Reconstruction of Gene Regulatory Networks from Gene Expression Data Using Decoupled Recurrent Neural Network Model

Nasimul Noman, Leon Palafox, and Hitoshi Iba

Graduate School of Information Science and Technology,
University of Tokyo, Tokyo 113-8656, Japan
{noman,leon,iba}@iba.t.u-tokyo.ac.jp

Abstract. In this work we used the decoupled version of the recurrent neural network (RNN) model for gene network inference from gene expression data. In the decoupled version, the global problem of estimating the full set of parameters for the complete network is divided into several sub-problems each of which corresponds to estimating the parameters associated with a single gene. Thus, the decoupling of the model decreases the problem dimensionality and makes the reconstruction of larger networks more feasible from the point of algorithmic perspective. We applied a well established evolutionary algorithm called differential evolution for inferring the underlying network structure as well as the regulatory parameters. We investigated the effectiveness of the reconstruction mechanism in analyzing the gene expression data collected from both synthetic and real gene networks. The proposed method was successful in inferring important gene interactions from expression profiles.

Keywords: Recurrent Neural Network model, gene network reconstruction, decoupled RNN, differential evolution.

1 Introduction

In recent years, with the advent of various gene expression assaying techniques, the study of the relationship among genes has been highlighted extensively. Gene expression data, whether in time-course format or steady state format, provides an opportunity to observe the interaction among thousands of genes simultaneously. Given that sufficient amount of gene expression data is available, in principle it is possible to derive the detailed quantitative model of the network that adequately represents the dynamics of the underlying system [1].

Nevertheless, several practical issues such as small sample size compared to the number of genes, the presence of biological noise and experimental noise, inadequate knowledge and representation of the complex dynamics and nonlinear relationship among the genes, the problem dimension, makes the adequate reconstruction of the network a very challenging task [2]. Several techniques have been proposed in the field of computational intelligence for algorithm based reconstruction of gene regulatory networks (GRN) that help biologists to form new hypotheses about biological systems and to design new experiments [2–4].

Y. Suzuki and T. Nakagaki (Eds.): WSH 2011 and IWNC 2012, PICT 6, pp. 93–103, 2013.

In order to apply a computational approach for inferring GRN from experimental data, a mathematical model is necessary to formalize the process of gene regulation. The modeling endeavor for GRN, started long ago, produced a large variety of models over the last couple of decades. The modeling of GRN has diverged in many directions such as: discrete versus continuous, linear versus non-linear, deterministic versus stochastic, graphical versus non-graphical, synchronous versus asynchronous etc. All of these modeling paradigms have their strength and weakness in terms of representation accuracy, computational feasibility, noise proneness and data requirement [5].

In this work we have used the recurrent neural network (RNN) model [6] along with a natural computational method to extract regulatory interactions among genes from gene expression data sets. The canonical RNNs are neural networks with delayed feedbacks. With the network of nonlinear processing elements, RNN can adequately capture the nonlinear and dynamic interaction among genes [7]. Many of the reconstruction approaches applied to infer GRN using RNN belongs to the field of natural computation. Some of the researchers used genetic algorithms (GA) for reconstructing the target network using the single layer RNN [6, 8] and the multilayer RNN [9] architecture. A couple of swarm intelligence approaches have been proposed in which swarm algorithms (particle swarm optimization and ant colony optimization) have been applied for estimating network structure and parameters [7, 10]. Evolutionary algorithms other than GA have also been used for estimating RNN parameters for GRNs [11]. Some hybrids of natural computations with other approaches have been also used for reverse engineering GRNs using RNN formalism [2, 12].

The RNN model offers a good compromise between the biological proximity and mathematical flexibility for representing GRNs. However, inference of GRN using RNN requires the estimation of $N(N+3)$ parameters, where N is the number of genes in the network. Generally, with the increase of the dimension, the problem complexity increases rapidly and locating the global optimum solution becomes difficult for the search algorithm. Therefore, in order to deal with the challenges of high-dimensionality, with increasing genes in the network, here we use a decoupled form of the RNN model for inferring GRN. Other than reducing the problem dimension, such decoupling facilitates the design of parallel algorithms for GRN inference. In this work we have used a natural computational approach called differential evolution (DE), belonging to the group of evolutionary algorithm, for identifying the regulatory interactions form expression profiles using the decoupled form of RNN. We tested the proposed method using artificial gene regulatory networks of different dimensions and a real network. Experiments showed that the proposed approach can provide a good estimate of the structure of genetic networks.

