The Third International Symposium on Optimization and Systems Biology (OSB'09)
Zhangjiajie, China, September 20–22, 2009
Copyright © 2009 ORSC & APORC, pp. 132–141

Evolving Additive Trees for Modeling Biochemical Systems

Yuehui Chen* Bin Yang Yaou Zhao
Qingfang Meng

Computational Intelligence Lab, School of Information Science and Engineering,
University of Jinan, Shandong, Jinan 250022, P.R. China

Abstract This paper presents a hybrid evolutionary method for identifying a system of ordinary differential equations (ODEs) from the observed time series. In this approach, the tree-structure based evolution algorithm and particle swarm optimization (PSO) are employed to evolve the architecture and the parameters of the additive tree models for the problem of system identification. Experimental results on modeling biochemical system show that the proposed method is more feasible and effective than other related works.

Keywords Additive tree models, Evolutionary Algorithms, Ordinary differential equations, Particle swarm optimization, Biochemical systems

1 Introduction

Many complex systems and nonlinear phenomena changing over time exist in physics, chemistry, economics, bioinformatics etc. Weather forecasting, quantum mechanics, wave propagation, stock market dynamics and identification of biological systems are examples [1]. The system of differential equations is a powerful and flexible model, which can describe complex relations among components [3]. So a lot of problems in these fields can be expressed by the ordinary differential equations (ODEs). Thus the differential equation identification is very important, and various methods have been proposed to infer ODEs during the last few years. The methods can be classified into two categories: one is to identify the parameters of the ODEs and another is to identify its structure. The former is exemplified by the Genetic Algorithms (GA), and the latter is by the Genetic Programming (GP). Cao and his colleagues use GP to evolve the ODEs from the observed time series in 1999 [2]. His main idea is to embed the genetic algorithm (GA) in genetic programming (GP), where GP is employed to discover and optimize the model's structure, and GA is employed to optimize the model's parameters. They show that the GP-based approach introduces numerous advantages over the most available modeling method. H.Iba propose the ODEs identification method by using the least mean square (LMS) along with the ordinary GP [3]. Some individuals are created at some intervals of generations and replace the worst individuals in the population by the LMS. I.G.Tsoulos

*Email: yhchen@ujn.edu.cn

and I.E.Lagar propose a novel method based on grammatical evolution [1]. This method forms generations of trial solutions expressed in an analytical closed form.

In 2005, we have proposed a new representation scheme of the evolved additive tree models for the system identification, especially the reconstruction of polynomials and the identification of linear/nonlinear systems. This model is robust, and it is easy to be analyzed by traditional techniques. This is because the evolved additive tree model is simple in form and is very similar with the traditional representation of the system to be reconstructed [10].

In this paper, we propose a hybrid evolutionary method, in which the tree-structure based evolution algorithm and particle swarm optimization (PSO) are employed to evolve the architecture and the parameters of the additive tree models for system of ordinary differential equation identification. The partitioning [4] is used in the process of identifying the system's structure. Each ODE in the ODEs is separately inferred.

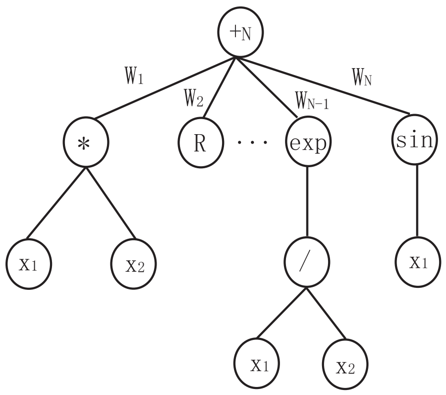
The paper is organized as follows. In Section 2, we describe the details of our method. In section 3, the four examples are performed to validate the effectiveness and precision of the proposed method. Conclusions are reported in Section 4.

2 Method

2.1 The models' structure optimization

2.1.1 The additive tree model

We use the tree-structure based evolution algorithm to evolve the architecture of the additive tree models for the system of ordinary differential equation identification. For this purpose, we encode the right-hand side of a ODE into an additive tree individual (Figure.1).



$$\frac{dX_i}{dt} = W_1 x_1 x_2 + W_2 R + W_{N-1} \exp(x_1/x_2) + W_N \sin(x_1)$$

Figure 1: Example of a ODE which encoded into an additive tree model.

