A Real-coded Genetic Algorithm for Process Control and Identification

Wei-Der Chang
Department of Computer and Communication
Shu-Te University
Kaohsiung 824, TAIWAN
wdchang@mail.stu.edu.tw

Abstract

A real-coded genetic algorithm (GA) applied into the system identification and control for a class of nonlinear systems is proposed in this paper. It is well known that GA is a globally optimal technique borrowing the concepts from biological evolutionary theory. The ordinary form of GA used for solving a given optimization problem is a binary encoding during operating procedures. For most of real control applications, however, a real-valued encoding is often used and is easy to be implemented directly in the computer programming. In this paper, based on using real-coded GA, a complete design procedure for estimating parameters of nonlinear system and then for designing an off-line PID controller is presented. Finally, some simulation results by examining a nonlinear process will be demonstrated to show the estimate and control performances by using the proposed method.

1. Introduction

Genetic algorithms as well as neural networks and fuzzy systems belong to the category of artificial intelligence. Based on the type of modeling the natural evolution, GA can search for optimal or near-optimal solutions for an optimization problem over the search domain, and have superior performance over the traditional optimal techniques, e.g., the gradient descent method. This is due to searching for solution from only one single direction on the search space [1]-[3]. Alternatively, GA can be regarded as a search method from multiple directions, because it inherently possesses crossover and mutation operations when searching procedures are performed. This implies that it has the ability to escape from a local minimum.

In the traditional GA, all the variables of interest must first be encoded as binary digits (genes) forming a string (chromosome). Then three standard genetic operations, i.e., reproduction, crossover, and mutation are performed to produce a new generation. Such procedures are repeated until the pre-specified number of generations is achieved, or the required accuracy is satisfied. For most of real control system designs, once a binary-coded GA is used, the relative parameters concerned with the plant and controller should be first encoded as binary alphabets in order to be suitably computed in the traditional way. After a series of genetic manipulations, the final binary alphabets are then returned as real numbers. This is an indirect optimization problem searching.

On the other hand, a real-coded GA has been also introduced to a wide variety of applications in recent years as stated in [4]-[7]. All genes in a chromosome are real numbers. It is more suitable for most of real control applications that genes are directly real values during genetic operations. Because the procedures of binary coding for a real number may suffer for the loss of precision depended on the number of the bits used. Expectably, it will be quite complicated and difficult to implement when the numerical values are large and have the decimal fraction. Moreover, the length of chromosomes used in the real-coded GA becomes much shorter than that in the binary-coded way. This implies that the computer programming for such algorithms can be easily performed.

The work of system identification is very important and essential for the control system engineering. According to a known mathematical or an estimated model for system, a controller will then be designed by using a lot of different control techniques such that the certain output response of system can be satisfied. In recent years, solving for identification problems using artificial intelligence techniques have been successively proposed, such as using fuzzy logic systems [8]-[10], neural networks [11]-[14], and neural-fuzzy systems [15][16]. In these studies, they focused on the identification problem that the system structure is assumed to be unknown. Conversely, if the model structure of system has been known, the residual problem is how to correctly evaluate the system parameters or coefficients for this kind of model structure. The least-squares method is a basic technique often used for parameters estimation. In [17], it has been successfully used to estimate the parameters in the static and dynamical systems, respectively. But, the least-squares method is only suitable for the model structure of system having the property of being linear in the parameters. Once the form of model structure is not linear in the parameters, this approach may be invalid. The same problem also occurs in using other estimate techniques such as maximum-likelihood and instrumental variable methods. These recursive schemes are in essence local search techniques that search for the optimum by using gradient method. They often fail in the search for global optimum if the search space is not differentiable or linear in the parameters [18]. On the topic of system identification, some studies based on using the traditional binary-coded GA were exploited as shown in [18][19]. In [18], they applied the binary-coded GA for estimating the locations of poles and zeros of a transfer function and then used this estimated model to design a discrete time pole placement adaptive controller. Similarly, Jiang and Wang [19] proposed a searching method for parameters estimation of nonlinear systems based on using the binary-ceded GA.