The rest of the paper is organized as follows. The next section describes the decoupled RNN model. In section 3, we present the DE algorithm for inferring RNN model based gene networks. The fourth section reports the experiments with the results to verify the effectiveness of the proposed method. In section 5 we conclude the paper with some general discussions.

2 Recurrent Neural Network (RNN) Model

The recurrent neural network (RNN) model formulates the genetic interactions in terms of a neural network in which the nodes correspond to genes and the connections correspond to regulatory interactions among genes [6, 13]. In canonical RNN model the interactions among genes is represented in terms of a tightly coupled system given by

$$\frac{de_i}{dt} = \frac{1}{\tau_i}\left(g\left(\sum_{j=1}^{N} w_{ij}e_j + \beta_i\right) - \lambda_i e_i\right) \tag{1}$$

where e_i represents the gene expression level for the i-th gene ($i \leq N$, N is the total number of genes in the network). w_{ij} represents the type and strength of the regulatory interaction from j-th gene towards i-th gene. The positive (negative) value of w_{ij} represents activation (repression) control of gene-j on gene-i. $w_{ij} = 0$ means gene-j has no regulatory control on gene-i. β_i represents the basal expression level and λ_i denotes the decay rate parameter of the i-th gene. The function $g(\cdot)$ introduces non-linearity to the model which is often given by the sigmoid function. In the canonical form the RNN model for GRN can be described by the following set of $N(N+3)$ parameters $\Omega = \{w_{ij}, \beta_i, \lambda_i, \tau_i\}$ where $i, j = 1, 2, \cdots, N$.

In this work we estimated the regulatory interactions towards a particular gene at a time, independent of interactivity on other genes. In other words, we have divided the $N(N+3)$ dimensional problem into N sub-problems of size $(N+3)$ and solved each separately. In sub-problem i $(i = 1, 2, \cdots, N)$ we estimated the model parameters $\Omega_i = \{w_{ij}, \beta_i, \lambda_i, \tau_i\}$ $(j = 1, 2, \cdots, N)$. Then we accumulate all the learned parameters to build the complete network model. Similar procedure for learning the interactions separately has been applied with other neural network models [14] or with some other GRN models [15, 16]. In addition to reducing the problem dimension, this decoupling procedure makes the parallel solution of each sub-problem possible.

3 Reverse Engineering Algorithm

Here we used a natural computational approach for reverse engineering GRN modeled by decoupled RNN. We employed differential evolution (DE) [17] for searching the optimal model parameters that can reproduce the target time courses of genes. DE is a new generation EA proven to be very successful in solving different complex problems arising in different domains. It has also been very effective in reverse engineering GRN using the canonical RNN model [11, 18]. Hence, we chose DE for identifying the regulatory interactions among genes using the decoupled RNN formalism.

Like most of the EAs, DE starts with a population of random solution where each individual of the population encodes a candidate solution for the problem under consideration. Here we apply a separate instance of DE for estimating the

parameters of each target gene in the network. In other words, in sub-problem i, every individual of DE represents the parameters for gene-i that is $\Omega_i = \{w_{ij}, \beta_i, \lambda_i, \tau_i\}$ $(j = 1, 2, \cdots, N)$. After random initialization, the fitness of the individual is calculated using the fitness function described later. Then for each individual x_G^i, $i = 1, \cdots, P$ in the current generation, G, a mutated individual is generated by the following *mutation* operation

$$y_G^i = x_G^j + F(x_G^k - x_G^l) \tag{2}$$

where x_G^j, x_G^k and x_G^l are random individuals selected from generation G such that j, k and $l \epsilon \{1, \cdots, P\}$ and $i \neq j \neq k \neq l$; P is the number of individuals in G and F is called the *scaling factor* – a parameter of DE.

