level associated with that gene to get the relative expression level, x_i (a value between 0 and 1, see above). We are, again, assuming that the maximal expression levels of each gene can be provided through empirical observation. Next, we "desquash" this relative expression to obtain s_i, the net regulative state of that gene.

Eq. 6 $$s_i = -\ln\left(\frac{1}{x_i} - 1\right)$$

Finally, we calculate a weight matrix row that relates the inputs to the net regulation state that we have obtained from "desquashing" our training outputs. Given a known matrix M (in our case, all the inputs from our data points), an unknown vector $\boldsymbol{a}$ (a transpose of the weight matrix row corresponding to our gene of interest), and a known vector ***b*** (in which each element corresponds to the relative expression level of our gene of interest at a state transition), such that $M\boldsymbol{a}=\boldsymbol{b}$, we need to calculate ***a***. If we have as many input data points as there are genes in the regulation network, then the problem is "fully determined", and algebraically easy to solve. If we have fewer data points than genes, then the problem is "under-determined" and there are many equally good solutions to ***a*** in the equation $\boldsymbol{a}=M^{-1}\boldsymbol{b}$, including the "correct" one that is equal to the original weight matrix row. Thus the problem becomes solving for the "correct" inverse of the input matrix *M*. We tested both singular value decomposition[17] and the Moore-Penrose "pseudo-inverse" to calculate the matrix inverse. By applying the test input data points to the derived weight row, we get the corresponding predicted outputs, and the difference between the predicted output and the real test output is expressed as a euclidean error.

Next, we employ the assumption that most of the weight matrix values should be zero. Weights that are zero in the original weight row will frequently have small values in the predicted weight row. Therefore, we look at the predicted weights, identify the smallest weight absolute value, and set the corresponding column of the input matrix *M* to zero. M^{-1} is recalculated to get a new prediction of the weight matrix row. In this fashion, we iteratively remove additional input matrix positions, calculating the euclidean error after each step. The weight row prediction that produces the smallest error is kept. This process is repeated for all genes in the regulatory network. The final output is a weight matrix relating the regulative effect of all genes on all other genes in the system.

To test this reverse engineering approach, we generated algorithms in MATLAB that implemented the methodology laid out above. We call this script package REM (Reverse Engineering of Matrices). We generated random regulatory networks as described in section 2.2, and used those random networks to generate $n + 2$ input/output data points for each regulatory system. We then applied REM, using *n* training data points and 2 test data points and compared the resulting predicted model system to the known model system. REM's prediction accuracy was dependent on the number of non-zero weights in known model system (*i.e.* the number of regulatory inputs to a gene). Weight matrix rows with few weights were predicted with higher sensitivity than rows with many non-zero weights.

To test this approach in a more realistic situation, we generated data sets, into which normally distributed noise had been inserted into the outputs. The noise was constrained to be less than 10% of the true "signal". REM was then applied to those "noisy" input/output data sets and the resulting predicted model systems compared with the known model system used to generate the data. The sensitivity of our derived models improved as more "training" data points were used (Figure 5). Further examination of the false positive weights, reveals that they are roughly 10-fold smaller than true positive weights when the model system is derived from nearly as many data points as there are genes. This suggests that simple weight filtering could be employed, but this idea has not been tested.

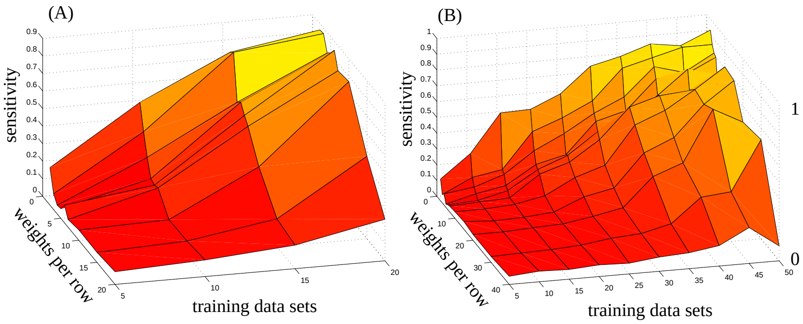

Figure 5. The general sensitivity surface of our weight matrix predictions as a function of the number of input data points and the number of non-zero weights per row. We randomly generated model regulatory networks and attempted to predict the weights of the model system using REM. Each data point represents the results from 100 weight matrix rows in a 20-gene (A) or 50-gene (B) network respectively with 2, 3, 4, 5, 6, 7, 10, 15, and 20 non-zero weights per row (also 30 and 40 non-zero weight for the 50-gene networks. Sn=TP/(TP+FP), where TP is true positives (correctly predicted non-zero weights) and FP is false positives (a weight that is zero, but was predicted to be non-zero).