Another topic discussed in this paper is focused on the off-line PID controller design using real-coded GA. It is well known that the use of PID control has a long history in control engineering and is acceptable for most of real applications due to its simplicity in architecture. The key for designing a PID controller is on how to determine three PID control gains, i.e., proportional gain K_p, integral gain K_i, and derivative gain K_d. For a single-input-single-output (SISO) system, the tuning methods proposed by Cohen-Coon (C-C) and Ziegler-Nichols (Z-N) were frequently used. Recently, some studies combining neural networks or fuzzy logic systems with PID control systems have been also introduced due to their powerful learning and adaptive capacity [11][20]-[22]. Three control gains are determined or adjusted by the uses of neural networks or fuzzy logic systems according to certain adaptation mechanism. Moreover, Some studies that applied traditional binary-ceded GA into the off-line PID controller design for linear systems as shown in [23][24] were proposed. In order to meet the binary way, it is necessary to encode three real-number gains as the form of binary alphabet. After genetic operations, the resulted binary coding is then decoded as real values required on the actual PID control systems. Moreover, since the kind of control strategy belong to off-line PID controller design, the mathematical model of plant should be known or be estimated in advance. However, in [23][24] they did not discuss in details how the illustrative system model was obtained. In this paper, based on using a real-coded GA, an overall design procedure for parameters estimation and successively for PID controller design for a class of nonlinear systems will be suggested. In this way, all unknown system parameters are first evaluated and according to this model the off-line PID control design strategy is proposed to obtain the proper three control gains. Both unknown system parameters and three undetermined PID control gains are in the form of real number during genetic operations.

2. Real-coded genetic algorithm

Before introducing three genetic operations, some notations normally used for GA will be first introduced. Let $\Theta = [\theta_1, \theta_2, \cdots, \theta_m]$ where Θ is a set of possible solution to the optimization problem and called a chromosome from the evolutionary point of view and θ_i, for $i \in \underline{m}$ and $\underline{m} = \{1, 2, \cdots, m\}$, in a chromosome is called a gene. A search space Ω_Θ for Θ is defined by

$$\Omega_\Theta = \{\Theta \in \Re^m \mid \theta_{1\,min} \le \theta_1 \le \theta_{1\,max}, \theta_{2\,min} \le \theta_2 \le \theta_{2\,max}, \cdots, \theta_{m\,min} \le \theta_m \le \theta_{m\,max}\}. (1)$$

All genes θ_i, for $i \in \underline{m}$, in the chromosome will be evolved inside this constrained space Ω_Θ during the genetic operations. Once a generated chromosome by genetic operations goes beyond Ω_Θ, then the original chromosome will be retained. Let N represent the number of chromosomes in the population, i.e., the size of population, and parameters p_r, p_c, and p_m are referred to as probabilities of reproduction, crossover, and mutation, respectively.

2. 1 Reproduction

There are two well known selection mechanisms, i.e., roulette wheel and tournament selections used for reproduction operation. The roulette wheel selection can be visualized by imagining a wheel where each chromosome occupies an area that is related to its value of objective function. When a spinning wheel stops, a fixed marker determines which chromosome will be selected to reproduce into the mating pool [5]. This kind of selection mechanism needs more numerical computations. In this study, the tournament selection is quite simple and suitable for checking whether a chromosome can reproduce or not according to its corresponding objective function. For the tournament selection, $p_r \times N$ chromosomes with minimum values of objective function are more added into the population, and correspondingly $p_r \times N$ chromosomes with maximum values of objective function are discarded from the population. This implies that the resulting population has the same size with the original. After the selection, all chromosomes are completely put in the mating pool. The next step is to generate new offspring by applying the following crossover and mutation operations on chromosomes in the mating pool.