Afterwards, the mutated individual y_G^i participates in a *crossover* operation with the current population member x_G^i and generates the *offspring* y_{G+1}^i. In the *crossover* operation, the genes (parameters) of the offspring y_{G+1}^i are randomly inherited from x_G^i or y_G^i determined by a parameter called *crossover factor* CF, i.e. if $r \leq CF$ (where r is a uniform random number in $[0, 1]$) then it is inherited from x_G^i otherwise from y_G^i. Finally, the offspring is evaluated and replaces its parent x_G^i in next generation if its fitness is the same or better than its parent. This is the *replacement* process for producing new generation. And this process is iterated generation after generation until a satisfactory solution is found or a maximum number of generations (G_{max}) have elapsed.

Because of the flexibility of the model, the search space contains many local optimum that traps the search algorithm and the global optimum remains undiscovered. In order to help the algorithm to get out of a local optimum we embedded a random restart strategy in DE that randomly reinitializes all the individuals except the elite one, if the difference between the best fitness (f_{best}) and the worst fitness (f_{worst}) of the current generation falls below a threshold ($\delta \cdot f_{best}$). After the random restart, the algorithm proceeds in its regular mode. When the optimization of one instance of DE finishes, we receive the set of parameters for one gene. Repeating this process for all genes and compiling them together we get the complete set of parameters for the whole network.

3.1 Fitness Evaluation Criteria

We need some assessment mechanism for evaluating the alternate GRN models we come across in course of the evolutionary process. The most commonly used model evaluation process is the quantitative difference between the response generated by the candidate model and the experimentally collected response. This evaluation process calculates the model fitness using a function called mean squared error (MSE). The reverse engineering of GRN, like other dynamic systems, can be done with higher accuracy if multiple time series for the same gene could be used. Since we are estimating the parameters of each gene separately, the fitness evaluation process takes the time courses of a particular gene in consideration. Using M sets of time dynamics, the MSE based fitness function for the sub-problem i, corresponding to gene-i, is given by

$$f(\Omega_i) = \frac{1}{TM} \sum_{k=1}^{M} \sum_{t=1}^{T} \left(e_{k,i}^{cal}(t) - e_{k,i}^{exp}(t) \right)^2 \tag{3}$$

where $e_{k,i}^{exp}(t)$ and $e_{k,i}^{cal}(t)$ represent the expression levels of i-th gene in the k-th set of time courses at time t in experimental and simulated data respectively.

Generally, very few genes or proteins regulate the expression level of a specific gene [19]. But the general model of RNN considers every possible interaction from each gene. Because of the model flexibility, if we allow all possible regulations then the search algorithm gets stuck to some local minimum that can generate the time course very closely. One effective way of recovering the target skeletal structure is to penalize the fitness score in proportional to the network complexity [20, 21]. Here we use a penalty term similar to that used in [22]. The penalized fitness function that was used for evaluating the models are given by

$$f(\Omega_i) = \frac{1}{TM} \sum_{k=1}^{M} \sum_{t=1}^{T} \left(e_{k,i}^{cal}(t) - e_{k,i}^{exp}(t) \right)^2 + c \sum_{j=1}^{N-I} (\widehat{w_{ij}}) \tag{4}$$

where $\widehat{w_{ij}}$ are the weights of interactions towards gene-i sorted in ascending order of their magnitude. I indicates a limit on the maximum allowed regulations for a gene. If the number of interactions exceeds this limit then the pruning term will penalize the fitness function. c represents the penalty constant.

4 Experimental Results

In this work, the suitability of the GRN inference using the decoupled RNN model was primarily validated using synthetic networks since the actual structure and parameter values are unknown for real networks. Two different networks of different sizes and architectures were used for this purpose. The reconstruction experiments were carried out under the ideal noise-free condition and with simulated noise corrupted gene expression data. We also attempted to reconstruct the SOS DNA repair network of *Echericha coli* using the proposed method.

4.1 Artificial Network Inference

In our first experiment with *in silico* networks, we investigated whether it is possible to infer the regulatory interactions and correct parameter values for a target gene network using the decoupled form of RNN model. In the reconstruction experiment we used a small scale network that has been studied by others in canonical RNN model [6, 10, 11]. The parameters for the RNN model for this four gene network, called NET1 hereafter, is shown in Table 1. In NET1 $\lambda_i = 1$ used for all genes as done in other work [6, 10, 11].

