Contents lists available at ScienceDirect

Computational Biology and Chemistry

journal homepage: www.elsevier.com/locate/compbiolchem

Using gene expression programming to infer gene regulatory networks from time-series data

Yongqing Zhang[a,*], Yifei Pu[a], Haisen Zhang[b], Yabo Su[a], Lifang Zhang[c], Jiliu Zhou[a]

[a] College of Computer Science, Sichuan University, Chengdu 610065, PR China
[b] College of Mathematic, Sichuan University, Chengdu 610065, PR China
[c] College of Chemistry, Sichuan University, Chengdu 610064, PR China

ARTICLE INFO

Article history:
Received 4 February 2013
Received in revised form
19 September 2013
Accepted 21 September 2013

Keywords:
Gene regulatory networks
Ordinary differential equation
Gene expression programming
Least mean square

ABSTRACT

Gene regulatory networks inference is currently a topic under heavy research in the systems biology field. In this paper, gene regulatory networks are inferred via evolutionary model based on time-series microarray data. A non-linear differential equation model is adopted. Gene expression programming (GEP) is applied to identify the structure of the model and least mean square (LMS) is used to optimize the parameters in ordinary differential equations (ODEs). The proposed work has been first verified by synthetic data with noise-free and noisy time-series data, respectively, and then its effectiveness is confirmed by three real time-series expression datasets. Finally, a gene regulatory network was constructed with 12 Yeast genes. Experimental results demonstrate that our model can improve the prediction accuracy of microarray time-series data effectively.

1. Introduction

The increasing availability of high-throughput measurements of transcripts has presented a golden opportunity to infer gene regulatory networks. Measuring the levels of gene expression in different conditions is useful in medical diagnosis, treatment, and drug design (Huang et al., 2010; Sun and Hurley, 2009). Many gene expression experiments produce time-series data with only a few time points owing to the high measurement costs. Accurate prediction of the behavior of gene regulatory networks (GRNs) will also speed up biotechnological projects; as such predictions are quicker and cheaper than lab experiments. Therefore, it is highly desired to infer the model of gene regulatory networks using gene expression time-series data. How to predict gene regulatory networks has become an important research area in bioinformatics.

Recently, many dynamic modeling of gene regulatory networks from time-series data has received more and more research interest (De Jong, 2002; Karlebach and Shamir, 2008), such as Boolean network (Akutsu et al., 1999; Bornholdt, 2008), dynamic Bayesian networks (Ghahramani, 1998; Murphy and Mian, 1999; Liu et al., 2006), neural networks (Lee and Yang, 2008), differential equations (De Jong, 2002; Chen et al., 1999; De Hoon et al., 2002; D'haeseleer et al., 1999), state-space model (Wu et al., 2004), stochastic model (Wang et al., 2008, 2010), and so no. A recent review to infer gene regulatory networks based on data integration in dynamical models can be seen in reference Hecker et al. (2009). The system of ordinary differential equations (ODEs) is a powerful and flexible model to describe complex relations, so many methods are proposed to infer genetic regulatory systems using ODEs. For example, Li et al. (2011) have proposed a new hybrid algorithm integrating ordinary differential equation models with local dynamic Bayesian network to infer gene regulatory network. Vilela et al. (2009) identified neutral biochemical network models from time-series data, combining Monte Carlo to optimize the parameters. Zhou et al. (2012) reconstructed GRN from time-series microarray data using stepwise multiple linear regression. Yang et al. (2012) proposed flexible neural tree model which is used for gene regulatory network reconstruction and time-series prediction from gene expression profiling. Unfortunately, most results reported on ODEs have been focused on fix structure of equations which describe the gene regulatory networks and the only one goal was to optimize parameters and coefficients. So it is the motivation of this paper to develop a system biology approach to determine the suitable form of equations which describe the network and to infer reverse engineer gene regulatory network from time-series data with higher accuracy and better scalability.

In our study, we cope with an arbitrary form in the right-hand side of the ODEs models. In order to identify the models, gene expression programming (GEP) is utilized to evolve the right-hand side of the ODEs from the observed time-series gene expression dataset. GEP is a new evolutionary algorithm which has good

* Corresponding author. +86 15208345900.
E-mail address: zhangyongqingscu@hotmail.com (Y. Zhang).

1476-9271/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compbiolchem.2013.09.004

performance to solve time-series prediction problem (Zuo et al., 2004). To decrease the complexity of the genetic network inference problem, the partitioning (Bongard and Lipson, 2007) is used in the process of identification of structure of system. Each ODE can be inferred separately and the research space reduces rapidly. In this paper, two synthetic time-series datasets obtained by E-cell system (Tomita et al., 1999) and three other real microarray datasets from Worm gene expression time-series dataset (Yeung et al., 2001), Human cell time-series dataset (Whitfield et al., 2002) and Yeast time-series dataset (Woolf and Wang, 2000; Schneider and Guarente, 1991) are used to test our method. Finally, a gene regulatory network was constructed with Yeast time-series dataset. Experiment results show that our method is capable of improving the prediction accuracy of microarray time-series data effectively.

2. Methods

2.1. Modeling gene regulation with ordinary differential equation

Ordinary differential equations (ODEs) is one of the most popular tools to model complex systems, which basic relationships are known between the system components. In the inverse problem, we often use ODEs to analysis the model from the observed time-series data.

To allow the flexibility of the model, we consider the following general form:

$$\frac{dx_i}{dt} = f_i(x_1, x_2, \ldots, x_n) \quad (i = 1, 2, 3, \ldots, n), \tag{1}$$

where x_i the state is variable and n is the number of the observed data time points.

In order to identify the system, GEP is used to evolve the ODEs from the observed time-series data. Although GEP could effectively find the suitable structures, it is sometimes difficult to optimize the parameters. So least mean square (LMS) (Ando et al., 2002) is employed to improve the effect of GEP.

2.2. GEP algorithm

2.2.1. Brief introduction of GEP

The GEP algorithm is described in detail in Ferreira (2001, 2006a). We give a brief introduction below.

The first step, how to define a problem by GEP, the encoding of the candidate solution and the definition of the fitness function. It is a most important and most difficult point in GEP. Each problem has specific the encoding and the fitness function. Adequate choices are the key factor for the success of the algorithm.

The next step, utilize the GEP algorithm itself including several stages. A basic flowchart of the algorithm is shown in Fig. 1.