Two instruction / operator sets I_0 and I_1 are used to generate the additive tree.

$I_0 = \{+_2, +_3, ..., +_N\}$

$I_1 = F \cup T = \{*, /, sin, cos, exp, rlog, x, R\}$

Here $F = \{*, /, sin, cos, exp, rlog\}$ and $T = \{x, R\}$ are respectively the function sets and the terminal sets, where $+_N, *, /, sin, cos, exp, rlog, x$, and R denote addition, multiplication, division, sine, cosine, exponent, logarithm, system inputs, random constant number taking N, 2, 2, 1, 1, 1, 1, 0 and -1 arguments respectively [10].

N is an integer number(the maximum number of the ODE terms), I_0 is the instruction set of the root node, and the instructions of other nodes are selected from the instruction set I_1. Note that if the right-hand sides of ODEs are the polynomial, the instruction set I_1 can be defined as $I_1 = \{*_2, *_3, ..., *_n, x_1, x_2, ..., x_n, R\}$.

We infer the system of ODEs with partitioning. Partitioning (equations that describe each variable of the system can be inferred separately) reduce the research space significantly. When using partitioning, a candidate equation for a signal variable is integrated by substituting references to other variables with data from the observed time series [4]. Thus each right hand side of the ODE system is evolved independently in parallel.

2.1.2 Evolving structure of the equation

Finding an optimal or near-optimal additive tree is an evolutionary process. In this study, the additive tree operators are used as follows.

(1) Mutation. In this paper, we choose three mutation operators to generate offsprings from the parents. These mutation operators are as following:
1) Change one terminal node: randomly select one terminal node in the tree and replace it with another terminal node which is generated randomly.
2) Grow: select a random leaf in the hidden layer of the tree and replace it with a newly generated subtree.
3) Prone: randomly select a function node in a tree and replace it with a terminal node selected in the set T.

(2) Crossover. First two parents are selected according to the predefined crossover probability P_c. And select one nonterminal node in the hidden layer for each additive tree randomly, and then swap the selected subtree.

(3) Selection. EP-style tournament selection is applied to select the parents for the next generation. This is repeated in each generation until the predefined number of generations or the best structure is found.

2.2 Parameter optimization of models using PSO

According to Fig.1, we check the parameters in equations, namely counting the number n_i(i=1,2,...,N, N is the number of the equations).

According to n_i, the particles are randomly generated initially. Each particle x_i represents a potential solution. A swarm of particles moves through space, with the moving velocity of each particle represented by the velocity vector v_i. At each step, each particle is evaluated and keep track of its own best position, which is associated with the best fitness it has achieved so far in a vector $Pbest_i$. And the best position among all the particles is kept as Gbest [6]. A new velocity for particle i is updated by

$$v_i(t+1) = v_i(t) + c_1 r_1 (Pbest_i - x_i(t)) + c_2 r_2 (Gbest(t) - x_i(t)) \quad (1)$$

Table 1: Parameters for experiments

	Exp1	Exp2	Exp3	Exp4
Population size	20	50	20	300
Generation	50	100	50	100
Crossover rate	0.7	0.7	0.7	0.7
Time series	1	1	1	10
Stepsize	0.01	0.01	0.05	0.01
Data point	30	100	48	10

where c_1 and c_2 are positive constant, and r_1 and r_2 are uniformly distributed random number in [0,1]. Based on the updated velocities, each particle changes its position according to the following equation:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2}$$

2.3 Fitness definition

The fitness of each variable is defined as the sum of squared error(SSE) :

$$fitness(i) = \sum_{k=0}^{T-1} \left(x'_i(t_0 + k\Delta t) - x_i(t_0 + k\Delta t)\right)^2 \tag{3}$$

where t_0 is the starting time, $\triangle t$ is the step size, T is the number of the data point, $x_i(t_0{+}k\triangle t)$ is the actual outputs of the i-th sample, and $x'_i(t_0{+}k\triangle t)$ is ODEs outputs. All outputs are calculated by using the approximate forth-order Runge-Kutta method. When calculating the outputs, some individuals may cause overflow. In this case, the individual which fitness becomes large will be weeded out from the population.