The artificial gene expression data was generated by simulating the canonical model of RNN in Table 1. The initial gene expression level was selected randomly. We generated $M = 10$ sets of gene expression profiles for NET1 where each time

Table 1. RNN model of the target synthetic network NET1

w_{ij}	1	2	3	4	β_i	τ_i
1	20.0	-20.0	0.0	0.0	0.0	10.0
2	15.0	-10.0	0.0	0.0	-5.0	5.0
3	0.0	-8.0	12.0	0.0	0.0	5.0
4	0.0	0.0	8.0	-12.0	0.0	5.0

Table 2. Inferred RNN model for NET1 from 5% noisy data

w_{ij}	1	2	3	4	β_i	τ_i
1	49.99	-20.14	-7.79	0.00	-3.33	10.04
2	18.15	-12.28	0.51	0.00	-6.21	5.19
3	0.00	-8.19	11.46	-0.77	0.48	4.25
4	0.00	-0.29	6.51	-9.20	-0.27	4.48

courses contains $T = 50$ time samples. In order to simulate the noise experienced in the real gene expression data we generated expression profiles adding 5% Gaussian noise. Then we tried to reverse engineer the target network from both the noise-free data and noise-corrupted data.

In our experiments, we inferred the regulators of gene-i ($i = 1, \cdots, N$) under the same experimental condition. For every sub-problem the algorithmic setup was as follows: $F = 0.5$, $CF = 0.9$, $P = 100$, $G_{max} = 10000$, $\delta = 1 \times 10^{-3}$, $c = 10$ and $I = 4$. The setting for DE parameters (F, CF, and P) is very typical [23] and other parameters were chosen based on the setting used in [22] or empirically. The search ranges for RNN parameters were as follows: $w_{ij} \in [-30.0, 30.0]$, $\beta_i \in [-10.0, 10.0]$, $\tau_i \in [0.0, 20.0]$. We did not include λ_i in our search as it was fixed in the target model. We implemented the algorithm in Java and experiments were run in a Intel® Core™$i7$ CPU 2.67 GHz computer with 8GB RAM. Each experiment was repeated 10 times to confirm the reliability of the stochastic search algorithm.

In the reconstruction experiments from noise-free gene expression data we could precisely estimate the network structure and the parameter values. In almost every optimization run the fitness score for the models reached to zero or very close to zero ($< 1 \times 10^{-15}$) and the estimated parameters were exactly the same as the target. Although it was a very simple and small network, these experiments verify that if sufficient expression data is given and the dynamics are free from noise, then it is possible to estimate the network structure and kinetics using the decoupled form of RNN model.

We also analyzed the performance of the reconstruction algorithm in inferring NET1 from noisy expression data. The experimental condition was exactly the same as before except $I = 3$ was used. Table 2 shows the estimated network structure and parameter values achieved in a sample run. From Table 2 it is evident that even in presence of noise all the regulatory interactions among the genes were identified correctly. However, the estimated parameter values for the

Table 3. Summary of the results of NET1 reconstruction from 5% noisy data

Measure	5% noisy data
Sensitivity	1.000
Specificity	0.500
Accuracy	0.750
MCC	0.577

Table 4. RNN model of the target synthetic network NET2

$w_{i,j}$	$w_{1,14} = -15$, $w_{5,1} = 10$, $w_{6,1} = -20$, $w_{7,2} = 15$, $w_{7,3} = 10$, $w_{8,4} = 20$, $w_{9,5} = -20$, $g_{9,6} = 10$, $g_{9,17} = 10$, $w_{10,7} = -10$, $w_{11,4} = -15$, $g_{11,7} = 15$, $w_{11,22} = -15$, $w_{12,23} = 10$, $w_{13,8} = 20$, $w_{14,9} = 15$, $w_{15,10} = -10$, $w_{16,11} = 15$, $w_{16,12} = -15$, $w_{17,13} = -20$, $w_{19,14} = -15$, $w_{20,15} = 10$, $w_{21,16} = -20$, $w_{23,17} = -10$ $w_{24,15} = -15$, $w_{24,18} = -20$, $w_{24,19} = 15$, $w_{25,20} = -10$, $w_{26,21} = 20$, $w_{26,28} = 20$, $w_{27,24} = -15$, $w_{27,25} = 10$, $w_{27,30} = 15$, $w_{28,25} = -15$, $w_{29,26} = 10$, $w_{30,27} = 15$, other $w_{i,j} = 0.0$
β_i	$\beta_i = 5$ for $i = \{2, 5, 6, 10, 16, 24, 28\}$ $\beta_i = -5$ for $i = \{15, 17, 27\}$ otherwise $\beta_i = 0$
τ_i	$\tau_i = 10$ for $i = \{1, \cdots, 30\}$
λ_i	$\lambda_i = 1$ for $i = \{1, \cdots, 30\}$