The process starts with the random creation of an initial population of chromosomes. Then each chromosome is translated into an expression tree and each individual is evaluated by fitness function. The individuals are selected in the light of fitness function to reproduce with modification. If the termination criterion is not met, some of the chromosomes are selected and reproduced, resulting in offspring. The new chromosomes will replace the old ones producing a new generation. The process continues until the termination criterion is met or a certain number of iterations run.

Since GEP offers great potential to solve complex modeling and optimization problems, it has been used in many fields such as data and text mining (Zhou et al., 2003; Karakasis and Stafylopatis, 2006), classification (Duan et al., 2006; Karakasis and Stafylopatis, 2008; Duan et al., 2009), time-series analysis (Zuo et al., 2004), neural network design (Ferreira, 2006b) and various engineering application (Si et al., 2011).

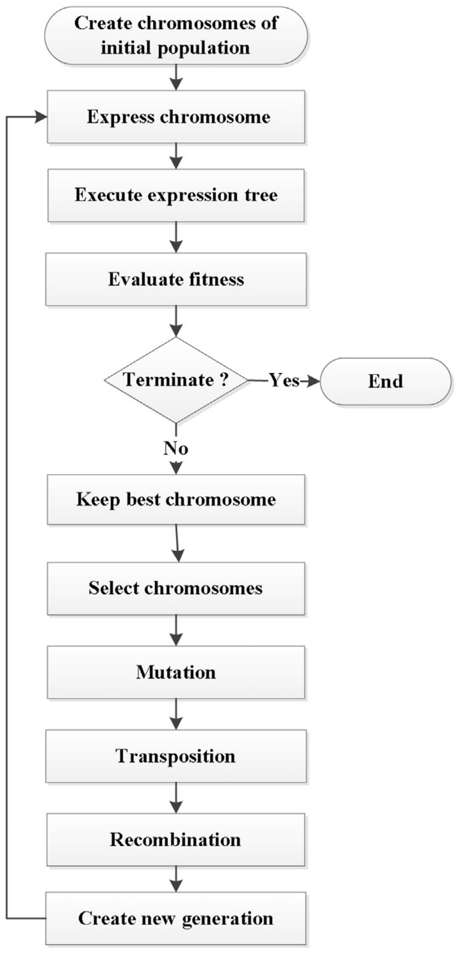

Fig. 1. The flowchart of basic GEP algorithm.

GEP and genetic programming (GP) are evolutionary algorithms. The fundamental difference between them resides in the nature of the individuals: in GEP the individuals are encoded as linear strings of fixed length which are afterwards expressed as nonlinear entities of different sizes and shapes; and in GP the individuals are nonlinear entities of different sizes and shapes. So that GEP provides new and efficient ways to program evolutionary computation (Ferreira, 2001).

2.2.2. Chromosome encoding

In GEP, the basic unit of an individual is called gene. The most distinctive feature of GEP is that each gene has access to a genotype and a corresponding phenotype: the genotype is a symbolic string of some fixed length, and the phenotype is the tree structure for the expression coded by that symbolic string. The symbolic string of a gene is composed of a head and a tail, both having fixed lengths. The head contains both function and term symbols, while the tail contains only term symbols. The function symbol represents a mathematical operator, such as addition, subtraction, multiplication, division, log and sin. The term symbol represents an attribute value. For each problem, the length of the head ($|head|$) and the length of the tail ($|tail|$) satisfy $|tail| = |head| \times (n-1) + 1$, where n is the maximum arity of functions under consideration. The head length ($|head|$) is determined by the user as the maximum number of functions in a gene; the length of a gene ($|head| + |tail|$) remains unchanged in the middle of an execution of a given GEP algorithm. In GEP, an individual problem may involve one or more genes to encode a candidate solution. For multiple genes in an individual

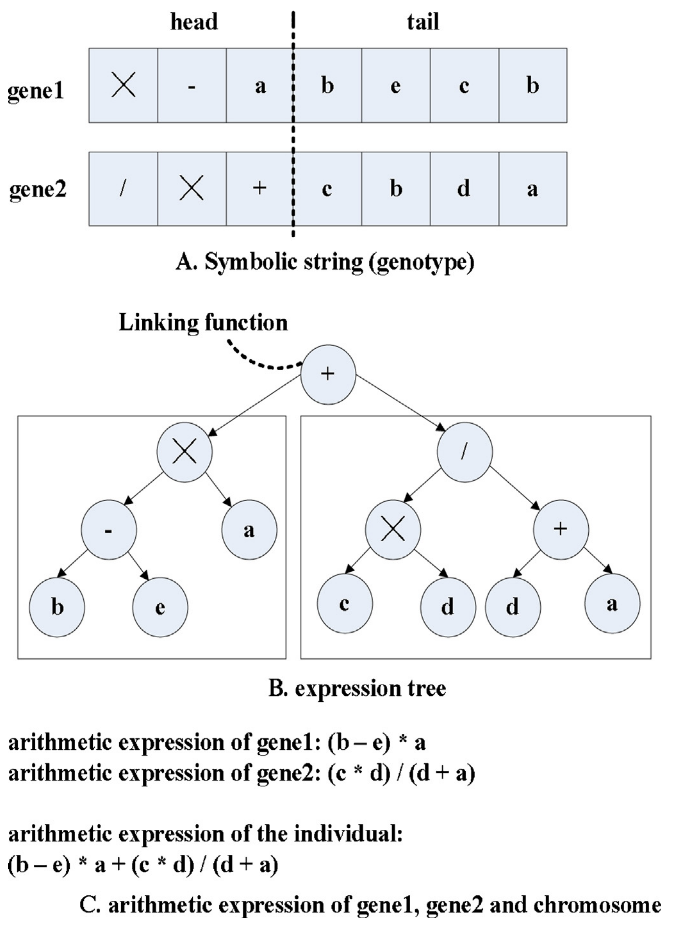
head
tail
gene1
gene2
A. Symbolic string (genotype)
Linking function
B. expression tree
arithmetic expression of gene1: (b – e) * a
arithmetic expression of gene2: (c * d) / (d + a)
arithmetic expression of the individual:
(b – e) * a + (c * d) / (d + a)
C. arithmetic expression of gene1, gene2 and chromosome

Fig. 2. The genotype, phenotype and arithmetic expression of a chromosome.

problem, the genes are connected by the linking function, such as "+".

