Nonlinear System Identification by Evolutionary Computation and Recursive Estimation Method

Jih-Gau Juang, Bo-Shian Lin
Department of Communication and Guidance Engineering
National Taiwan Ocean University, Keelung, Taiwan
jgjuang@mail.ntou.edu.tw

Abstract—**Nonlinear system identification using evolutionary computation and recursive estimation method is presented. Four different recursive estimation methods, recursive least-squares, recursive least-squares with exponential forgetting, stochastic algorithm, and projection algorithm, combined with evolution algorithm are used in this study. Conventional system identification using recursive estimation methods are also given for comparison. After test, the proposed scheme has better convergence and accuracy on parameter estimation than the conventional estimation method.**

I. INTRODUCTION

When a physical system structure and parameters are not available or dependent on time or operating conditions, a mathematical model representing the system behavior may not be obtained. For this case, the system parameters should be obtained using a system identification procedure. The purpose of system identification is to construct a mathematical model of a physical system from input/output data for system analysis or controller design. Research and development of linear system identification has been studied for more than three decades [1]-[3]. However, identification of nonlinear systems is a respectively new topic of interest. Nonlinear characteristics such as saturation, dead-zone, etc., are inherent in many real systems. In order to analyze and control such systems, identification of nonlinear characteristics is necessary. Hence, nonlinear system identification becomes more challenging and has received much attention [4]-[7].

The role of identification consists in describing the behavior of a given plant by a model suitably selected within an appropriate class of systems [8]. The selection criterion exploits the information contained in the observation data available over finite time horizon. Characterizing the class of models is a key point to make the problem feasible that is to make consistent with data [8]. Any identification procedure consists of collecting the data, selecting a set of candidate models and finding the best model within the candidate ones [9]. The least-squares method is a basic technique for parameter estimation [10]. Karl Friedrich Gauss formulated the principle of least squares at the end of the eighteenth century and used it to determine the orbits of planets and asteroids. Gauss stated that, according to this principle, the unknown parameters of a mathematical model should be chosen in such a way that the sum of the squares of the differences between the actually observed and the computed values, multiplied by numbers that measure the degree of precision, is a minimum. The least-squares method can be applied to a large variety of problems. In adaptive control the observations are obtained sequentially in real time. It is then desirable to make the computations recursively to save computation time. Computation of the least-squares estimation can be arranged in such a way that the results obtained at time t-1 can be used to get the estimates at time t. The computation to the least-squares problem is then be rewritten in a recursive form called "recursive least-squares estimation" (RLS) [11].

RLS methods have been widely used with several advantages such as easy numerical solution and fast parameter convergence [12]-[15]. The method of stochastic approximations has excellent computational characteristics, but slow convergence, which necessitated recycling of data to achieve adequate models. It is also subject to considerable bias in the presence of noise, and poor reliability. The recursive instrumental variables approach provides unbiased estimation with fast convergence, but lower reliability than achieved by RLS [12]. The RLS identification method is very reliable with rapid convergence and it produces accurate models. It also gives consistent modeling accuracy over a wide range of operating conditions [15].

Although RLS has good performance on the parameter estimation, there are some initial settings must be decided before staring the computation. It is not easy to obtain an optimal set of initial values. In this paper, Evolutionary Computation is applied to search optimal values of the RLS operational factors of the system identification model. Evolutionary Computation [16]-[20] approaches such as Evolutionary Programming (EP), Evolution Algorithm (EA), Genetic Algorithm (GA), and Genetic Programming (GP) are computational models of natural evolutionary processes as key elements in the design and implementation of computer-based problem solving systems. The common conceptual base is simulating the evolution of individual structures via processes of selection and perturbation. These processes depend on the perceived performance (fitness) of the individual structures as defined by the environments. In this paper, GA and EA are

This work was supported by the National Science Council, Taiwan, ROC, under Grant NSC92-2213-E-019-005.

implemented in system identification scheme and are described briefly. Comparisons on convergence and accuracy of different estimation methods and evolutionary computation are given.