2.4 Summary of the proposed algorithm

The optimal design of each ODE can be described as follows.

(1) Randomly create an initial population(the structure and its corresponding parameters).
(2) Structure optimization is achieved by the additive tree variation operators, which is described in subsection 2.1.
(3) At some interval of generations, select the better structures to optimize parameters. Parameter optimization is achieved by PSO as described in subsection 2.2. In this process, the structure is fixed.
(4) If satisfactory solution is found, then stop; otherwise go to step (2).

3 Experimental results

We have prepared four tasks to test the effectiveness of our method. Experimental parameters are summarized in Table 1.

3.1 Example 1: E-cell simulation

In this part of the simulation, we use data of a metabolic network, called the E-cell system (a part of the biological phospholipid pathway). The network includes two reactions catalyzed by Glycerol kinase (EC2.7.1.30) and Glycerol-1-phosphatase (EC3.1.3.21). The network's external input is ATP [11]. Glycerol and sn-Glycerol-3phosphate are produced and consumed by the two reactions [11]. This network can be approximated as

$$\begin{cases} \dot{X_1} = -10.32X_1X_3 \\ \dot{X_2} = 9.72X_1X_3 - 17.5X_2 \\ \dot{X_3} = -9.7X_1X_3 - 17.5X_2 \\ \dot{X_4} = -2.3X_1 \end{cases} \tag{4}$$

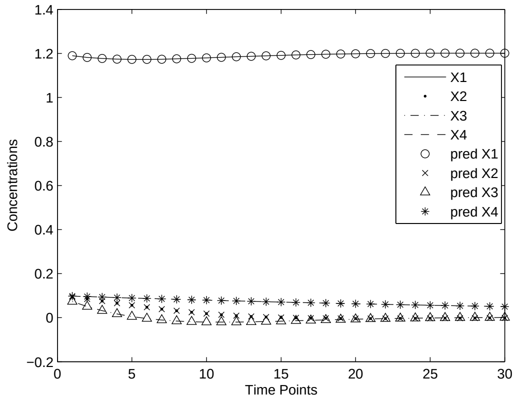


Figure 2: Time series of the acquired model.

The last equation is added to the model for testing whether the proposed method could produce false positives. The time series was generated for the above set of reactions with initial conditions $\{1.2,\ 0.1,\ 0.1,\ 0.1\ \}$ for $\{X_1,\ X_2,\ X_3, X_4\}$. Experimental parameters for this task are shown in Table 1. The used instruction set $I_0 = \{+_2, +_3, +_4, +_5\}$ and $I_1 = \{*, X_1, X_2, X_3, X_4\}$. We have acquired the system of eq.(6), which gave the sums of squared errors as (X_1, X_2, X_3,X_4)=(4.0×10^{-12}, 9.0×10^{-12}, 3.0×10^{-12}, 3.6×10^{-12}). The time series generated is shown in Fig.3 along with that of the target.

The resulting model using our method is listed in Table 2. Comparing with the true value, we can confirm that our generating system is almost identical to the target ODEs. We also have made further compare to examine the effectiveness of our proposed approach with GP+RLS and GP+KF. Obviously, the parameters of our model are closer to the targeted model than GP+RLS and GP+KF. And the GP+KF needs the 1000 individuals firstly. Our initial population size is only 20.

3.2 Experiment 2: Three-species Lotka-Volterra model

The Lotka-Volterra model describes interactions between two species, i.e., predators and preys, in an ecosystem [7]. The following ODEs represent a three-species Lotka-

Table 2: Obtained Parameters by GP+RLS [13], GP+KF [12] and Our Proposed Method

	true value	GP+RLS	GP+KF	Our Method
w_{11}	-10.32	-9.64	-10.34	-10.319973
w_{21}	9.72	13.42	8.87	9.720075
w_{22}	-17.5	21.8	-17.42	-17.500157
w_{31}	-9.7	-5.63	-9.74	-9.699986
w_{32}	17.5	12.64	17.15	-17.500034
w_{41}	-2.3	-2.14	-2.24	-2.299913