Suppose the function set is {+, −, ×, /}, the term set is {*a*, *b*, *c*, *d*, *e*}, the linking function is +. Fig. 2 illustrates the expression tree and corresponding arithmetic expression of a 2-gene individual problem. The two genes in the individual have the same head length (3) and total length (7); but their expression trees (phenotype) are different, and so their arithmetic expressions are different. For *gene*1, the coding region is the first 5 symbols (so the expression does not contain the "*cb*" at the end). For *gene*2, the coding region is the whole string. The linking functions (+) connects the expression trees of *gene*1and *gene*2 together to make up the expression tree of the individual.

2.2.3. Fitness function definition

In this paper, the fitness function of each individual program is defined as the sum of the square error for the degree of the equations:

$$\text{fitness}(i) = \sum_{j=1}^{T} (x_i'(t0 + j\Delta t) - xi(t0 + j\Delta t))^2 \tag{2}$$

In which t_0 is the starting time, Δt is the step size, T is the number of the data time points, $xi(t0 + j\Delta t)$ is the given target time-series of *i*-th sample ($j = 1, 2, \ldots, T$) and $x'_i(t0 + j\Delta t)$ is the time-series acquired by calculating the ODEs.

Obviously, the lower the fitness is, the better is the individual. In this work, all these time-series data are calculated by the fourth-order Runge–Kutta method (Butcher, 1987).

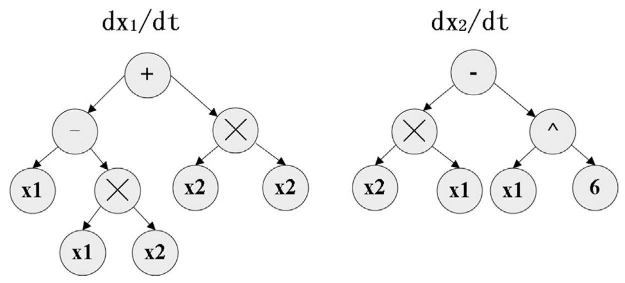

Fig. 3. An example of ODEs model by GEP.

2.2.4. Reproduction

There are three kinds of genetic operators for reproduction, namely mutation, transposition, and recombination. Mutation operator and transposition operate on a single individual, and recombination operator takes place on two individuals. A mutation randomly changes a symbol in a chromosome into another symbol, as long as it does not introduce function symbol in the tail. Transposition rearranges a part of the chromosome to another location in the same chromosome. Recombination exchanges some elements between two randomly chosen chromosomes to form two new chromosomes. All new chromosomes created by GEP-style modifications are syntactically correct candidate solutions. This feature distinguishes GEP from GP, where some genetic modifications (such as mutation) can produce invalid solutions. More details can be found in Ferreira (2001). The individuals of each new generation undergo the same processes of evaluation, selection and reproduction with modification as in the preceding generation. The evolution process repeats until some stop condition (given in terms of number generations, quality of solutions, and so on) is satisfied.

2.3. Inference of the ODEs model using GEP

To identify a system of differential equations, we encode right-hand sides of ODEs into a GEP individual. For example, consider the two trees in Fig. 3. This shows the following system of ODEs:

$$\begin{cases} \dfrac{dx_1}{dt} = ax_1 - bx_1x_2 + cx_2^2 \\ \dfrac{dx_2}{dt} = dx_1x_2 - ex_1^6 \end{cases} \tag{3}$$

where the *coefficients a*, *b*, *c*, *d*, *e* are derived by LMS described in Section 2.4 in this paper.

We infer the ODEs model with partitioning method (Bongard and Lipson, 2007). Partitioning allows the algorithm to synthesize equations describing each variable separately, even though their behaviors may be coupled. When using partitioning, a candidate equation for a signal variable is integrated by substituting references to other variables with observed time-series data. This allows us to infer the structure of systems comprising more variables and higher degree of coupling than were inferred by other methods (McKinney et al., 2006).

2.4. Parameter optimization of models using least mean square (LMS)

To find the optimal coefficients of a GEP individual, the LMS method is employed below (Ando et al., 2002). Assume that we want to acquire the approximate expression in the following form. It means each expression of differential equations by using GEP:

$$y(x_1, x_2, \ldots, x_L) = \sum_{k=1}^{M} a_k F_k(x_1, x_2, \ldots, x_L) \tag{4}$$

where $F_k(x_1, x_2, \ldots, x_L)$ is the basis function, $x_1, x_2, \ldots, x_L$ are the independent variables, $y(x_1, x_2, \ldots, x_L)$ is the dependent variable, and M is the number of the basis function. Let a be the vector of coefficients, i.e. $(a_1, a_2, \ldots, a_M)$. Then, our purpose is to minimize χ^2 described in (5) to acquire a. χ^2 is the minimum square error by the real value and the predicted value of gene expression time-series data.(5)$\chi^2 = \sum_{i=1}^{N}\left(y(i) - \sum_{k=1}^{M} a_k F_k(x_1, x_2, \ldots, x_L)\right)^2$ where x_1, x_2, …, x_L and $y(i)$ are data given for the LMS method and N is the number of data points. Let b be the vector of ($y(1)$, $y(2)$, …, $y(N)$) and A be the $N \times M$ matrix described below:(6)

$$\begin{pmatrix} F_1(x_1(1), x_2(1), \ldots, x_L(1)) & \ldots & \ldots & F_M(x_1(1), x_2(1), \ldots, x_L(1)) \\ F_1(x_1(2), x_2(2), \ldots, x_L(2)) & \ldots & \ldots & F_M(x_1(2), x_2(2), \ldots, x_L(2)) \\ \ldots & \ldots & \ldots & \ldots \\ F_1(x_1(N), x_2(N), \ldots, x_L(N)) & \ldots & \ldots & F_M(x_1(N), x_2(N), \ldots, x_L(N)) \end{pmatrix}$$

Then, (7) should be satisfied to minimize χ^2.

$$(A^T \cdot A) \cdot a = A^T \cdot b \qquad (7)$$

Thus, a can be acquired by solving this equation.

The coefficient in the approximate expressions of the right-hand sides of the equations can be derived by using A and b ($y(1)$, $y(2)$, …, $y(N)$) acquired above.

3. Experimental results and discussion

In this paper, to confirm the effectiveness of the proposed approach, at first, it has been applied to a synthetic genetic network inference problem. For this, we have considered both the noise-free and noisy data. Even with the presence of noise, the proposed approach has successfully reverse engineer the network from the synthetic data. Afterwards, this approach is tested using three real-world gene expression time-series datasets. Specifically, the accuracy of inferring gene regulatory networks has been tested with five different cases:

1. Inferring gene network from synthetic gene expression data without noise.
2. Inferring gene network from synthetic gene expression data with 5% and 10% noise.
3. Inferring gene network from Worm data with short time-series.
4. Inferring gene network from Human cell data with longer time-series.
5. Inferring gene network from Yeast time-series data and constructing gene interactions with 12 genes.