II. RECURSIVE ESTIMATION METHODS

In this section four different parameter estimation methods, RLS, RLS with forgetting factor, Stochastic Approximation (SA), and Projection Algorithm (PA), are utilized to system identification. State equation of a simple nonlinear system is given as follow

$$X(t+1) = AX(t) + CX_{nl}(t) + Bu(t)$$
$$= \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} X(t) + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} x_{nl}(t) + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u(t) \quad (1)$$

where a_j, b_i and c_i are the system's parameters to be identified, X is $(x_1, x_2)^T$, x_{nl} is x_1x_2, and u is the control signal $u(t) = 1 \; for \; t \geq 0$. The real values of the system are given as follows

$$A = \begin{bmatrix} -0.5 & 6.0 \\ -6.0 & -0.5 \end{bmatrix}, C = \begin{bmatrix} 0.5 \\ 1.0 \end{bmatrix}, B = \begin{bmatrix} 1.5 \\ 2.0 \end{bmatrix}.$$

A. Recursive least-squares estimation

The recursive calculations are summarized in the following equations:

$$K(t) = P(t)\varphi(t) = P(t-1)\varphi(t)[I + \phi^T(t)P(t-1)]\varphi(t)]^{-1}$$
$$P(t) = [I - K(t)\varphi^T(t)]P(t-1)$$
$$\hat{\theta}_i(t) = \hat{\theta}_i(t-1) + K(t)[y(t) - \varphi^T(t)\hat{\theta}_i(t-1)] \quad (2)$$

where $\varphi^T = [x_1, x_2, x_1x_2, u]$, P(0)=α²*I, α=1000,

and $\hat{\theta}_1 = \begin{bmatrix} a_1 \\ a_2 \\ c_1 \\ b_1 \end{bmatrix}$, $\hat{\theta}_2 = \begin{bmatrix} a_3 \\ a_4 \\ c_2 \\ b_2 \end{bmatrix}$.

Table I shows the results at 50 cycles. Some parameters have not converged yet. Parameters estimation convergence occurs in 80 cycles as shown in Figure 1.

TABLE I
. Results at 50 cycles.

a_1	a_2	a_3	a_4
-0.50001	5.9999	-5.9999	-0.49995
c_1	c_2	b_1	b_2
0.50013	0.99993	1.5000	1.9999

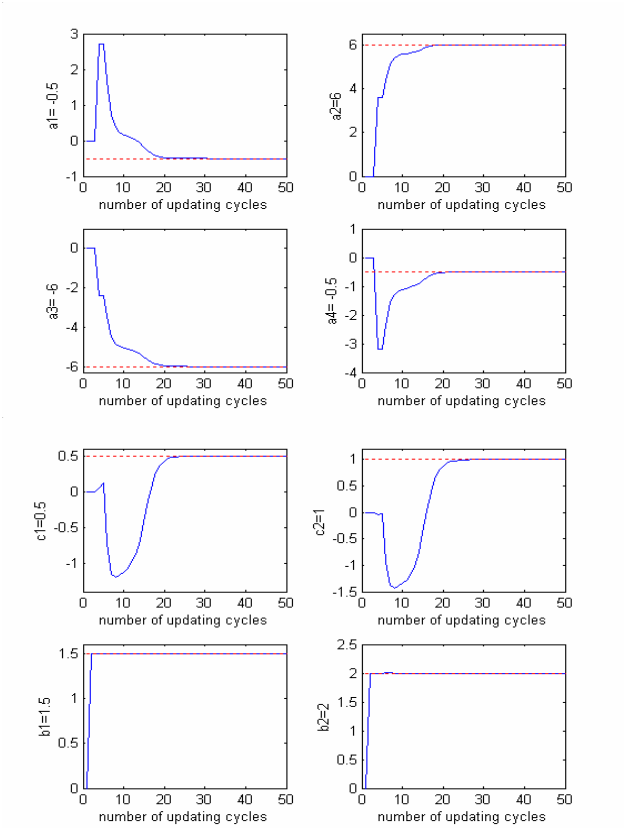


Figure 1. Parameter estimation using RLS.