network kinetics were not very precise. Additionally, some false positive regulations were predicted. Nevertheless, if we consider the magnitude of these false positives then it is obvious that those were pretty small compared to real regulations. The summary of the prediction in terms of sensitivity, specificity, accuracy and Mathew's correlation coefficient (MCC) is presented in Table 3. Here, we used the standard definitions for these measurements based on positive/negative value of w_{ij}. These results show that the prediction had a full 1.00 sensitivity and a MCC greater than 0.5, however, the specificity was 0.5 which indicates prediction of 50% false positive regulations. In an overall, the approach did a correct estimation of NET1 structure and good approximation of the parameters.

Next we experimented with a larger network (NET2) with $N = 30$ genes to investigate the inference capability of the algorithm. The structure of the network was very sparse and it is the same architecture that was used in [21]. The parameters of NET2 were chosen arbitrarily as shown in Table 4. We performed the reconstruction experiment both in noise-free condition and 5% noise corrupted environment. The experimental conditions were once again kept the same except $I = 5$ was used to limit the maximum number of regulations for a particular gene.

It is known that with the increase of dimensionality the problem complexity increases rapidly and the GRN prediction problem is not an exception. Therefore, in predicting the correct regulations, even in ideal condition, the inference algorithm had some difficulty. In noise-free condition, the method identified more than 50% regulations correctly and identified most of the true negatives. However, the algorithm inferred many false positives and some true negatives. The summary of the prediction from noise-free condition and 5% noisy condition is

Table 5. Summary of the results of NET2 reconstruction

Measure	Noise free data	5% noisy data
Sensitivity	0.611	0.305
Specificity	0.996	0.981
Accuracy	0.981	0.954
MCC	0.7246	0.329

presented in Table 5. It is shown in Table 5 that when the gene expression data was noisy, the prediction accuracy deteriorated in terms of sensitivity and MCC. However, the specificity and accuracy was not affected much because these values were dependent on the choice of I. If we had chosen a smaller value of I, then it was possible to increase the prediction accuracy even more. However, it can be summarized that the prediction method could made an overall estimate of the network structure for a reasonably large and sparse network given sufficient gene expression data and a reasonable level of noise.

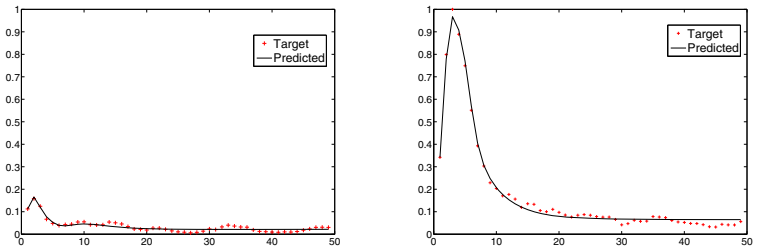
4.2 SOS DNA Repair Network Inference

We tested the proposed algorithm in the reconstruction of well-known SOS DNA repair network in *Escherichia coli*. The SOS network, consisting of 40 genes, is initiated when any damage in DNA or interference in DNA replication process is detected [24, 25]. However, the core repair system is controlled by the interplay between *RecA* and *LexA* proteins. More details about the working mechanism of the SOS DNA repair system in *E. coli* could be found in [26].