In addition, to measure the modeling results, we define the average mean square error of training dataset (MSE_Training) and testing dataset (MSE_Testing) of variable x as

$$\text{MSE\_Training} = \frac{1}{m}\sum_{j=1}^{m}(x'_j - xj)^2$$
$$\text{MSE\_Testing} = \frac{1}{\tau}\sum_{j=m+1}^{m+\tau}\left(x'_j - xj\right)^2 \qquad (8)$$

where x_j is the observed value of x at the time T_j, x'_j is the training value and the testing value of x, m and τ the number of data points to be training and to be testing respectively.

Experimental parameters are summarized in Table 1. Function and terminal sets F and T are describe as follows,

$$F = \{+, -, \times\}$$

$$T = \left\{x_1, x_1^2, x_2, x_2^2, \ldots, x_n, x_n^2\right\}$$

Table 1
GEP parameters for experiments.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4	Exp. 5
Population size	50	50	50	50	50
Gene size	4	4	4	4	4
Generation	100	100	100	100	100
Crossover rate	0.4	0.4	0.4	0.4	0.4
Mutation rate	0.1	0.1	0.1	0.1	0.1
Step size	0.05	0.05	0.05	0.05	0.05
Data points	49	49	17	48	17

3.1. Inference of the synthetic data without noise

In this part, we use data of a metabolic network, called the E-cell system, which consists of three substances. E-cell simulation environment is a software package for cellular and biochemical modeling and simulation (Tomita et al., 1999). This network can be approximated as:

$$\frac{dx_1}{dt} = -k_1 x_1 \bullet x_3$$
$$\frac{dx_2}{dt} = k_1 x_1 \bullet x_3 - k_2 x_2 \qquad (9)$$
$$\frac{dx_3}{dt} = -k_1 x_1 \bullet x_3 + k_2 x_2$$

Note that the parameters k_1, k_2 and k_3 are unknown for the simulation experiment. In our experiment, the time-series of the synthetic data is from 0 to 48, including 49 time points. From them, the first 44 points are used to build ODE models and the next 5 points are used to evaluate the prediction result of the model. Experimental parameters are shown in Table 1.

The best kinetic model we have obtained in 10 runs is

$$\frac{dx_1}{dt} = -10.2015 x_1 \cdot x_3$$
$$\frac{dx_2}{dt} = 9.204 x_1 \bullet x_3 - 17.309 x_2 \qquad (10)$$
$$\frac{dx_3}{dt} = -9.588 x_1 \bullet x_3 + 17.401 x_2$$

The time-series data generated by Eq. (10) is shown in Fig. 4 along with that of the target. The MSE_Training of 44 points was (x_1, x_2, x_3) = (0.00245, 0.00181, 0.00179), the MSE_Testing of 5 points was (x_1, x_2, x_3) = (0.00315, 0.00260, 0.00274). As can be seen, the acquired time-series data is quite close to the target one.

Our algorithm has been implemented in Java programming language. The time required for solving a typical run of the associated GRNs problem is approximately 1.0 min in a PC with 3.1 GHz Intel

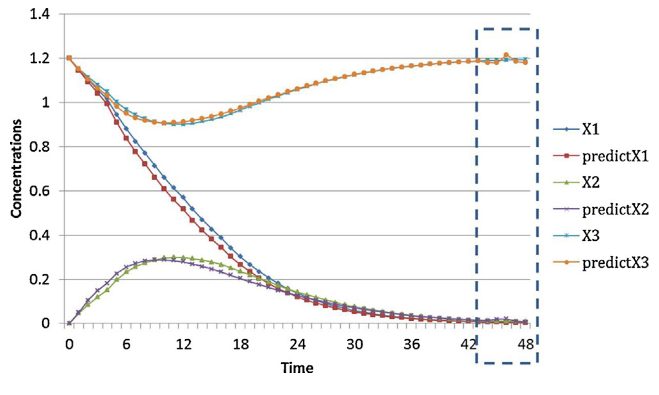

Fig. 4. Dynamic of expression level of the synthetic data without noise, the last 5 points dataset is used for testing the model.

Table 2
MSE of Synthetic data prediction with noise.

	Dataset	X_1	X_2	X_3
Synthetic data without noise	Training	0.00245	0.00181	0.00179
	Testing	0.00427	0.00301	0.00570
Synthetic data with 5% noise	Training	0.00351	0.00352	0.00590
	Testing	0.00396	0.00511	0.00603
Synthetic data with 10% noise	Training	0.00360	0.00501	0.00797
	Testing	0.00407	0.00772	0.00957

Core i5-2400 processor and 4.0 G of RAM. The program has been run with the same experimental setup for 10 runs.

3.2. Inference of the synthetic data with noise

To test the robustness of our method to the real noisy world, we conducted the E-cell synthetic data with noise-add datasets. 5% and 10% random Gaussian noises were added to the target time-series data. Both of these experiments are also conducted with 10 runs using the similar setup described in the previous section. We also take the first 44 points to build ODE models and the next 5 points to evaluate the prediction result of the model. The MSE values averaged by GEP and LMS over 10 runs are shown in Table 2. From Table 2, it is shown that the result of MSE will get worse with the higher noise in the datasets, but the trend of MSE from increasing is not at all obvious. The observed and estimated time-series dynamics data of X_2 in three different datasets obtained by our method is shown in Fig. 5. We can observe that the proposed method worked effectively to acquire the better individual with noisy environments. The computational time of the proposed work for inferring this gene regulatory network is also small. It takes about 1.5 and 1.8 min on the above-mentioned PC configuration, respectively.

3.3. Inference of the Worm time-series data

The third dataset is from the Worm gene expression time-series data (Yeung et al., 2001). It includes 237 genes expressed 17 equally spaced time points and the observation interval is 10 min. From them, the first 14 points are used as modeling samples and the next 3 points are used as test samples to evaluate the prediction results of the model. We select the first 4 genes expression time-series for our purpose (Wang et al., 2009). Experimental parameters are shown in Table 1.

