Inference of Genetic Regulatory Networks with Recurrent Neural Network Models

Rui Xu, Xiao Hu, and Donald C. Wunsch II
Applied Computational Intelligence Laboratory, Dept. of Electrical and Computer Engineering,
University of Missouri – Rolla, Rolla, MO 65409-0249 USA
rxu@umr.edu, xhu@umr.edu, dwunsch@ece.umr.edu

Abstract— **Large-scale gene expression data coming from microarray experiments provide us a new means to reveal fundamental cellular processes, investigate functions of genes, and understand relations and interactions among them. To infer genetic regulatory networks from these data with effective computational tools has become increasingly important. Several mathematical models, including Boolean networks, Bayesian networks, dynamic Bayesian networks, and linear additive regulation models, have been used to explore the behaviors of regulatory networks. In this paper, we investigate the inference of genetic regulatory networks from time series gene expression in the framework of recurrent neural network model.**

Keywords—**Genetic regulatory networks, Recurrent neural networks, Back-Propagation through time, Particle swarm optimization.**

I. Introduction

With the rapid advancement of DNA microarray technologies [1], to infer genetic regulatory networks from time series gene expression data has become increasingly important in order to reveal fundamental cellular processes, investigate functions of genes, and understand complex interactions among genes [2, 3]. A genetic regulatory network consists of a set of DNA, RNA, proteins, and other molecules, and describes regulatory mechanisms among these components. Since all cells for a specific organism include the same genetic material, it is important to know which proteins are synthesized, or which genes are expressed, under certain conditions. This is achieved through the actions of some proteins, which activate or inhibit the transcription rates of certain genes by binding to certain regions of these genes. Therefore, the transcription of a specific gene, or the control of its gene expression, can be regarded as a combinatorial effect of a set of other genes.

Several computational models have been applied to investigate the behaviors of regulatory networks. Boolean networks consider a gene has only active or inactive states [4]. The effect of other genes on the state change of a given gene is described through a Boolean function. Although Boolean networks make it possible to explore the dynamics of a genetic regulatory system, they ignore the effect of genes at intermediate levels. Bayesian networks are graph models that estimate complicated multivariate joint probability distributions through local probabilities [5]. Bayesian networks are effective in dealing with noise, incompleteness, and stochastic aspects of gene expression data. However, Bayesian networks do not consider dynamical aspects of gene regulation and leave temporal information unhandled. Recently, dynamic Bayesian networks (DBN) attract more attention [6, 7]. DBN can model behaviors emerging temporally, and is effective in handling problems like hidden variables, prior knowledge, and missing data. The disadvantage of DBN is that DBN cannot scale well to large-scale data sets. For the linear additive regulation models [8, 9], the expression level of a gene at a certain time point can be calculated by the weighted sum of the expression levels of all genes in the network at a previous time point. Although linear additive regulation can reveal certain linear relations in the regulatory systems, it lacks the capability to capture the nonlinear dynamics between gene regulations. Considering the limitation of these methods, in this paper, we utilize recurrent neural networks (RNN) to infer genetic regulatory networks from time series gene expression data. In using RNNs for genetic network inference, we are mainly concerned with the ability of RNNs to interpret complex temporal behavior. Generalized recurrent neural network models can be considered as signal processing units forming a global regulatory network.

The paper is organized as follows. Section II focuses on the model of genetic networks with recurrent neural networks. Section III illustrates an application to the SOS DNA repair system. We conclude the paper in Section IV.

II. Recurrent Neural Networks

A. Model

For a continuous time system, the models can be represented through a neural network formulation [8-11],

$$\tau_i \frac{de_i}{dt} = f\left(\sum_{j=1}^{N} w_{ij} e_j + \sum_{k=1}^{K} v_{ik} u_k + \beta_i\right) - \lambda_i e_i , \qquad (1)$$

where e_i is the gene expression level for the i^{th} gene ($1 \le i \le N$, N is the number of genes in the system), $f()$ is a nonlinear function (usually, sigmoid function is used $f(z) = 1/(1+e^{-z})$), w_{ij} represents the effect of the j^{th} gene on the i^{th} gene ($1 \le i, j \le N$), u_k is the k^{th} ($1 \le k \le K$, K is the number of external variables) external variable, v_{ik} represents the effect of the k^{th} external variable on the i^{th} gene, τ is the time constant, β is the bias term, and λ is the decay rate parameter. A negative value of w_{ij} represents