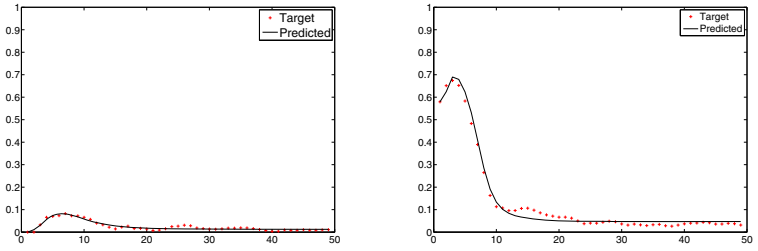
We used the gene expression data set collected in Uri Alon Lab[1]. The data set contains expression levels of 8 genes (*uvrD, lexA, umuD, recA, uvrA, uvrY, ruvA* and *polB*) of the SOS DNA repair network. Gene expression levels were measured after irradiation of the DNA with UV light. Four experiments were done for various light intensities (Exp. 1 & 2: 5 Jm^{-2}, Exp. 3 & 4:20 Jm^{-2}) in each of which 50 samples were collected at 6 minutes interval for the above 8 genes [27]. For reconstructing the network we used only the first data set and preprocessed it by ignoring the sample at first time point (which was zero) and normalizing in the range [0,1].

We identified the regulators of each gene under the same algorithmic settings except we included the decay rate as a search parameter. The search ranges were as follows: $w_{ij} \in [-10.0, 10.0], \beta_i \in [-10.0, 10.0], \tau_i \in [0.0, 10.0]$ and $\lambda_i \in [0.0, 1.0]$. The reconstruction algorithm was repeated for 10 independent trials for each gene. In each run the reconstruction process achieved a very small fitness score indicating that the estimated model could match the target time course pretty well. Fig. 1 compares the target dynamics and the estimated model generated dynamics for some selected genes of the target network. From Fig. 1 it is evident that the estimated decoupled models for the genes captured the system response adequately.

[1] `http://www.weizmann.ac.il/mcb/UriAlon/`



(a) *uvrD* (b) *lexA*



(c) *umuDC* (d) *uvrA*

Fig. 1. Target and estimated dynamics for the SOS DNA repair network (y axis represents normalized expression level and x axis represents samples at six minute intervals)

Table 6. Predicted regulatory interactions in SOS network

	uvrD	lexA	umuDC	recA	uvrA	uvrY	ruvA	polB
uvrD	+	−						
lexA	+			+				
umuDC		−	+			+		
recA	+	−						
uvrA						+		
uvrY		−		+				
ruvA		−						+
polB	+	−		+				

+ (−) represents activation (repressive) control

However, the predicted regulations and parameter values were very different from run to run in our experiments. We applied Z-score analysis to identify the robust regulators from multiple trial runs. Based on our analysis we reconstructed the network structure presented in Table 6. As shown in Table 6, the essential regulatory interactions were identified by the proposed method. Inhibitory interactions of *lexA* gene on most of the other genes were identified correctly in addition to the activation of *lexA* by *recA*. Nevertheless, the prediction also includes a number of false positives which are either unknown regulations or the side effect of noise.

5 Conclusion

Large scale gene network inference has been always impeded by the computational requirement imposed by the underlying model. Recurrent neural network (RNN) model has been found to be a good candidate for estimating the GRN from gene expression data in terms of biological flexibility and computational feasibility. However, the model contains a large number of parameter which still makes the search very complicated for large scale networks. In this work, we investigated the decoupling of the model in which the regulators of each gene are identified independently in separate search instances. We used a natural computation based search algorithm, called differential evolution, for inferring the regulators of each gene. Experimenting with two artificial GRNs and analyzing a real gene expression profile, we verified the practicability of the proposed approach. Moreover, such decoupling mechanism not only makes the identification of large networks computationally feasible but also facilitates the immediate parallelization or distributed implementation of the reconstruction algorithm.

References

1. Das, S., Caragea, D., Welch, S.M., Hsu, W.H. (eds.): Handbook of Research on Computational Methodologies in Gene Regulatory Networks, 1st edn. Medical Information Science Reference, PA (2009)
2. Zhang, Y., Xuan, J., de los Reyes, B.G., Clarke, R., Ressom, H.W.: Reverse engineering module networks by PSO-RNN hybrid modeling. BMC Genomics 10(suppl. 1), S15 (2009)
3. Gardner, T.S., di Bernardo, D., Lorenz, D., Collins, J.J.: Inferring genetic networks and identifying compound mode of action via expression profiling. Science 301(5629), 102–105 (2003)
4. Friedman, N.: Inferring cellular networks using probabilistic graphical models. Science 303(5659), 799–805 (2004)
5. D'Haeseller, P., Liang, S., Somogyi, R.: Genetic network inference: from co-expression clustering to reverse engineering. Bioinformatics 16(8), 707–726 (2000)
6. Wahde, M., Hertz, J.: Coarse-grained reverse engineering of genetic regulatory networks. Biosystems 55(1-3), 129–136 (2000)
7. Ressom, H.W., Zhang, Y., Xuan, J., Wang, Y.J., Clarke, R.: Inference of gene regulatory networks from time course gene expression data using neural networks and swarm intelligence. In: IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology (CIBCB), pp. 435–442 (2006)
8. Wahde, M., Hertz, J.: Modeling genetic regulatory dynamics in neural development. Journal Computational Biology 8(4), 429–442 (2001)
9. Chiang, J.H., Chao, S.Y.: Modeling human cancer-related regulatory modules by GA-RNN hybrid algorithms. BMC Bioinformatics 8(91) (2007)