2. 2 Crossover

The N chromosomes in the mating pool are randomly divided into $N/2$ pairs where serve as parents and will be crossed each other. Suppose that Θ_1 and Θ_2 are parents of a given pair, c is a random number chosen from $[0,1]$. If $c \geq p_c$, then the following crossover operations for Θ_1 and Θ_2 are performed

$$\begin{aligned}
&\textit{if } obj(\Theta_1) < obj(\Theta_2) \\
&\Theta_1 \leftarrow \Theta_1 + r(\Theta_1 - \Theta_2), \\
&\Theta_2 \leftarrow \Theta_2 + r(\Theta_1 - \Theta_2), \\
&\textit{else} \\
&\Theta_1 \leftarrow \Theta_1 + r(\Theta_2 - \Theta_1), \\
&\Theta_2 \leftarrow \Theta_2 + r(\Theta_2 - \Theta_1),
\end{aligned} \tag{2}$$

where $obj(\Theta_1)$ and $obj(\Theta_2)$ stand for values of objective function obtained from Θ_1 and Θ_2, respectively, and $r \in [0,1]$ is a random number deciding the crossover grade of these two. It is easily observed from Figs. 1 and 2 that after the use of the crossover operation both Θ_1 and Θ_2 are moved toward the direction where the value of objective function is smaller. If $c < p_c$, no crossover operation is performed.

2. 3 Mutation

The mutation operation follows the crossover and provides a possible mutation on some chosen chromosomes Θ. Only the randomly selected $p_m \times N$ chromosomes in the current population are mutated. The formula of mutation operation for the chosen Θ is given by

$$\Theta \leftarrow \Theta + s \times \Phi, \tag{3}$$

where s is a positive constant and $\Phi \in \Re^m$ is the random noise vector to produce a perturbation on Θ.

The procedures that have once run reproduction, crossover, and mutation operations are called a generation. The algorithm stops if the desired value of objective function is satisfied or the pre-specified number of generations is achieved. Notice again that if a generated chromosome during genetic operations is outside the search space Ω_Θ, then the original chromosome will be retained. The overall design steps based on using real-coded GA can be summarized as follows.

I. First, create a population with the size of N chromosomes, in which all genes are generated from Ω_Θ in (1).
II. Evaluate the corresponding value of objective function for each chromosome in the population.
III. If the pre-specified number of generations G is reached or the value of objective function produced by a chromosome in the population is less than a desired value of ε, then stop.
IV. Perform operations of reproduction, crossover in (2), and mutation in (3). Notice that if the resulted chromosome under operations is outside the Ω_Θ, then the original is retained.
V. Go back to Step II.

3. Nonlinear system identification

A class of nonlinear systems described as the form of the discrete dynamic equations are considered in this study as follows

$$x(k+1)=f(k,\,x(k),\,u(k),\,P_1),\;\; x(0)=x_0\,,$$
$$y(k)=h(k,\,x(k),\,u(k),\,P_2), \tag{4}$$

where $u\in\Re$ is the input, $x\in\Re^n$ are the system states, $y\in\Re$ is the output, P_1 and P_2 represent sets of unknown system parameters that probably contain the time delay, $f(\cdot)$and $h(\cdot)$ are nonlinear functions. Suppose that the estimated nonlinear systems that will fit the plant of (4) are modeled as

$$\hat{x}(k+1)=\hat{f}(k,\,\hat{x}(k),\,u(k),\,\hat{P}_1),$$
$$\hat{y}(k)=\hat{h}(k,\,\hat{x}(k),\,u(k),\,\hat{P}_2), \tag{5}$$

where $\hat{x}$ and $\hat{y}$, respectively, are the estimated system states and output, driven by the actual input u as (4), $\hat{f}(\cdot)$ and $\hat{h}(\cdot)$ are the estimates of $f(\cdot)$ and $h(\cdot)$, $\hat{P}_1$ and $\hat{P}_2$ are the estimates of P_1 and P_2, respectively, found by the use of real-coded GA. For simplification, let $\Theta=[\theta_1,\theta_2,\cdots,\theta_m]$ where m is referred to as the total number of unknown system parameters be a rearranging vector that collects all parameters in $\hat{P}_1$ and $\hat{P}_2$. In order to precisely obtain Θ, the following assumptions are requested for nonlinear systems.