The set of time-series data began from randomly generated initial values and was obtained by solving the set of differential equations on the targeted model. In this work, we apply the GEP to evolve the right-hand side of the equation. 10 runs were carried out. By applying our method, we have acquired the following equations (9):

$$
\begin{aligned}
\frac{dx_1}{dt} &= x_1^2 - x_1 + x_1 \cdot x_4 \\
\frac{dx_2}{dt} &= x_1^2 + x_4 \\
\frac{dx_3}{dt} &= x_1 + x_3 + x_3^2 \\
\frac{dx_4}{dt} &= x_1^2 + x_1 + x_4^2
\end{aligned}
\tag{11}
$$

And then, we use LMS to optimize the parameters in ODEs, we have acquired the following equations in a typical run:

$$
\begin{aligned}
\frac{dx_1}{dt} &= -2.106x_1^2 - 1.763x_1 + 5.565x_1 \cdot x_4 \\
\frac{dx_2}{dt} &= -0.651x_1^2 + 0.5132x_4 \\
\frac{dx_3}{dt} &= 3.830x_1 - 0.546x_3 + 2.866x_3^2 \\
\frac{dx_4}{dt} &= -0.076x_1^2 - 2.385x_4^2 - 2.072x_1
\end{aligned}
\tag{12}
$$

The performance of GEP and GEP + LMS in worm gene regulation prediction is given in Table 3. From Table 3, we can see that GEP and GEP + LMS can achieve good result in predicting worm dataset, and the LMS is an effective method to optimize parameter in ODEs model. The Fig. 6 demonstrates the trendy of gene regulatory network. As can be seen, the acquired time-series data is quite close to the target one. Thus, we can confirm the research become more effective by using GEP along with LMS method. The computational time of the proposed work for inferring this gene regulatory network takes about 1.2 min on the above-mentioned PC configuration.

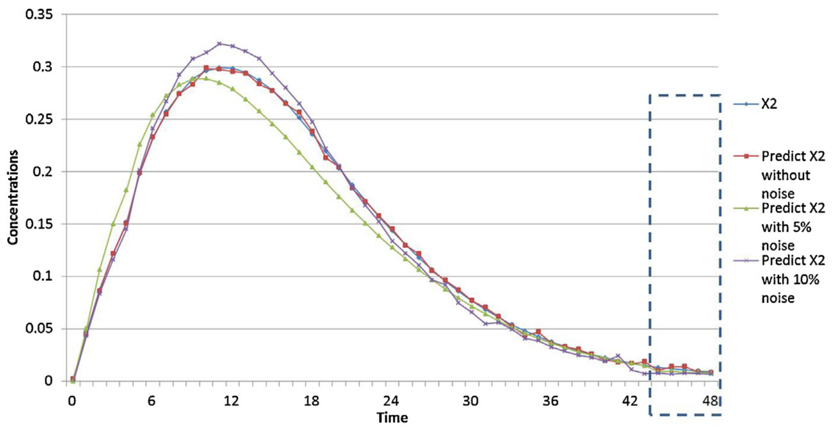

Fig. 5. Dynamic of expression level of the synthetic data without noise, with 5% noise and with 10% noise respectively, compares with X_2; the last 5 points dataset is used for testing the model.

Table 3
MSE of Worm gene regulation prediction.

Number of time points	Method	Dataset	X_1	X_2	X_3	X_4
17 time points	GEP	Training	0.00063	0.00161	0.00058	0.00209
		Testing	0.00102	0.00215	0.00098	0.00285
	GEP + LMS	Training	0.00001	0.00006	0.00012	0.00006
		Testing	0.00008	0.00013	0.00040	0.00019

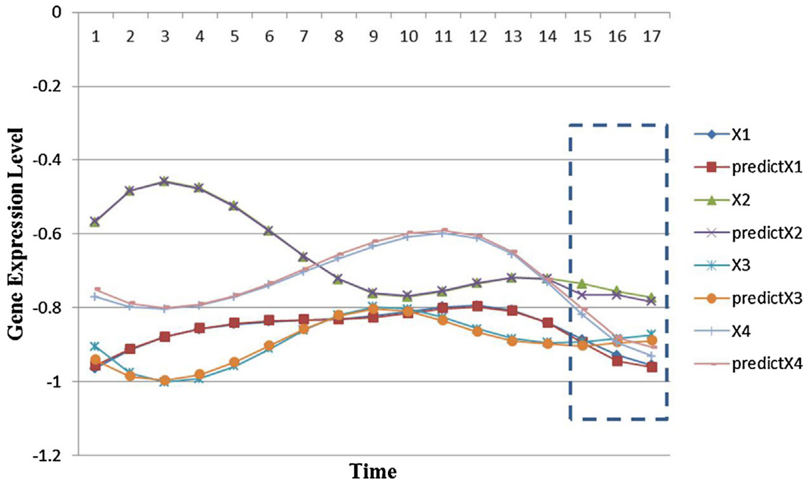

Fig. 6. Dynamic of expression level of the four genes in Worm by 17 time points, the prediction result is by GEP + LMS, the last 3 points dataset is used for testing the model.

3.4. Inference of the Human cell time-series data

The fourth dataset is from the Human cell time-series data. The dataset was first generated by Whitfield et al. (2002) in order to identify genes that are periodically expressed in cell cycles. It consists of 1099 genes expressed 48 equally spaced time points. We select the first 5 genes expression time-series data for our purpose (Wang et al., 2009). The whole dataset was downloaded from http://genome-www.stanford.edu/Human-CellCycle/Hela/data/dataPlusScores_all5.txt.

Experimental parameters are shown in Table 1. In order to conduct a comparative experiment with different data points to the influence of the result, we choose 15, 25, 35, 48 time points in the experimental. In each experimental, we take the last 5 points to predict the model, and other point to build ODE models, such that 10, 20, 30, 43 points respectively. The average mean square errors of training dataset and testing dataset are listed in Table 4. We can see that the more time points, the bigger MSE of prediction, but the MSE is acceptable in most of the cases. Many gene expression experiments produce time-series data with only a few time points owing to the high measurement costs. So our method could use to model the gene regulatory network by gene time-series data. Figs. 7 and 8 show the prediction of gene expression patterns of 15 time points and 35 time points. We can see that the prediction of Fig. 7 is more accurate than the prediction of Fig. 8. The computational time of the proposed work for inferring this gene regulatory network with 48 time points takes about 2.2 min on the above-mentioned PC configuration.