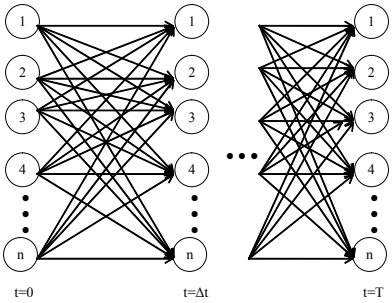

Fig 1. The description of a genetic network through a recurrent neural network model. Here, the regulatory network is assumed to be sparsely connected, although, the network can be constructed in a fully connected form principally.

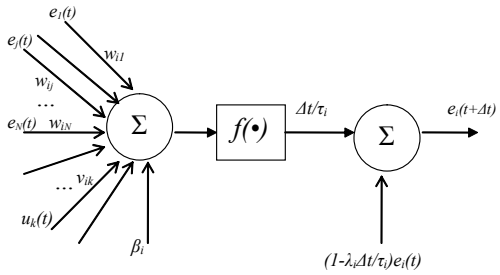

Fig. 2. A node (neuron) in the recurrent neural network model.

the inhibition of the j^{th} gene on the i^{th} gene, while a positive value indicates the activation controls. When w_{ij} is zero, there is no influence of the j^{th} gene on the expression change of the i^{th} gene.

This model can also be described in a discrete form (for computational convenience, since we only have measurement at some certain time points):

$$e_i(t+\Delta t) = \frac{\Delta t}{\tau_i} f\left(\sum_{j=1}^{N} w_{ij} e_j(t) + \sum_{k=1}^{K} v_{ik} u_k(t) + \beta_i\right) + \left(1 - \frac{\lambda_i \Delta t}{\tau_i}\right) e_i(t) \quad (2).$$

Fig. 1 depicts a recurrent neural network, which is unrolled in time from $t=0$ to T with an interval Δt, for modeling genetic network. Here, each node corresponds to a gene and a connection between two nodes defines their interaction. Fig. 2 illustrates a node in the recurrent neural network, which realizes the equation in (2).

B. Training Algorithms

There exist ample algorithms for RNN training. Here, we discuss the most commonly used algorithms, Back-Propagation through time (BPTT) [12] and particle swarm optimization (PSO) [13], for learning functional and structural parameters of regulatory networks from time series gene expression data.

BPTT was first proposed by Paul Werbos [12], which has been widely used for training a recurrent network, as an extension of the standard back-propagation algorithm. It may be derived by unfolding the temporal operation of the network into a layered feedforward network, the topology of which grows by one layer at every time step [14]. In the problem of gene network inference, the objective is to do reverse engineering to recover genetic network, which behaves in a manner reflected from the gene expression measurement. In other words, the regulatory interactions w_{ij} are to be recovered. Considering minimizing $J(e)$, some cost function of trajectory taken by e between 0 and T, for instance,

$$J(e) = \sum_{t=0}^{T} \sum_{i=1}^{n} (e_i(t) - d_i(t))^2 \quad (3)$$

which measures the deviation of network output $e(t)$ from the measurement (target) $d(t)$. More elaborate error terms can be easily added. By using BPTT, we find the derivatives of the cost function J with respect to the individual weights w_{ij} of the network. These derivatives can be used to do gradient descent on the weights, updating them in the direction that minimizes E:

$$\Delta w_{ij} = -\eta \frac{J(e)}{w_{ij}} \quad (4).$$

The learning rule in the discrete time is:

$$z_i(t+\Delta t) = \Delta t \xi(t) + (1 - \frac{\lambda_i \Delta t}{\tau_i}) z_i(t+\Delta t) + \Delta t \sum_{j=1}^{n} \frac{1}{\tau_j} w_{ij} z_j(t+\Delta t) f'\left(\sum_{k=1}^{n} w_{kj} e_j(t) + \beta_j\right) \quad (5)$$