B. RLS with exponential forgetting factor

In [21] a time-varying weighting of the data is introduced in the loss function. The most recent data is given unit weight, but data that is n time units old is weighted by λ^n. The method is therefore called "RLS with exponential forgetting". The calculations are then modified as follows.

$$K(t) = P(t)\varphi(t) = P(t-1)\varphi(t)[\lambda I + \phi^T(t)P(t-1)]\varphi(t)]^{-1}$$
$$P(t) = [I - K(t)\varphi^T(t)]P(t-1)/\lambda$$
$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)[y(t) - \varphi^T(t)\hat{\theta}(t-1)] \quad (3)$$

where λ is the forgetting factor and is set to 0.5. All parameters are obtained in less than 30 cycles, as in Figure 2, and the accuracy is better than the conventional RLS.

C. Stochastic approximation (SA)

Stochastic approximation is a simplified RLS method and is given in the following equations:

$$y(t) = \varphi^T(t)\theta(t-1) + n(t)$$
$$P(t) = [\sum_{i=1}^{t} \varphi^T(i)\varphi(i)]^{-1}\gamma$$
$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)[y(t) - \varphi^T(t)\hat{\theta}(t-1)] \quad (4)$$

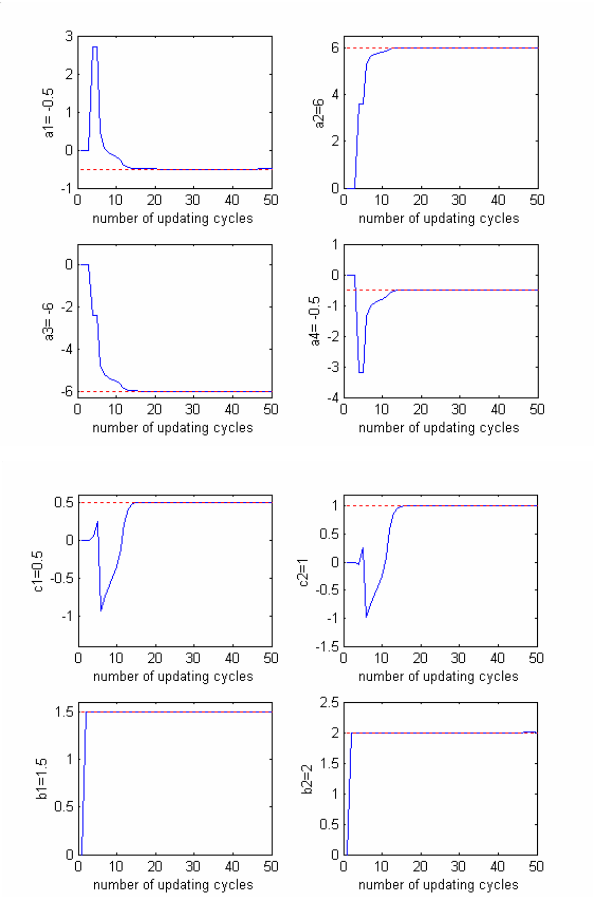


Figure 2. Parameter estimation using RLS with exponential forgetting factor.

where $n(t)$ is a random noise with zero mean, γ is an adjustable factor. The factor γ will affect estimation performance and the speed of convergence. SA has less computational complexity. Computational time of the SA is less than the RLS but its accuracy is worse than RLS. This disadvantage can be overcome by adding GA or EA in the parameter estimation scheme, which will be discussed in Section III.