1. The output of system must be measurable and finite in each sampling step.
2. Every parameter of system must be in connection with the output, i.e., Θ could be evaluated from the measurement of the output.

Before proceeding with the genetic operations, a performance index or an objective function should be first defined because this will significantly influence on how the evolutionary type on Θ is performed. In general, the GA only needs to evaluate the objective function to guide its search. There is no requirement for derivatives that are often used in solving for the traditional optimization problems. In this study, the total summation of square error (*SSE*) is taken as an objective function, which is given by

$$SSE=\sum_{k=1}^{M}[y(k)-\hat{y}(k)]^2=\sum_{k=1}^{M}e^2(k), \tag{6}$$

where M is the given sampling number and e is the error between y and $\hat{y}$. The residual problem is to correctly find out the values of Θ based on using real-coded GA in such a way that the value of *SSE* in (6) is minimized as possibly. Fig. 3 shows the block diagram for parameters estimation using real-coded GA.

4. PID controller design

After a process of obtaining the actual system parameters by using real-coded GA, it follows that a genetic approach to designing an off-line PID controller will be suitably proposed.

The continuous form of a PID controller, with input e and output u, is given as

$$u(t)=K_p\left[e(t)+\frac{1}{T_i}\int_0^t e(\tau)d\tau+T_d\frac{d}{dt}e(t)\right], \tag{7}$$

where K_p is the proportional gain, T_i is the integral time constant, and T_d is the derivative time constant. We can also rewrite (7) as

$$u(t)=K_pe(t)+K_i\int_0^t e(\tau)d\tau+K_d\frac{d}{dt}e(t), \tag{8}$$

where $K_i=K_p/T_i$ and $K_d=K_pT_d$, respectively, stand for the integral gain and the derivative gain. For convenience, let $\Theta=[\theta_1,\theta_2,\theta_3]^T$ $=[K_p,K_i,K_d]^T$ represent the vector of PID controller gains called a chromosome and three gains K_p, K_i, and K_d in Θ are genes that represent a set of potential solution to GA-based a PID controller tuning problem. We emphasize again that a chromosome Θ in this study is direct real-valued coding and evolved to produce next generation with better performance by applying evolutionary operators.

The off-line tuning strategy combined the read-coded GA with the PID control system is simply depicted in Fig. 4, where y_d is the desired output, y is the plant output, and u is the control input generated by the PID controller as defined in (8). In order to match the type of PID control system, the objective function of *SSE* in (6) should be rewritten as

$$SSE=\sum_{k=1}^{M}[y_d(k)-y(k)]^2=\sum_{k=1}^{M}e^2(k)\,. \tag{9}$$

Similarly to that of parameters estimation, our control aim is to design three PID controller gains using the search technique of real-code GA such that the value of *SSE* in (9) is minimized.

5. Simulation results

Consider an unstable nonlinear system whose dynamic equations of discrete form are in the following [19]

$x_1(k+1) = a_1 x_1(k) x_2(k)$, $x_1(0) = 1$,

$x_2(k+1) = a_2 x_1^2(k) + u(k)$, $x_2(0) = 1$,

and

$$y(k) = a_3 x_2(k) - a_4 x_1^2(k). \qquad (10)$$

In this example, the actual values of unknown system parameters in (10) are assumed to be $\Theta = [a_1, a_2, a_3, a_4] = [0.5, 0.3, 1.8, 0.9]$. Note that it should be more careful and need a trial-and-error work when the search space Ω_Θ is constructed for parameters estimation and for PID controller design, because the nonlinear system of (10) is inherently unstable. Otherwise, the infinite system output may occur during the computational process.