$$\frac{\partial J}{\partial w_{ij}} = \frac{1}{\tau_j} \sum_{t=0}^{T} e_i(t) f'\left(\sum_{k=1}^{n} w_{kj} e_j(t) + \beta_j\right) z_j(t+\Delta t) \Delta t \quad (6)$$

$$z_i(t) = \frac{\partial J^+}{\partial e_i(t)} \quad (7)$$

$$\xi_i(t) = \frac{\partial J}{\partial e_i(t)} \quad (8)$$

where ∂^+ denotes the ordered derivatives of Werbos [12]. Since it is usually difficult to have the measurements of the external variables, it is a common practice to ignore the term $\sum_{k=1}^{K} v_{ik} u_k(t)$ in the derivation of the learning algorithm.

PSO is a new evolutionary computation technique for global optimization, which comes from the idea of simulation of social behavior [13]. PSO is particularly useful in evolving neural networks when there exist many local optima, and traditional gradient-based search algorithms are easy to get stuck. Different from genetic algorithm [15], a random velocity is associated with each potential solution, called a particle, which are considered to "be flown through the problem space" [13]. The basic idea of PSO lies in accelerating each particle towards its corresponding *pbest* and the *gbest* locations at each time step, in which *pbest* is the previous best solution according to the calculated fitness and *gbest* is the best overall value in the whole swarm. It has been shown that PSO require less computational cost and can achieve faster convergence than conventional back-

propagation in training feedforward neural networks for approximating a nonlinear function [16]. The procedure for implementing PSO for training a RNN in a batch mode for modeling of genetic networks is as follows:

i). Initialize a population of particles with random positions and velocities of D dimensions. The dimensionality D of the problem space is dependent on the number of genes in the regulatory system.

ii). Calculate the estimated time series and evaluate the optimization fitness function for each particle. Here, the design of fitness function aims to minimize the cumulative error between the network outputs and the targets.

iii). Compare each particle's fitness value with its *pbest*. If current value is better than *pbest*, reset both *pbest* value and location to the current value and location.

iv). Compare each particle's fitness value with *gbest*. If current value is better than *gbest*, reset *gbest* to the current particle's array index and value.

v). Update the velocity and position of the particle with the following equations.

$$V_{iD} = W_I \times V_{iD} + c_1 \times rand_1 \times (pbest_{iD} - X_{iD}) + c_2 \times rand_2 \times (gbest_{iD} - X_{iD}) \quad (9)$$

$$X_{iD} = X_{iD} + V_{iD} \quad (10)$$

where X_{iD} and V_{iD} are the position and velocity of the i^{th} particle, respectively, W_I is the inertia weight, c_1 and c_2 are the acceleration constants, and $rand_1$ and $rand_2$ are uniform random functions.

vi). Return to step ii until a stop condition is satisfied.

Typically, the gene expression data currently available contain measurements of thousands of genes, but only with a limited number of time points (less than 50). This so-called "curse of dimensionality" [14] limits the application of many data-driven computational models and makes it very difficult to infer a fully determined large-scale regulatory network and make accurate prediction of future expression level. Fortunately, biological knowledge on genetic regulatory networks assumes that a gene is only regulated by a limited number of genes. In other words, the regulatory networks are sparsely connected and most weights values are zeroes. It is reasonable to identify the weights whose values are non-zeroes from these data, which indicate the interactions among genes. Wahde and Hertz proposed a procedure for unraveling the potential interactions between genes by iteratively searching non-significant parameters [15]. However, to identify these non-significant parameters is not trivial. We propose a new algorithm to use PSO to evolve both connection weights and network structures (network connectivity) [17]. This avoids the exhaustive enumeration of all possible connectivity (although generally much less than N, but the search space is still large) and has the potential to explore large-scale regulatory networks. Meanwhile, PSO has many desirable characteristics, e.g., easy to implement, flexibility in balancing global and local exploration, and particularly, the memory mechanism for keeping previous best solutions, which make PSO a powerful tool to explore sophisticated space and suitable for regulatory networks inference.