By applying our method, we have acquired the following ODEs for 15 time points:

$$\begin{aligned}
\frac{dx_1}{dt} &= x_1^2 + x_2 \cdot x_3^4 + x_2 \cdot x_5 + x_3 + x_5 \\
\frac{dx_2}{dt} &= x_2^2 + x_2 + x_3 \cdot x_5 + x_4 \cdot x_5^2 + x_5^2 \\
\frac{dx_3}{dt} &= x_1^2 + x_1^2 \cdot x_3 + x_4 + x_5 \\
\frac{dx_4}{dt} &= x_2 \cdot x_3 + x_3 \cdot x_4 \cdot x_5^2 + x_5 \\
\frac{dx_5}{dt} &= x_1^2 + x_1 \cdot x_2^2 + x_3 \cdot x_2 + x_3 \cdot x_4^2 + x_4 \cdot x_5
\end{aligned} \tag{13}$$

Table 4
MSE of gene regulation prediction with four differential cases.

Number of time points	Dataset	X_1	X_2	X_3	X_4	X_5
15 time points	Training	0.00018	0.00036	0.00048	0.00069	0.00011
	Testing	0.00022	0.00011	0.00033	0.00245	0.00018
25 time points	Training	0.00330	0.00109	0.00058	0.00028	0.00140
	Testing	0.00837	0.00557	0.00198	0.00087	0.00246
35 time points	Training	0.00825	0.00207	0.05707	0.00130	0.01712
	Testing	0.02340	0.00218	0.04158	0.00212	0.03631
48 time points	Training	0.00849	0.00217	0.05973	0.00620	0.01689
	Testing	0.01083	0.00626	0.03431	0.01103	0.03608

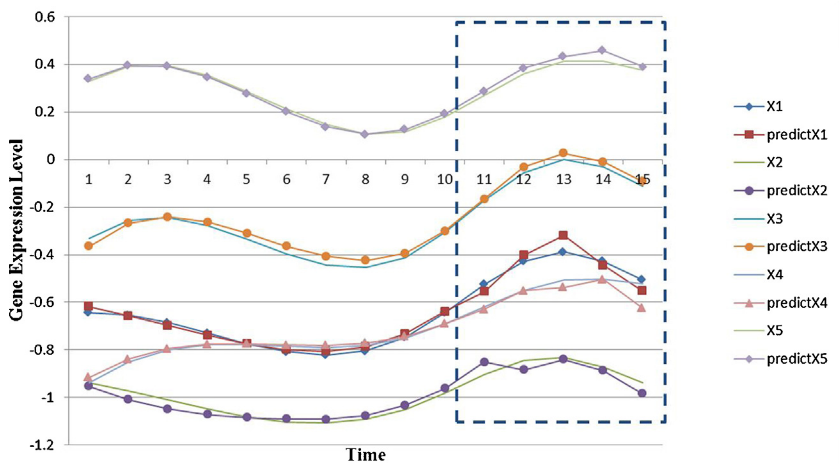

Fig. 7. Dynamic of expression level of the five genes in Human cell by 15 times and the prediction result is by GEP + LMS; the last 5 points dataset is used for testing the model.

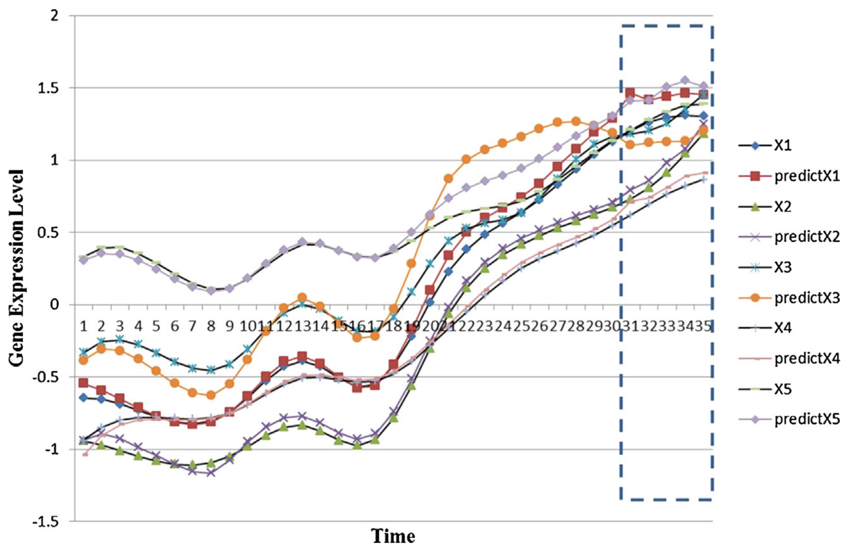

Fig. 8. Dynamic of expression level of the five genes in Human cell by 35 times and the prediction result is by GEP + LMS; the last 5 points dataset is used for testing the model.

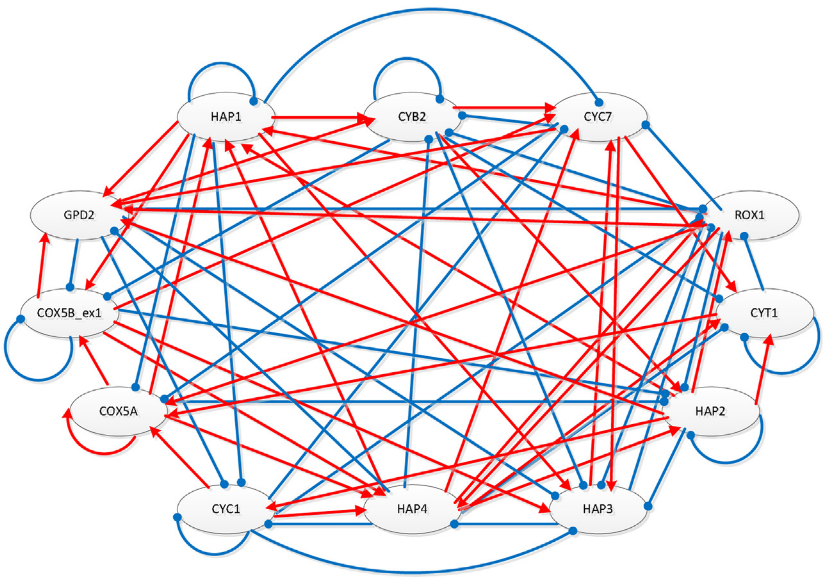

Fig. 9. Inferred gene interaction network for 12 Yeast regulatory genes. Each node represents a gene and the presence of an edge between the two nodes represents the interaction between the two genes. Symbols "→" and "—●", shown by red and blue edges, illustrate activator and repressor interactions, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