Parameters estimation

In parameters estimation part, the used genetic parameters are given by

$\theta_{1min} = 0$, $\theta_{1max} = 2$, $\theta_{2min} = 0$, $\theta_{2max} = 2$, $\theta_{3min} = 0$, $\theta_{3max} = 2$, $\theta_{4min} = 0$, $\theta_{4max} = 2$, $M = 8$, $N = 20$, $p_r = 0.2$, $p_c = 0.3$, $p_m = 0.2$, $s = 0.1$, $G = 200$,

Each element in the noise vector Φ in (3) is also a random number from $[-0.1, 0.1]$. The comparing results between real-coded and binary-coded GAs are demonstrated in Tab.1, and the corresponding convergence curves are also shown in Figs. 5 and 6. For such an unstable nonlinear system, it is obvious from simulation results that the more accurate estimate than one by using the binary-coded GA [19] can be achieved.

PID controller design

According to the above estimating results, it follows that the procedure of designing an off-line PID controller will be performed. The control objective is to wish that the plant output y is regulated to the desired output $y_d = 2$. In this simulation, the search space Ω_Θ for genetic operations is seriously constructed by

$\theta_{1min} = 0.0$, $\theta_{1max} = 1.0$, $\theta_{2min} = 0.0$, $\theta_{2max} = 1.0$, $\theta_{3min} = 0.0$, $\theta_{3max} = 0.2$,

and the used parameters are chosen as follows

$M = 50$, $N = 10$, $p_r = 0.2$, $p_c = 0.5$, $p_m = 0.1$, $s = 0.1$, $G = 3000$.

Each element in the noise vector Φ in (3) is also a random number chosen from $[-0.01, 0.01]$. After genetic operations, the three PID control gains are, respectively, $K_p = 0.8413$, $K_i = 0.9932$, and $K_d = 0.0095$ under $G = 3000$ generations. This implies that it needs a great number of evolutionary generations such that better PID controller can be obtained for such a nonlinear system. Finally, the output response is shown in Fig. 7.

6. Conclusions

The application of a real-coded genetic algorithm to the system parameters identification and to the PID controller tuning, respectively, have been proposed for a class of nonlinear system in this paper. The system model considered in this study need not be linear in the parameters, although it is always necessary to request the property of being linear for most of traditional estimate methods. Each of unknown system parameters that will be estimated is regarded as a gene and collect them to construct a chromosome in the form of real number throughout. With the use of real-coded GA, they are then evolved by reproduction, crossover, and mutation operations, respectively, to generate a new excellent offspring until the pre-specified number of generations is reached. Based on the precisely estimated model and also the use of genetic approach, it is quite suitable to determine the PID controller in the off-line style. Similarly, three control gains are directly regarded as genes and form a chromosome. Real-value computations are all used throughout three genetic operations. Finally, an unstable nonlinear process system is illustrated to demonstrate the excellent performance by using our proposed method compared with other method for system parameters estimation.

References

[1] L. Davis, Handbook of Genetic Algorithms, Van Nostrand, New York, 1991.

[2] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[3] Y.P. Huang, C.H. Huang, Real-valued genetic algorithm for fuzzy grey prediction system, Fuzzy Sets and Systems 87 (1997) 265-276.