D. Projection algorithm (PA)

Projection algorithm is developed from Kaczmarz's algorithm [22]. Parameter estimation is given in the following equation:

$$\hat{\theta}(t)=\hat{\theta}(t-1)+\frac{\gamma\varphi(t)}{\alpha+\varphi^T(t)\varphi(t)}[y(t)-\varphi^T(t)\hat{\theta}(t-1)] \qquad (5)$$

where $\alpha \geq 0$ and $0 \leq \gamma \leq 2$. PA has two factors that need to be adjusted, α and γ. Both factors will affect the performance and convergence of parameter estimation. PA has less computational complexity than RLS and SA. But with improper values of α and γ, such as α=6 and γ=1.2, estimation takes more than 4000 cycles to converge.

III. EVOLUTIONARY COMPUTATION

In previous sections the operational factors are chosen by trial and error. Since GAs have the potential for global optimization, they are suitable for determination of the factors, λ , γ, and α, which give accurate desired system performance. GAs are search and optimization algorithms based on the principle of natural evolution and population genetics. The basic principles of GAs were first proposed by Holland [18]. GA presumes that the potential solution of any problem is an individual and can be represented by a set of parameters. A positive value, generally known as a fitness value, is used to reflect the degree of "goodness" of the chromosome for the problem which would be highly related with its objective value [23].

Throughout a genetic evolution, the fitter chromosome has a tendency to yield good quality offspring which means a better solution to any problem. In a practical GA application, a population pool of chromosomes has to be installed and these can be randomly set initially. In each cycle of genetic operation, termed as an evolving process a subsequent generation is created from the chromosomes in the current population. This can only succeed if a group of these chromosomes, generally called "parents" or a collection term "mating pool" is selected via a specific selection routine. The genes of the parents are mixed and recombined for the production of offspring in the next generation. It is expected that from this process of evolution (manipulation of genes), the "better" chromosome will create a larger number of offspring, and thus has a higher chance of surviving in the subsequent generation, emulating the survival-of-the-fittest mechanism in nature. A scheme called Roulette Wheel Selection [24] is one of the most common selection techniques and is used in this paper.

The cycle of evolution is repeated until a desired termination criterion is reached. This criterion can also be set by the number of evolution cycles (computational runs), or the amount of variation of individuals between different generations, or a pre-defined value of fitness. In order to facilitate the evolution cycle, two fundamental operators: Crossover and Mutation are required, although the selection routine can be termed as the other operator. In this paper a real-value type GA, as called Evolutionary Algorithm, is used as follows:

Crossover: $w_1^n = w_1 + \sigma(w_1 - w_2), w_2^n = w_2 - \sigma(w_1 - w_2)$;

Mutation: $w^n = w + S \cdot random\_noise$;

where w_i is the ith chromosome, σ is a random value between -0.5 to +0.5, *random_noise* is -0.5 to +0.5, S=0.2. The choice of mutation rate p_m and crossover rate p_c as the control parameters can be a complex nonlinear optimization problem to solve. Furthermore, their settings are critically dependent upon the nature of the objective function.

A. RLS with exponential forgetting factor and EA

In here, population size is 10, crossover rate is 0.8, and mutation rate is 0.01. The fitness function is defined as

$$\text{Fitness function}=\left(\sum_{i=1}^{8} E\right)^{-1} \tag{6}$$

$$E=\sum_{t=1}^{N} e_{a1}+\sum_{t=1}^{N} e_{a2}+\sum_{t=1}^{N} e_{a3}+\sum_{t=1}^{N} e_{a4}+\sum_{t=1}^{N} e_{c1}+\sum_{t=1}^{N} e_{c2}+\sum_{t=1}^{N} e_{b1}+\sum_{t=1}^{N} e_{b2}$$

where e is the error of the estimated value. After 27 generations optimal forgetting factor is obtained and $\lambda = 0.00012253$. The fitness values versus number of generations is shown in Figure 3. All parameters are obtained in less than 10 cycles, as shown in Figure 4, with same accuracy as in Subsection II-*B*.