Volterra model:

$$\begin{cases} \dot{X_1} = (1 - X_1 - X_2 - 10X_3)X_1 \\ \dot{X_2} = (0.992 - 1.5X_1 - X_2 - X_3)X_2 \\ \dot{X_3} = (-1.2 + 5X_1 + 0.5X_2)X_3 \end{cases} \tag{5}$$

This system models the introduction of third species, i.e., a predator, into a two-species system of competition, i.e., preys. More precisely, X_1 and X_2 are the number of preys competing with each other, whereas X_3 represents the number of predators. [11]

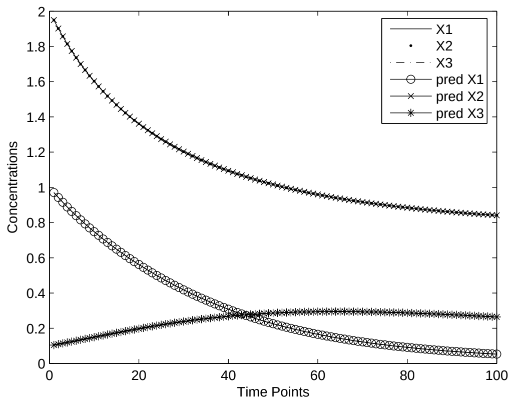


Figure 3: Time series of the acquired model for Lotka-Volterra model.

The time series are generated for the above set of ODEs with initial conditions {1.0, 2.0, 0.1 } for $\{X_1, X_2, X_3\}$. The generated time series are shown in Fig.4. Experimental parameter for this task are shown in Table 1. The used instruction set $I_0 = \{+_3, +_4, +_5, +_6\}$ and $I_1 = \{*, X_1, X_2, X_3\}$. We have acquired the system of eq.(6), and note that the two systems of ODEs, i.e., eqs.(5) and eqs.(6), are almost identical except for slightly different coefficients. The sums of squared errors of ODEs are 8.1×10^{-11}.

$$\begin{cases} \dot{X_1} = 1.00122X_1 - 1.0019{X_1}^2 - 0.9991X_1X_2 - 10.0093X_1X_3 \\ \dot{X_2} = 0.9934X_2 - 1.5008X_1X_2 - 0.9995{X_2}^2 - 1.001X_2X_3 \\ \dot{X_3} = -1.1988X_3 + 5.001X_1X_3 + 0.4984X_2X_3 \end{cases} \tag{6}$$

We conduct further experiments with the above model to compare with the performance of the Multi Expression Programming (MEP), where the structure of the ODE is inferred by the Multi Expression Programming (MEP) and the parameters of the ODE are optimized by using particle swarm optimization (PSO). With the same population size and iteration, the SSE of MEP is 7.431×10^{-7}, and the SSE of our method is only 1.936×10^{-9}. So our proposed method performs better than MEP in precision.

3.3 Experiment 3: bimolecular reaction

A bimolecular reaction equations [14]are described below:

$$X_2 + X_1 \xrightarrow{K_1} X_3 \tag{7}$$

$$X_3 \xrightarrow{K_2} X_4 + X_2 \tag{8}$$

The corresponding rate equations for all the four species are as follows:

$$\begin{cases} \dot{X_1} = -2X_1X_2 \\ \dot{X_2} = -2X_1X_2 + 1.2X_3 \\ \dot{X_3} = 2X_1X_2 - 1.2X_3 \\ \dot{X_4} = 1.2X_3 \end{cases} \tag{9}$$

The time series are generated for the reactions with initial conditions$\{1, 0.1, 0, 0\}$for $\{X_1, X_2, X_3, X_4\}$ which is shown in Fig.4 along with targeted time series. Experimental parameters for this task are shown in Table 1. The used instruction set $I_0 = \{+2, +4, +5, +6\}$ and $I_1 = \{*, X_1, X_2, X_3, X_4\}$. We have acquired the system of eq.(11), which give the sums of squared errors as (X_1, X_2, X_3, X_4)=(1.0×10^{-11}, 5.7×10^{-12}, 3.0×10^{-12}, 1.5×10^{-11}).