[4] S. Tsutsui, D.E. Goldberg, Search space

boundary extension method in real-coded genetic algorithm, Information Sciences 133 (2001) 229-247.
[5] A. Blanco, M. Delgado, M.C. Pegalajar, A real-coded genetic algorithm for training recurrent neural networks, Neural Networks 14 (2001) 93-105.
[6] K. Deb, S. Gulati, Design of truss-structures for minimum weight using genetic algorithms, Finite Elements in Analysis and Design 37 (2001) 447-465.
[7] M. Zamparelli, Genetically trained Cellular neural networks, Neural Networks 10 (6) (1997) 1143-1151.
[8] L.X. Wang, Course in Fuzzy Systems and Control, Prentice-Hall, NJ, 1997.
[9] L.X. Wang, Stable adaptive fuzzy controllers with application to inverted pendulum tracking, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics 26 (5) (1996) 677-691.
[10] Y. Tang, N. Zhang, Y. Li, Stable fuzzy adaptive control for a class of nonlinear systems, Fuzzy Sets and Systems 104 (1999) 279-288.
[11] S. Omatu, M. Khalid, R. Yusof, Neuro-Control and Its Applications, Springer, London, 1995.
[12] R.J. Wai, H.H. Lin, F.J. Lin, Hybrid controller using neural networks for identification and control of induction servo motor drive, Neurocomputing 35 (2000) 91-112.
[13] F.C. Chen, Back-propagation neural networks for nonlinear self-tuning adaptive control, IEEE Control Systems Magazine April (1990) 44-48.
[14] Y.M. Park, M.S. Choi, K.Y. Lee, An optimal tracking neuro-controller for nonlinear dynamic systems, IEEE Transaction on Neural Networks 7 (5) (1996) 1099-1110.
[15] Y.G. Leu, T.T. Lee, W.Y. Wang, Observer-based adaptive fuzzy-neural control for unknown nonlinear dynamical systems, IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics 29 (5) (1999) 583-591.
[16] C.T. Chen, S.T. Peng, Intelligent process control using neural fuzzy technique, Journal of Process Control 9 (1999) 493-503.
[17] K.J. Astrom, B. Wittenmark, Adaptive Control, Addison-Wesley, Massachusetts, 1995.
[18] K. Kristinsson, G.A. Dumont, System identification and control using genetic algorithms, IEEE Transactions on Systems, Man, and Cybernetics 22 (5) (1992) 1033-1046.
[19] B. Jiang, B.W. Wang, Parameter estimation of nonlinear system based on genetic algorithm, Control Theory and Applications 17 (1) (2000) 150-152.
[20] B. Hu, G.K.I. Mann, R.G. Gosine, New methodology for analytical and optimal design of fuzzy PID controller, IEEE Transactions on Fuzzy Systems 7 (5) (1999) 521-539.
[21] C.T. Chao, C.C. Teng, A PD-like self-tuning fuzzy controller without steady-state error, Fuzzy Sets and Systems 87 (1997) 141-154.
[22] A.E.B. Ruano, P.J. Fleming, D.I. Tones, Connectionist approach to PID autotuning, IEE Proceedings-D 139 (3) (1992) 279-285.
[23] B. Porter, A.H. Jones, Genetic tuning of digital PID controllers, Electronic Letters 28 (9) (1992) 843-844.
[24] C. Vlachos, D. Williams, J.B. Gomm, Genetic approach to decentralised PI controller for multivariable processes, IEE Proceedings-Control Theory Applications 146 (1) (1999) 58-64.

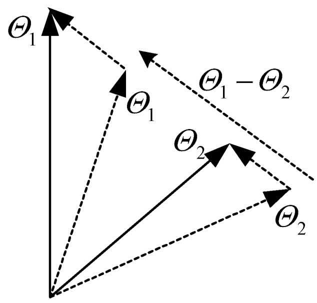

Fig. 1. $obj(\Theta_1) < obj(\Theta_2)$.

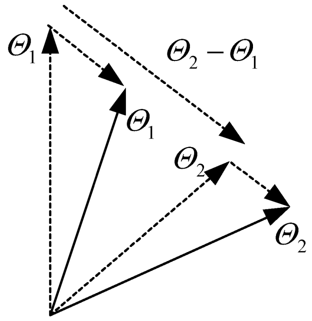

Fig. 2. $obj(\Theta_1) > obj(\Theta_2)$.