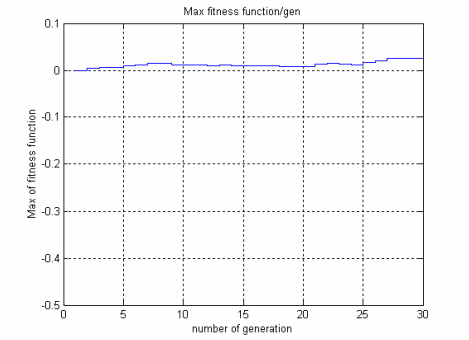


Figure 3. Genetic searching.

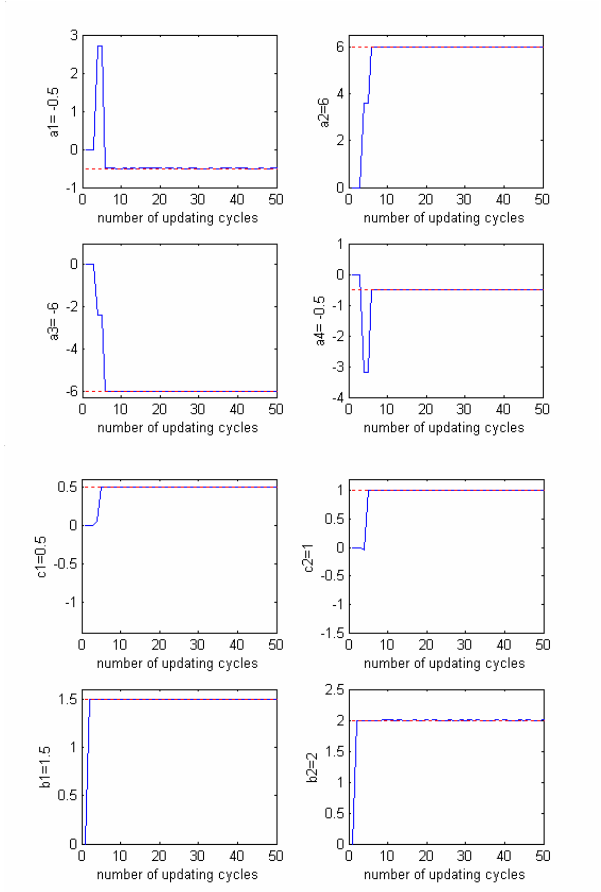


Figure 4. RLS with λ=0.00012253.

B. SA and EA

The constant factor γ in SA is searched by EA. Crossover and mutation rules use the same calculations as mentioned previously. Fitness function uses (6). The searched value of γ is 11.3028. Parameter estimations are shown in Figure 5. Convergence performance of parameters of matrices *A* and *B* is better than *C*. However, the overshoot is quite a large of the a_i's and b_i's in the beginning cycles. Estimated values of b_1 and b_2 at 6th cycle are 626 and 778 respectively. To overcome this problem, different fitness function is utilized to search optimal γ as follow:

$$\text{Fitness function}=\left(\sum_{i=1}^{8} M_o\right)^{-1} \tag{7}$$

where M_o is the maximal overshoot value of each parameter in estimation process. The searched γ is 3.5041. Estimation results are shown in Figure 6. Overshoot has dropped dramatically. But the convergence is slower than before.