$$\begin{cases} \dot{X_1} = -1.9920X_1X_2 \\ \dot{X_2} = -1.1983X_1X_2 + 1.9920X_3 \\ \dot{X_3} = 1.9920X_1X_2 - 1.1983X_3 \\ \dot{X_4} = 1.1983X_3 \end{cases} \tag{10}$$

Compared with eq.(10) [14], our predicted model and parameters are closer to the target system. The method is executed with time series with different time step size and with different initial condition. And the results have scarcely any change.

$$\begin{cases} \dot{X_1} = -1.99999X_1X_2 \\ \dot{X_2} = 1.20000X_3 - 1.99999X_1X_2 \\ \dot{X_3} = -1.20000X_3 - 2.00000X_1X_2 \\ \dot{X_4} = 1.99999X_3 \end{cases} \tag{11}$$

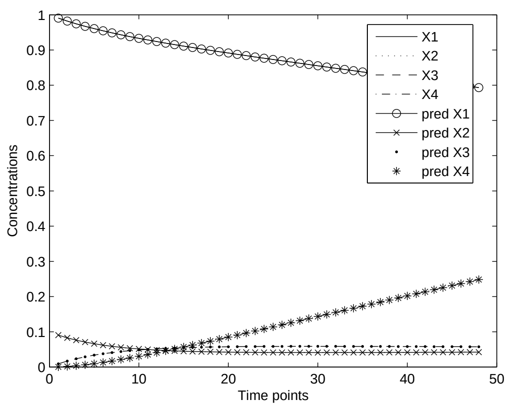


Figure 4: Time series of the acquired model for bimolecular reaction.

Table 3: Parameters of the genetic network system

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	5.0			1.0		-1.0	10.0	2.0				
2	10.0	2.0					10.0		2.0			
3	10.0		-1.0				10.0		-1.0	2.0		
4	8.0			2.0		-1.0	10.0				2.0	
5	10.0				2.0		10.0					2.0

3.4 Experiment 4: a gene regulatory network

Figure 5 shows the example of a gene regulatory network. This type of network can be modeled by a so-called S-system model [15]. This model is based on approximating kinetic laws with multivariate power-law functions. The model consists of n nonlinear ODEs and the generic form of equation i is given as follows:

$$X_i^{'}(t) = \alpha_i \prod_{j=1}^{n} X_j^{g_{ij}}(t) - \beta_i \prod_{j=1}^{n} X_j^{h_{ij}}(t) \tag{12}$$

where X is the vector of dependent variable, α and β are vectors of non-negative rate constants and g and h are matric of kinetic orders [15].

The parameters of the genetic network are given in Table 3. And the time series consists of 10 different experiments which initial conditions are created randomly with 11 uniformly sampled data-points per variable. The used instruction set $I_0 = \{+2, +3, +4\}$, $F = \{*, a^x\}$, and we identify the correct model(parameters in the Table 4) with Intel Pentinum Dual 2.00GHz processor and 1GB memory in 1.9 h averagely. Shinichi Kikuchi [15] obtained one false positive interaction (h_{53} = 0.7) using 70 h on a super-computer with a cluster of 1G processors (Pentium 3, 933 MHz). Obviously our approach is more accurate and significantly faster compared with the genetic algorithm approach by Shinichi

Table 4: Parameters estimated by our method

i	α_i	g_{i1}	g_{i2}	g_{i3}	g_{i4}	g_{i5}
1	5.0			0.9528		-0.9948
2	10.0	1.9984				
3	10.0		-0.9978			
4	7.9579			2.0		-1.047
5	10.0				1.9943	
	β_i	h_{i1}	h_{i2}	h_{i3}	h_{i4}	h_{i5}
1	10.0	1.8775				
2	10.0		2.0			
3	10.0		-0.9988	1.9988		
4	9.9524				2.0	
5	10.0					2.0

Kikuchi.

The above four experiments have been used widely to test the performance of the methods inferring the ordinary differential equations for identification of biochemical systems in the previous research [11, 12, 13, 14, 15]. From our experimental results, we can see that not only the linear differential equation but also the nonlinear differential equation could be correctly identified. And compared with the general methods, our method not only can identify correctly the biochemical systems especially parameter through the very short iterative times, but also needs the less initial population. So our proposed method works well for modeling biochemical systems.