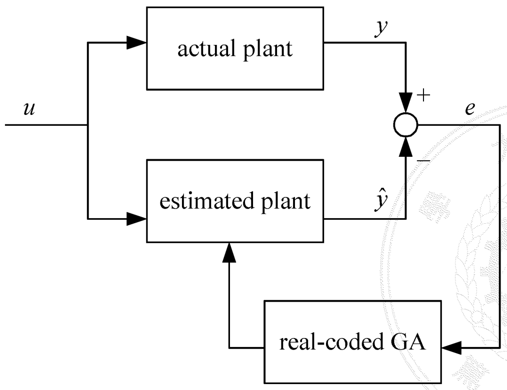

Fig. 3. Block diagram for parameters estimation using real-coded GA.

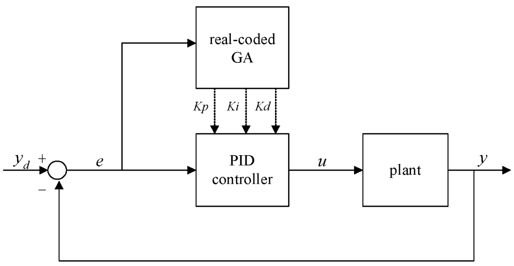

Fig. 4. Off-line tuning strategy for PID controller design using real-coded GA.

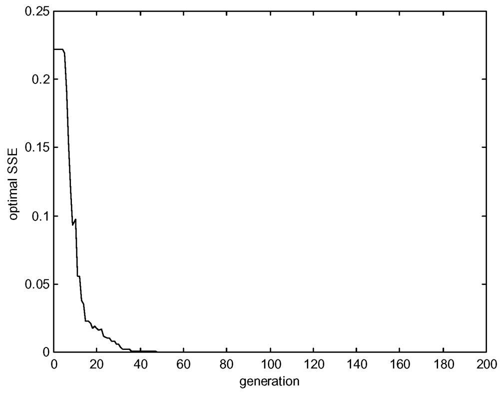

Fig. 5. Convergence curve of SSE for parameters estimation.

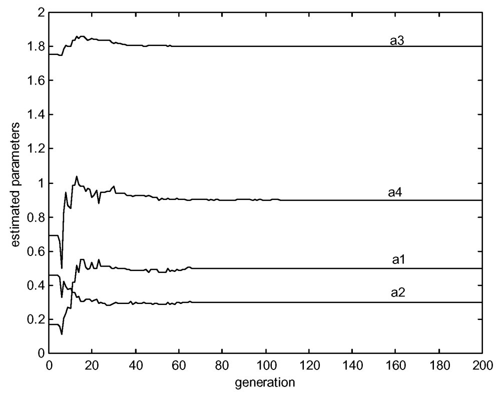

Fig. 6. Convergence curves of the corresponding parameters: a_1, a_2, a_3, *and* a_4.

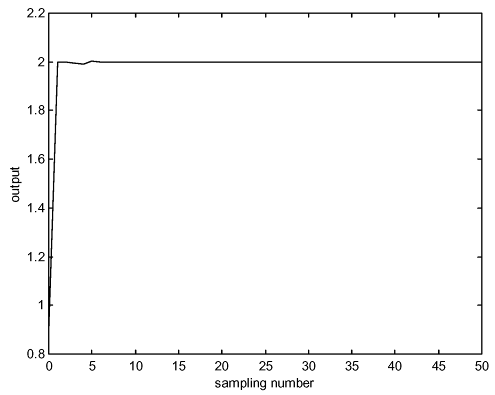

Fig. 7. Output response with $y_d = 2$.

	a_1	a_2	a_3	a_4
actual parameters	0.5	0.3	1.8	0.9
real-coded method	0.4987	0.2993	1.8001	0.9001
binary-coded method	0.4916	0.3014	1.8432	0.9267

Tab. 1. Comparison results of parameters estimation between the real-coded method and the traditional binary-coded method proposed by [19].