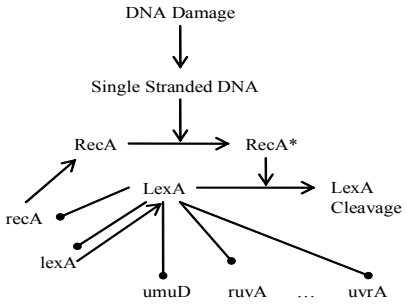

Fig. 3. The SOS DNA Repair network [5]. Inhibitions are represented by −●, while activations are represented by →.

III. Results

We employ the proposed model to analyze the SOS DNA Repair network in bacterium *E. coli* depicted in Fig. 3 [18]. The data include the expression measurements for 8 major genes through 50 time points, sampled every 6 minutes. When damage occurs, protein RecA becomes activated and mediates LexA autocleavage by binding to single-stranded DNA molecule. Protein LexA is a master repressor that represses all genes when no damage occurs. The drop in LexA expression levels causes the activation of the SOS genes. After the damage is repaired, the expression level of RecA falls, which causes the accumulation of LexA. LexA binds sites in the promoter regions of these SOS genes and represses their expression. The cells return to their original states.

Fig. 4 shows the real gene expression profiles and the learned profiles with both BPTT and PSO. We can see the proposed model can effectively capture the dynamics of most genes in the network, and the major change trends of the gene expression levels are reflected in the learning curves. The average mean square errors between the real profiles and learned profiles are 0.002 and 0.022 for BPTT and PSO, respectively. In order to infer the potential relations among genes, we run the algorithm based on PSO search for 10 times and check the best solutions for each individual in the swarm (100 in total). We identify the non-zeroes weights by choosing those whose average number of times selected in the solutions is above certain threshold. The results show that we can identify the inhibition of LexA on uvrD, recA, and ruvA, and the activation of recA on lexA. Several false positives are also included. Similar analysis is employed on the results achieved by BPTT. The experiments are run by thirty times. The means and the standard deviations of the learned weight matrix are used to infer the potential regulations among genes. The results also clearly show the inhibition of the LexA on umuD, uvrA and polB, and the activation of recA on lexA. As we can see, a combination of PSO and BPTT provides a robust revealing

of the internal regulatory interactions among genes. More results and analysis from both PSO and BPTT can be found in [17] and [19], respectively.

IV. Conclusion

To understand the gene regulatory mechanisms is one of the central tasks in molecular genetics. Inference of genetic regulatory networks based on time series gene expression data from microarray experiments becomes an important and effective way to achieve this goal. Herein, we introduce how to employ recurrent neural networks to model regulatory systems and reveal the potential interactions. Given the similarity between recurrent neural network and gene networks, we believe that recurrent neural networks, such as the one we have proposed here, will play an important role in unraveling the mystery of gene regulation relationships. However, with the limited data, current research only focuses on the modeling of network from synthetic data, or simulation of small-scale network including only several genes or gene clusters. No attempt has been made to infer large-scale genetic regulatory networks. High quality time series gene expression data with sufficient number of time points is particularly important. In the meantime, further improvement for the current computational models is also required in order to explore gene regulation more effectively.

Acknowledgment

Partial support for this research from the National Science Foundation, and from the M.K. Finley Missouri endowment, is gratefully acknowledged.

References

[1] M. Eisen and P. Brown, "DNA Arrays for Analysis of Gene Expression," Methods Enzymol, vol. 303, pp. 179-205, 1999.

[2] P. D'haeseleer, S. Liang, and R. Somogyi, "Genetic Network Inference: From Co-expression Clustering to Reverse Engineering," Bioinformatics, vol. 16, no. 8, pp. 707--726, 2000.

[3] H. De Jong, "Modeling and Simulation of Genetic Regulatory Systems: A Literature Review", Journal of Computational Biology, vol. 9, pp. 67-103, 2002.

[4] I. Shmulevich, E. Dougherty, and W. Zhang, "From Boolean to Probabilistic Boolean Networks as Models of Genetic Regulatory Networks", Proceedings of the IEEE, vol. 90, no. 11, pp. 1778- 1792.

[5] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using Bayesian Networks to Analyze Expression data", Journal of Computational Biology, vol. 7, pp. 601-620, 2000.