4 Conclusion

In this paper, a hybrid evolutionary method of evolving ODEs is proposed. By several experiments, we succeed in creating the systems of ODEs which are very close to the target systems. The experimental results show the effectiveness and veracity of the proposed method. The proposed method has two advantages. (1) The evolved additive tree model is robust and easy to analyze by using traditional techniques. This is because the evolved additive tree model is simple in form and is very similar with the traditional representation of the system. So we can acquire the best structure of the ODE only by a small population. (2) With partitioning, each ODE of the ODEs can be inferred separately and the research space reduces rapidly, so we can acquire the best system very fast.

In the future work, we will apply our approach to solve some real problems in physics, chemistry, economics, bioinformatics etc. Weather forecasting, quantum mechanics, stock market dynamics and identification of biological systems are some examples.

Acknowledgment

This research was supported by the NSFC (60573065), the the Natural Science Foundation of Shandong Province (Y2007G33), and the Key Subject Research Foundation of Shandong Province.

References

[1] I.G.Tsoulos, I.E.Lagaris, Solving differential equations with genetic programming, Genetic Programming and Evolvable Machines, Volume 7, Issue 1, pp.1389-2576, 2006.

[2] H.Cao, L.Kang, Y.Chen, J.Yu, Evolutionary Modeling of Systems of Ordinary Differential Equations with Genetic Programming, Genetic Programming and Evolvable Machines, vol.1, no.40, pp.309-337, 2000.

[3] Erina Sakamoto, H.Iba, Inferring a system of differential equations for a gene regulatory network by using genetic programming, Proc. Congress on Evolutionary Computation, pp.720-726, 2001.

[4] Bongard J., Lipson H., Automated reverse engineering of nonlinear dynamical systems, Proceedings of the National Academy of Science, 104(24), pp. 9943-9948, 2007.

[5] Oltean, M., Grosan, C., Evolving digital circuits using multi expression programming, In: Zebulum, R. et al., NASA/DoD Conference on Evolvable Hardware, 24-26 June, Seattle. IEEE Press, NJ, pp. 87-90, 2004.

[6] Yuehui Chen, Bo Yang, Ajith Abraham, Flexible Neural Trees Ensemble for Stock Index Modeling, Neurocomputing, Vol. 70, Issues 4-6, pp. 697-703, 2007.

[7] Y.Takeuchi, Global Dynamical Properties of Lotka-Volterra Systems, Singapore: World Scientific, 1996. mechanisms from measured time-series, J. Phys. Chem, pp.970-979, 1995.

[8] Gennemark P, Wedelin D. Efficient algorithms for ordinary differential equation model identification of biological systems. IET Syst Biol, 1(2):120-129, 2007 .

[9] Savageau, M.A., Biochemical systems analysis: a study of function and design in molecular biology, (Addison-Wesley, Reading, MA,1976).

[10] Yuehui Chen, Ju Yang, Yong Zhang and Jiwen Dong, Evolving Additive tree models for System Identification, International Journal of Computational Cognition, Vol.3, No.2, pp. 19-26, 2005.

[11] Hitoshi Iba, Inference of differential equation models by genetic programming, Elsevier Science Inc, Volume 178, Issue 23, pp. 4453-4468, 2008.

[12] Lijun Qian, Haixin Wang, Dougherty, E.R. Inference of Noisy Nonlinear Differential Equation Models for Gene Regulatory Networks Using Genetic Programming and Kalman Filtering, Signal Processing, IEEE Transactions on, Volume: 56, Issue: 7, pp. 3327-3339, 2008.

[13] S. Ando, E. Sakamoto, and H. Iba, Evolutionary modeling and inference of gene network, Inf. Sci., vol. 145, pp. 237-259, 2002.

[14] Srividhya, JCrampin, EJMcSharry, PESchnell, S, Reconstructing biochemical pathways from time course data, Proteomics, 7(6), pp.828-838, 2007.

[15] Shinichi Kikuchi, Daisuke Tominaga1, Masanori Arita, Dynamic modeling of genetic networks using genetic algorithm and S-system, BIOINFORMATICS, 19(5), pp. 643-650, 2003.