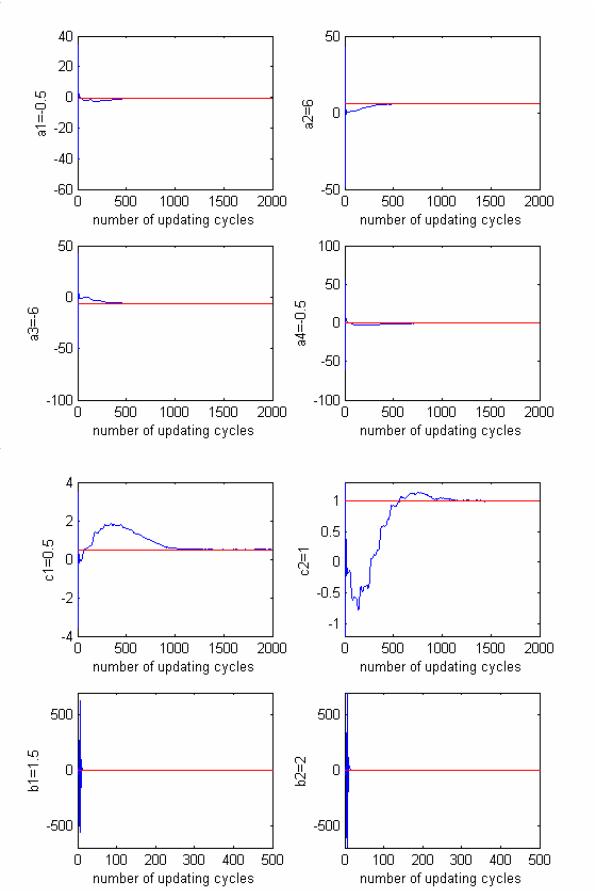


Figure 5. SA with γ=11.3028.

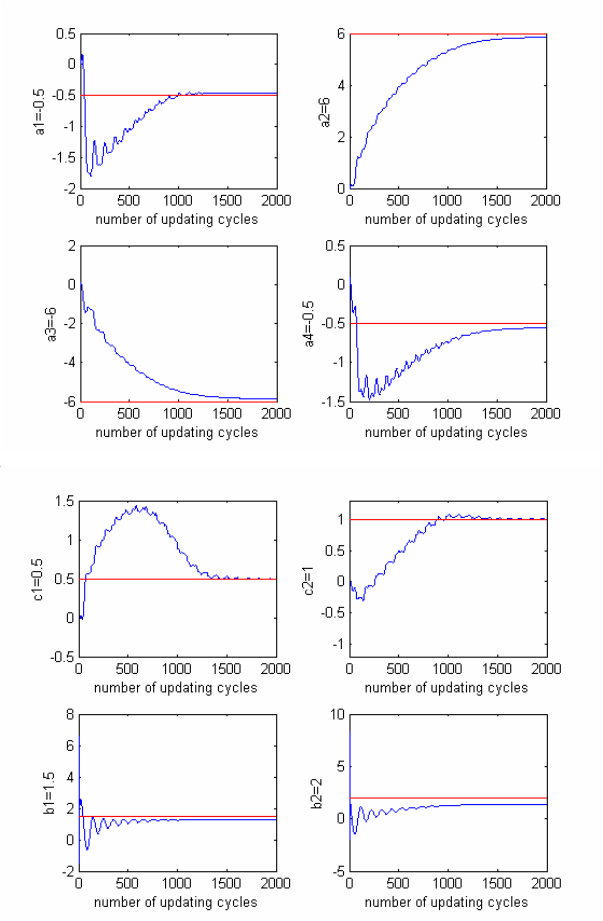
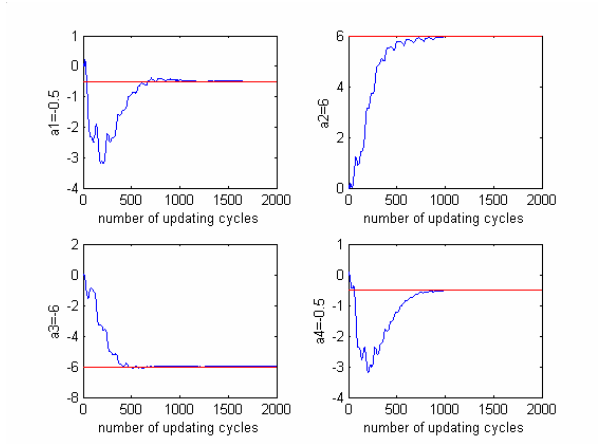


Figure 6. SA with γ=3.5041.

To avoid large overshoot and slow convergence, combination of (6) and (7) is used in next test. In the beginning cycles, (6) is activated. After 10 cycles (7) is then enabled. The values of γ are: 0 to 10cycles:γ=3.5041, 11 to 2000 cycles: γ=11.3028. Estimations are shown in Figure 7. With noise added the estimated values are still very close to the real values, as shown in Table II.