10. Xu, R., Wunsch II, D.C., Frank, R.L.: Inference of genetic regulatory networks with recurrent neural network models using particle swarm optimization. IEEE/ACM Transaction on Computational Biology and Bioinformatics 4(4), 681–692 (2007)
11. Noman, N., Palafox, L., Iba, H.: Inferring Genetic Networks with Recurrent Neural Network Model using Differential Evolution. In: Handbook of Bio-and Neuroinformatics - Part-C: Machine Learning Methods for Information Processing. Springer (2012)
12. Keedwell, E., Narayanan, A.: Discovering gene networks with a neural-genetic hybrid. IEEE/ACM Transaction on Computational Biology and Bioinformatics 2(3), 231–242 (2005)
13. Vohradský, J.: Neural model of the genetic network. The Journal of Biological Chemistry 276(39), 36168–36173 (2001)
14. Grimaldi, M., Visintainer, R., Jurman, G.: RegnANN: Reverse engineering gene networks using artificial neural networks. PLoS ONE 6(12), e28646 (2011)
15. Noman, N., Iba, H.: On the reconstruction of gene regulatory networks from noisy expression profiles. In: Proceedings of the World Congress on Computational Intelligence 2006, pp. 8712–8719 (July 2006)
16. Song, L., Kolar, M., Xing, E.P.: KELLER: Estimating time-varying interactions between genes. Bioinformatics 25(12), i128–i136 (2009)
17. Storn, R., Price, K.V.: Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4), 341–359 (1997)
18. Mondal, B.S., Sarkar, A.K., Hasan, M.M., Noman, N.: Reconstruction of gene regulatory networks using differential evolution. In: Proceedings of 13th International Conference on Computer and Information Technology (ICCIT 2010), pp. 440–445 (2010)
19. Arnone, M., Davidson, E.: The hardwiring of development: Organization and function of genomic regulatory systems. Development 124(10), 1851–1864 (1997)
20. Kikuchi, S., Tominaga, D., Arita, M., Takahashi, K., Tomita, M.: Dynamic modeling of genetic networks using genetic algorithm and S-sytem. Bioinformatics 19(5), 643–650 (2003)
21. Kimura, S., Ide, K., Kashihara, A., Kano, M., Hatakeyama, M., Masui, R., Nakagawa, N., Yokoyama, S., Kuramitsu, S., Konagaya, A.: Inference of S-system models of genetic networks using cooperative coevolutionary algorithm. Bioinformatics 21(7), 1154–1163 (2005)
22. Noman, N., Iba, H.: Reverse engineering genetic networks using evolutionary computation. Genome Informatics 16, 205–214 (2005)
23. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Springer, Heidelberg (2005)
24. Michel, B.: After 30 years of study, the bacterial SOS response still surprises us. PLoS Biology 3(7), e255 (2005)
25. Janion, C.: Some aspects of the SOS response aystem - a critical survey. Acta Biochimica Polonica 48(3), 599–610 (2001)
26. Little, J.W., Edmiston, S.H., Pacelli, L.Z., Mount, D.W.: Cleavage of the Escherichia coli lexA protein by the recA protease. Proceedings of National Academy of Science (PNAS) 77(6), 3225–3229 (1980)
27. Perrin, B.E., Ralaivola, L., Mazurie, A., Bottani, S., Mallet, J., d'Alché-Buc, F.: Gene networks inference using dynamic bayesian networks. Bioinformatics 19, 138–148 (2003)