3. Limitations of this approach

Like boolean models, these models make assumptions about the behavior of regulatory systems that are known to be untrue. For example, the assumption that all genetic interactions can be treated as independent events is contradicted by known transcriptional regulators that have different activities depending on their protein partners[14,15]. Also like boolean networks, we treat the control of transcription as a discrete time system. This assumption is necessary to make the problem computationally tractable. The most significant limitation of our modeling methodology comes from the biological processes being simplified by the weight matrix. Specifically, each weight attempts to relate a gene's expression level to that gene product's regulatory effect. Thus, the weights minimally assume a linear relationship between the number of copies of a gene's mRNA and the amount of resultant active gene product present in the cell. As systematic protein concentration data becomes available, we may be able to incorporate such data into our models to remove this linear assumption. These weights also encompass to some degree the contribution of each gene on each other gene's mRNA stability or rate of degradation. Each weight effectively represents the summation of all the positive and negative influences on a gene's final expression level, though this assumes that each gene will respond to all the positive and negative influences with the same sigmoidal curve.

Another significant limitation of any attempt to reverse engineer regulatory pathways from expression data (simulated or real) comes from the loss of information that occurs when a gene is being expressed at nearly maximal levels or is nearly completely repressed. As a gene's expression level asymptotically approaches 0 or maximal expression, it becomes impossible to "desquash" the output levels to any useful predictions of input regulatory state. Furthermore, the deleterious effects of noisy data are exacerbated near these limits, as small inaccuracies in expression level detection results in progressively larger errors in regulatory state calculation. Finally, it is worth reiterating that this system relies on the assumption that a gene's maximal expression level can be determined empirically, probably from the maximal observed level.

4. Future directions and perspectives

One limitation of our current analysis is that the infinite state space of these models is largely unexplored. While a complete analysis, like that performed in boolean networks is impossible[13], a more thorough and systematic analysis might be informative. Another problem is how to display the expression information across various simulated time course experiments. The DDLab[16] tool accomplishes this for boolean networks, but is not applicable to this method, again due to the lack of state space constraints. A possible solution to both problems may be derived from "binning" all possible expression states for each gene. For example, for those genes in our system that display highly non-linear responses, we may be able to represent their expression states with two "bins": near complete expression and near complete repression (thus making these genes exactly equivalent to the boolean networks). Other genes that display highly linear response over the regulatory range of the model would require more bins. In this way, we can rationally reduce the state space to a more tractable finite size without throwing out the more complex intergenic relationships that makes these models useful.

Similarly, our understanding of these models would be enhanced by tools that would construct regulatory models in which all possible combinations of model components were allowed to vary. To limit the number of possible weight combinations, we would again employ a binning approach, allowing the weights to be set to predetermined values that represented various ranges of regulatory interactions. Because our models treat the regulation of each gene as an independent event, we could build our understanding of larger regulatory systems by first studying how the combinatorial actions of various regulatory genes could influence a single gene, either with or without environmental inputs. Once various interesting regulatory single gene models were built and their behavior understood, higher order models could be built from those single gene model component and the higher order gene modeling problem investigated.

5. Conclusions

While much work remains before we understand the global behavior of these types of networks, this work lays a basis for modeling transcriptional regulatory networks with weight matrices. Though they still include idealizations of known gene regulation, these networks allow for inclusion of many different regulatory interactions and responses, like those observed in biological systems. TReMM provides a framework within which more accurate non-linear modeling components may be included. These networks allow for the facile inclusion of environmental variables with which we can describe the external conditions that influence the internal transcription regulation. Finally, by numerical analysis, this approach lends itself to predicting regulatory interactions from observed sets of expression data.

Acknowledgements

We thank B. Fornberg and A. Lapedes for stimulating discussions and ideas.

References

1. Berk, A.J. *Biochim. Biophys. Acta* **1009** 103-109 (1989).
2. Liang, S., Fuhrman, S., Somogyi, R., *Proc. Pacific Symposium on Biocomputing* **3** 18-29 (1998).
3. Fuhrman, S., Wen, X., Michaels, G.S., Somogyi, R., *Proc. International Conference on Complex Systems* 21-26 (1997).
4. Somogyi, R. and Sniegoski, C.A., *Complexity* **1**(6) 45-63 (1996).
5. Kaufmann, S.A., The Origins of Order, Self-Organization and Selection in Evolution. Oxford University Press (1993).
6. Ruvkun, G. and J. Guisto. *Nature* **338** 313-319 (1989).
7. Katz,W.S., R.J. Hill, T.R. Clandini, and P.W. Sternberg. *Cell* **82** 297-307 (1995).
8. Kalthoff, K. *Analysis of Biological Development*. McGraw-Hill:1996. 498-508.
9. Hopfield, J.J., *Proc. Nat. Acad. Sci.* **79** 2554-2558 (1982).
10. Mjolsness, E., Sharp, D.H., Reinitz, J., *J. Theor. Biol.* **152** 429-453 (1991).
11. Reinitz, J., Sharp, D.H., *Mechanisms of Development* **49** 133-158 (1995).
12. Thomas, R., Thieffry, D., Kaufman, M., *Bul. Math. Biol.* **57**(2) 247-276 (1995).
13. Wuenche, A., *Proc. Pacific Symposium on Biocomputing* **3** (1998).
14. Garrell, J. and S. Campuzano. *Bioessays* **13** 493-498 (1991).
15. Lamb, P. and S.L. McKnight. *TIBS* **16** 417-422 (1991).
16. Wuenche, A., http://www.santefe.edu/~wuenche/ddlab.html (1996).
17. Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. Numerical Recipes in C, 2nd Ed.. Cambridge University Press (1992) pp. 59-70.