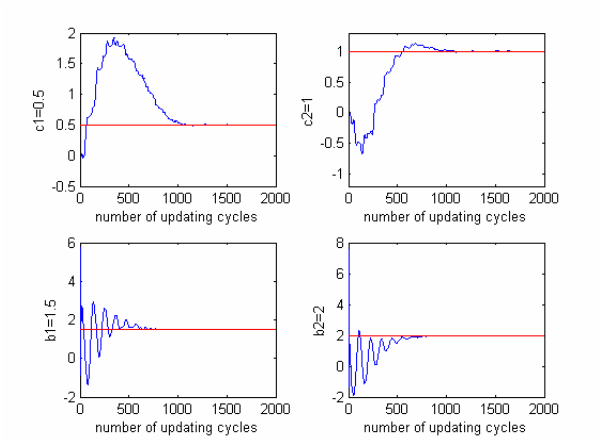


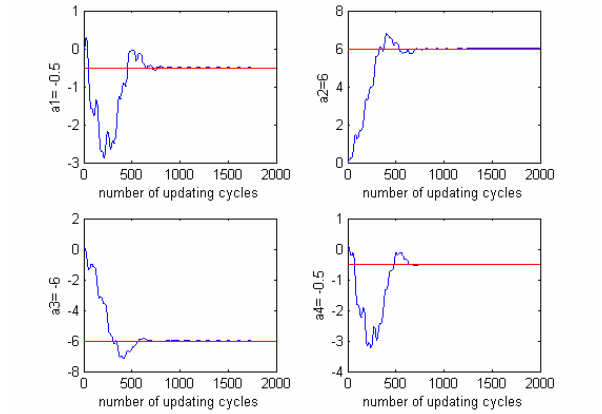
Figure 7. SA with 0 to 10cycles:γ=3.5041, 11 to 2000 cycles: γ=11.3028

TABLE II
Estimations at 2000 cycles.

a_1	a_2	a_3	a_4
-0.49982	6.0002	-5.9997	-0.5000
c_1	c_2	b_1	b_2
0.49993	1.0002	1.4997	2.0005

C. PA and EA

The factors γ and α in PA are searched by EA. Fitness function uses (6). The searched values are γ= 0.1017 and α=0.0012. Estimations are shown in Figure 8. All estimations converge to its real values in 1800 cycles as shown in Table III. Compare to Subsection II-*D*, convergence performance is improved with EA added.



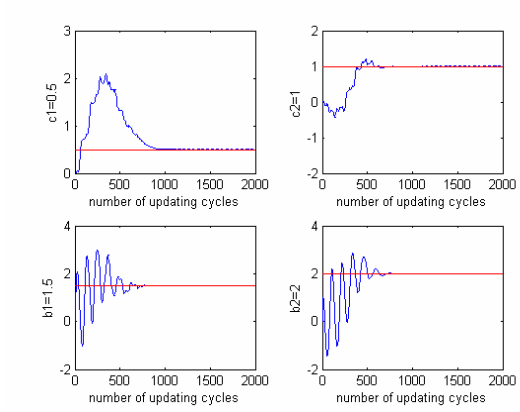


Figure 8. PA with γ=0.1017, α=0.0012.

TABLE III
Estimations at 2000 cycles.

a_1	a_2	a_3	a_4
-0.5000	6.0000	-6.0000	-0.5000
c_1	c_2	b_1	b_2
0.5000	1.0000	1.5000	2.0000

IV. CONCLUSION

Recursive estimation methods have been widely used with several advantages such as easy numerical solution and fast parameter convergence. But there are some initial settings that must be decided before staring the computation. It is not easy to obtain an optimal set of initial factors. In this paper, Evolutionary Computation is applied to search optimal values of the operational factors of the system identification model. In this paper four parameter estimation methods are presented. A simple nonlinear system is used to test the proposed methods. Theoretically, this nonlinear system can be identified. However, in real estimations, the convergence and accuracy are affected by the operational factors. Proper fitness functions are proposed to search adequate operational factors in parameter estimations. From simulation results, the parameters of the unknown nonlinear system can be obtained. With EA the combination scheme of the system identification model has shown that the proposed method has fast convergence rate without losing accuracy.