[6] B. Perrin, L. Ralaivola, A. Mazurie, S. Battani, J. Mallet, and F. d'Alché-Buc, "Gene Networks Inference Using Dynamic Bayesian Networks", Bioinformatics, vol. 19, Suppl.2, pp. ii138-ii148, 2003

[7] Y. Tamada, S. Kim, H.Bannai, S. Imoto, K. Tashiro, S. Kuhara, and S. Miyano, "Estimating Gene Networks from Gene Expression Data by Combing Bayesian Network Model with Promoter Element Detection", Bioinformatics, vol. 19, Suppl.2, pp. ii227- ii236, 2003.

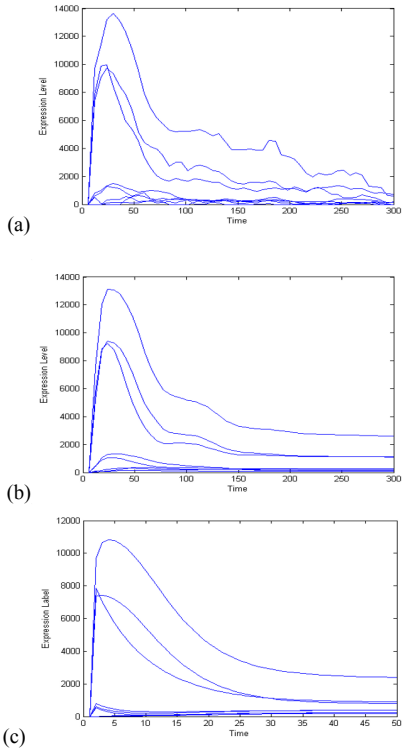

Fig. 4. (a) The measured gene expression profiles; (b) The learned mean expression profiles with PSO; (c) The learned expression profiles with BPTT.

[8] P. D'haeseleer, "Reconstructing Gene Network from Large Scale Gene Expression Data", Dissertation, University of New Mexico, 2000.

[9] E. van Someren, L. Wessels, and M. Reinders, "Genetic Network Models: A Comparative Study", In Proc. of SPIE, Micro-arrays: Optical Technologies and Informatics (BIOS01), vol. 4266, pp. 236--247, 2001.

[10] E. Mjolsness, T. Mann, R. Castaño, and B. Wold, "From Co-expression to Co-regulation: An Approach to Inferring Transcriptional Regulation among Gene Classes from Large-scale Expression Data", in Advances in neural Information Processing Systems 12, pp. 928-934, MIT Press, 2000.

[11] D. Weaver, C. Workman, and G. Stormo, "Modeling Regulatory Networks with Weight Matrices", Proceedings of the Pacific Symposium on Biocomputing, pp. 112-123, 1999.

[12] P.J. Werbos, "Backpropagation Through Time: What It Does And How to Do It", Proceedings of IEEE, 78(10), pp. 1550-1560, 1990.

[13] J. Kennedy, R. Eberhart, Y. Shi, "Swarm Intelligence", Morgan Kaufmann Publishers, 2001.

[14] S. Haykin, "Neural Networks: A Comprehensive Foundation", *2nd Ed*., Prentice Hall, New Jersey, 1999.

[15] M. Wahde and J. Hertz, "Modeling Genetic Regulatory Dynamics in Neural Development", Journal of Computational Biology, 8, 429-442, 2001.

[16] V. Gudise and G. Venayagamoorthy, "Comparison of Particle Swarm Optimization and Backpropagation as Training Algorithms for Neural Networks", Proceedings of the 2003 IEEE Swarm Intelligence Symposium, pp. 110-117, 2003.

[17] R. Xu and D. Wunsch, "Genetic Regulatory Networks Inference with Particle Swarm Optimization," in preparation, 2004.

[18] M. Ronen, R. Rosenberg, B. Shraiman, and U. Alon, "Assigning Numbers to the Arrows: Parameterizing a Gene Regulation Network by Using Accurate Expression Kinetics", Proc. Natl. Acad. Sci., vol. 99, no. 16, pp.10555-10560, 2002.

[19] X. Hu, A. Maglia and D. Wunsch, "A General Recurrent Neural Network Approach to Model Genetic Regulatory Networks," in preparation, 2004.