And then, we use LMS to optimize the parameters in ODEs model, we have acquired the following equations in a typical run:

$$\begin{aligned}
\frac{dx_1}{dt} &= 0.921x_1^2 - 11.353x_2 \cdot x_3^4 + 2.266x_2 \cdot x_5 - 14.657x_3 + 2.313xdx_2 \\
\frac{dx_2}{dt} &= -1.158x_2^2 - 0.639x_2 - 0.129x_3 \cdot x_5 + 1.579x_4 \cdot x_5^2 + 0.213x_5^2 \\
\frac{dx_3}{dt} &= 5.388x_1^2 + 3.282x_1^2 \cdot x_3 + 2.663x_4 - 0.540x_5 \\
\frac{dx_4}{dt} &= -0.527x_2 \cdot x_3 - 1.597x_3 \cdot x_4 \cdot x_5^2 + 0.677x_5 \\
\frac{dx_5}{dt} &= 9.286x_1^2 + 5.051x_1 \cdot x_2^2 - 3.439x_2 \cdot x_3 - 0.031x_3 \cdot x_4^2 + 2.459x_4 \cdot x_5
\end{aligned} \tag{14}$$

3.5. Inference of the Yeast time-series data

In this part, to test the performance of reconstruction of our method in the real microarray data, we use the time-series gene expression data from Yeast protein synthesis. The measurement data is originally from http://www.wanghaixin.com/biowq2007.html, where related references are also available. The data has 17 equally spaced time points and the observation interval is 10 min. From them, the first 14 points are used as modeling samples and the next 3 points are used as test samples to evaluate the prediction results of the model. Here, the data for 12 genes (HAP1, CYB2, CYC7, ROX1, CYT1, HAP2, HAP3, HAP4, CYC1, COX5A, COX5B_ex1, and GPD2) are picked because the relations among them have been revealed by biological experiments (Woolf and Wang, 2000; Schneider and Guarente, 1991). The states of 12 genes are represented by $x_1, x_2, x_3, \ldots, x_{12}$, respectively. The same set of genes were used by Qian et al. (2008) and Yang et al. (2012) and their method successfully inferred the regulatory network of the selected genes. Experimental parameters are shown in Table 1.

The following model is obtained by the proposed algorithm:

$$\begin{aligned}
\frac{dx_1}{dt} &= 0.1659x_{10} + 0.0997x_4^4 - 1.0313x_1 + 0.2625x_8 \cdot x_6 \\
\frac{dx_2}{dt} &= -0.2833x_2 \cdot x_3 + 1.1193x_1 - 0.5501x_4 \cdot x_8 + 0.8241x_{12} \\
\frac{dx_3}{dt} &= -0.388x_1 + 0.5849x_2 \cdot x_8 - 0.5219x_4^3 + 0.2123x_7 \cdot x_{11} - 0.9605x_3 \cdot x_9 \\
\frac{dx_4}{dt} &= -0.6562x_4 \cdot x_5 - 0.5022x_7 \cdot x9 + 1.4782x_6 \cdot x_8 8 - 0.9084x_9 \cdot x_{12} \\
\frac{dx_5}{dt} &= -0.2554x_5 \cdot x_8 + 0.2762x_3 \cdot x_6 \cdot x_8 - 1.2979x_2 \cdot x_5 \\
\frac{dx_6}{dt} &= 0.601x_2 \cdot x_8 - 0.567x_4 \cdot x_{11} - 0.3157x_6 \cdot x_{10} \\
\frac{dx_7}{dt} &= 0.4107x_1 - 0.41x_2 \cdot x_6 - 0.4036x_4 \cdot x_{12} - 0.2615x_2 \cdot x_9 + 0.1977x_3 \cdot x_{11} \\
\frac{dx_8}{dt} &= -0.1701x_7 \cdot x_8 + 1.0336x_8 + 0.0632x_4 \cdot x_{10} + 0.1396x_9 \cdot x_{11} \\
\frac{dx_9}{dt} &= -0.1624x_1 + 0.2601x_4 \cdot x_6 - 0.4585x_5 \cdot x_{12} - 0.4389x_8 \cdot x_9 \\
\frac{dx_{10}}{dt} &= 0.8247x_{10} - 0.0778x_1 \cdot x_3 + 0.0551x_4 \cdot x_5 + 0.1357x_9 \\
\frac{dx_{11}}{dt} &= 0.0549x_{10} - 0.2786x_2 \cdot x_{12} - 0.1711x_3 \cdot x_{11} + 0.512x_1 \\
\frac{dx_{12}}{dt} &= -0.4774x_8^2 + 0.578x_6 \cdot x_{11} + 0.2804x_4 + 1.3163x_3 \cdot x_1^2
\end{aligned} \tag{15}$$

The detailed interactions among the 12 genes deduced from the obtained model are shown in Fig. 9 which illustrates the extracted genetic network of activator and repressor interactions. From Fig. 9, we can see that HAP1 represses CYC7, and CYB2 activates CYC7. It is also observed that HAP1 activates COX5B_ex1 and HAP2. These relationships are in agreement with the biological experimental findings in Woolf and Wang (2000) and Schneider and Guarente (1991). The computational time of the proposed work for inferring this gene regulatory network is much longer. It takes about 51 min on the above-mentioned PC configuration. The computational time of this approach is considerably less time for small-scale gene regulatory networks, but for large-scale of gene regulatory networks the computational time is more.

4. Conclusion

In conclusion, we have developed a new algorithm, ordinary differential equation integrated gene expression programming and least mean square, for inference of gene regulatory networks from time-series data. Our method has two advantages: (1) using GEP method could succeed in creating the gene regulatory networks of ODEs model, which are very close to the targeted system; (2) with partitioning, we can acquire the best model very fast, and each node of the genetic regulatory network can be inferred separately. The proposed method has been first verified by synthetic data with noise-free and noisy time-series data, respectively, and then its effectiveness is confirmed by three real time-series expression datasets. In all the cases, even with the presence of noise, the results demonstrate that the gene regulatory model predicted by our method is both accurate and stable.

Acknowledgements

We thank the anonymous reviewers for their constructive comments and criticisms that helped a lot in improving the quality of the paper. We also thank Dr. Yong Cong for useful discussions and suggestions.

This research was supported by the National Natural Science Foundation of China (60972131), Scientific Research Foundation for Returned Scholars, Ministry of Education of China (20111139), Science and Technology Support Foundation of Sichuan Province (2011GZ0201), Science and Technology Innovation Youth Foundation of Sichuan Province (2011-014) and Outstanding Youth Teacher Foundation of Sichuan University.