REFENRENCES

[1] T.C. Hsia, *System Identification*. D.C. Heath and Co., USA, 1977.
[2] L.D. Landau, *System Identification and Control Design,* Prentice Hall, Englewood Cliffs, N.J., USA, 1990.
[3] T. Söderström and P. Stoica, *System Identification*, Prentice Hall, Cambridge, U.K., 1989.
[4] T. Kara and İ Eker, "Identification of Nonlinear Systems for Feedback Control", *Proceedings of the Forth GAP Engineering Congress,* Şanlıurfa, Turkey, pp.520-526, June 06-08, 2002.
[5] R. Haber and H. Unbehausen, "Structure identification of nonlinear dynamic systems – a survey of input/output approaches," *Automatica*, vol. 24, pp. 651–677, 1990.
[6] J.C. G´ometz and M. Basualdo, "Nonlinear identification of multicomponent batch distillation processes," *Proceedings of the IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2000*, pp. 989–994, 2000.
[7] J. Sj¨oberg, Q. Zhang, L. Ljung, A. Benveniste, B. Deylon, P.-Y. Glorennec, H. Hjalmarsson, and J. Juditsky, "Nonlinear black–box modeling in system identification: a unified overview," *Automatica*, vol. 31, pp. 1691–1724, 1995.
[8] L. Benvenuti, A. De Santis, and L. Farina, "On model consistency in compartmental systems identification," *Automatica*, vol. 38, pp. 1969-1976, 2002.
[9] L. Ljung, *System Identification: Theory for the User*, Prentice Hall, Englewood Cliffs, N.J., USA. 1987.
[10] C.L. Lawson and R.J. Hanson, *Solving Least Squares Problems*, Englewood Cliffs, N.J.: Prentice-Hall, 1974.
[11] L. Ljung and T. Soderstrom, *Theory and Practice of Recursive Identification*, Cambridge, Mass.: MIT Press, 1983.
[12] P.E. Wellstead and M.B. Zarrop, *Self-tuning Systems: Control and Signal Processing*, John Wiley and Sons, U.K., 1991.
[13] C.C. Hang, T.H. Lee, and K.H. Weng, *Adaptive Control*, Instrument Society of America, USA, 1993.
[14] B. Huang, "Process identification based on last principal component analysis," *Journal of Process Control*, vol. 11, pp. 19-33, 2001.
[15] D.G. Robertson and J.H. Lee, "On the use of constraints in least squares estimation and control," *Automatica*, vol. 38, pp. 1113-1123, 2002.
[16] L.J. Fogel, A.J. Owens, and M.J. Walsh, "Artificial intelligence through a simulation of evolution," *Proc. of the 2nd Cybernetics Science Symp.*, pp. 131–155, 1965, reprinted in D.B. Fogel (eds.), *Evolutionary Computation, The Fossil Record*, pp. 230–254, IEEE Press, 1995.
[17] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
[18] J.H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.
[19] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.
[20] H.-P. Schwefel, *Evolution and Optimum Seeking*, J. Wiley, 1995.
[21] R.L. Plackett, "Some Theorems in Least Squares," *Biometrika,* vol. 37, pp. 149-157, 1950.
[22] S. Kaczmarz, "Approximate solution of systems of linear equations." *Int. J. Control,* vol. 57, pp. 1269-1271, 1993.
[23] K.F. Man, K.S. Tang, and S. Kwong, *Genetic algorithms*. London: Springer-Verlag, 1998.
[24] C.T. Lin and C.S. Lee, *Neural fuzzy systems*, Upper Saddle River, NJ: Prentice Hall, 1996.