References

Akutsu, T., et al., 1999. Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. In: Pacific Symposium on Biocomputing, pp. 17–28.

Ando, S., et al., 2002. Evolutionary modeling and inference of gene network. Information Sciences 145, 237–259.

Bongard, J., Lipson, H., 2007. Automated reverse engineering of nonlinear dynamical systems. Proceedings of the National Academy of Sciences 104, 9943–9948.

Bornholdt, S., 2008. Boolean network models of cellular regulation: prospects and limitations. Journal of the Royal Society Interface 5, S85–S94.

Butcher, J.C., 1987. The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods:. Wiley-Interscience.

Chen, T., et al., 1999. Modeling gene expression with differential equations. In: Pacific Symposium on Biocomputing, p. 4.

D'haeseleer, P., et al., 1999. Linear modeling of mRNA expression levels during CNS development and injury. In: Pacific Symposium on Biocomputing, pp. 41–52.

De Hoon, M., et al., 2002. Inferring gene regulatory networks from time-ordered gene expression data of *Bacillus subtilis* using differential equations. In: Biocomputing 2003: Proceedings of the Pacific Symposium, pp. 17–28.

De Jong, H., 2002. Modeling and simulation of genetic regulatory systems: a literature review. Journal of Computational Biology 9, 67–103.

Duan, L., et al., 2006. Distance guided classification with gene expression programming. Advanced Data Mining and Applications, 239–246.

Duan, L., et al., 2009. Mining class contrast functions by gene expression programming. Advanced Data Mining and Applications, 116–127.

Ferreira, C., 2001. Gene expression programming: A new adaptive algorithm for solving problems. arXiv preprint cs/0102027.

Ferreira, C., 2006a. Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence (Studies in Computational Intelligence). Springer-Verlag Inc., New York.

Ferreira, C., 2006b. Designing neural networks using gene expression programming. Applied Soft Computing Technologies: The Challenge of Complexity, 517–535.

Ghahramani, Z., 1998. Learning dynamic Bayesian networks. Adaptive Processing of Sequences and Data Structures, 168–197.

Hecker, M., et al., 2009. Gene regulatory network inference: data integration in dynamic models—A. Biosystems 96, 86–103.

Huang, H., et al., 2010. Bayesian approach to transforming public gene expression repositories into disease diagnosis databases. Proceedings of the National Academy of Sciences 107, 6823–6828.

Karakasis, V.K., Stafylopatis, A., 2006. Data mining based on gene expression programming and clonal selection. In: IEEE Congress on Evolutionary Computation. CEC 2006, pp. 514–521.

Karakasis, V.K., Stafylopatis, A., 2008. Efficient evolution of accurate classification rules using a combination of gene expression programming and clonal selection. IEEE Transactions on Evolutionary Computation 12, 662–678.

Karlebach, G., Shamir, R., 2008. Modelling and analysis of gene regulatory networks. Nature Reviews Molecular Cell Biology 9, 770–780.

Lee, W.P., Yang, K.C., 2008. A clustering-based approach for inferring recurrent neural networks as gene regulatory networks. Neurocomputing 71, 600–610.

Li, Z., et al., 2011. Large-scale dynamic gene regulatory network inference combining differential equation models with local dynamic Bayesian network analysis. Bioinformatics 27, 2686–2691.

Liu, T.F., et al., 2006. Model gene network by semi-fixed Bayesian network. Expert Systems with Applications 30, 42–49.

McKinney, B., et al., 2006. Hybrid grammar-based approach to nonlinear dynamical system identification from biological time series. Physical Review E 73, 021912.

Murphy, K., Mian, S., 1999. Modelling gene expression data using dynamic Bayesian networks, Technical report. In: Computer Science Division, University of California, Berkeley, CA.

Qian, L., et al., 2008. Inference of noisy nonlinear differential equation models for gene regulatory networks using genetic programming and kalman filtering. IEEE Transactions on Signal Processing 56, 3327–3339.

Schneider, J.C., Guarente, L., 1991. Regulation of the yeast CYT1 gene encoding cytochrome c1 by HAP1 and HAP2/3/4. Molecular and Cellular Biology 11, 4934–4942.

Si, H., et al., 2011. Study of human dopamine sulfotransferases based on gene expression programming. Chemical Biology and Drug Design 78, 370–377.

Sun, D., Hurley, L.H., 2009. The importance of negative superhelicity in inducing the formation of G-quadruplex and i-motif structures in the c-Myc promoter: implications for drug targeting and control of gene expression. Journal of Medicinal Chemistry 52, 2863–2874.

Tomita, M., et al., 1999. E-CELL: software environment for whole-cell simulation. Bioinformatics 15, 72–84.

Vilela, M., et al., 2009. Identification of neutral biochemical network models from time series data. BMC Systems Biology 3, 47.

Wang, Z., et al., 2008. On delayed genetic regulatory networks with polytopic uncertainties: robust stability analysis. IEEE Transactions on NanoBioscience 7, 154–163.

Wang, Z., et al., 2009. An extended Kalman filtering approach to modeling nonlinear dynamic gene regulatory networks via short gene expression time series. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 6, 410–419.

Wang, Y., et al., 2010. On robust stability of stochastic genetic regulatory networks with time delays: a delay fractioning approach. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 40, 729–740.

Whitfield, M.L., et al., 2002. Identification of genes periodically expressed in the human cell cycle and their expression in tumors. Molecular Biology of the Cell 13, 1977–2000.

Woolf, P.J., Wang, Y., 2000. A fuzzy logic approach to analyzing gene expression data. Physiological Genomics 3, 9–15.

Wu, F.X., et al., 2004. Modeling gene expression from microarray expression data with state-space equations. In: Pacific Symposium on Biocomputing, pp. 581–592.

Yang, B., et al., 2012. Reverse engineering of gene regulatory networks using flexible neural tree models. Neurocomputing.

Yeung, K.Y., et al., 2001. Model-based clustering and data transformations for gene expression data. Bioinformatics 17, 977–987.

Zhou, C., et al., 2003. Evolving accurate and compact classification rules with gene expression programming. IEEE Transactions on Evolutionary Computation 7, 519–531.

Zhou, Y., et al., 2012. Data simulation and regulatory network reconstruction from time-series microarray data using stepwise multiple linear regression. Network Modeling and Analysis in Health Informatics and Bioinformatics, 1–15.

Zuo, J., et al., 2004. Time series prediction based on gene expression programming. Advances in Web-Age Information Management, 55–64.