[Article ID] 1003- 6326(2002)02- 0349- 07

Identification of ARMAX based on genetic algorithm[①]

HE Shang-hong(贺尚红)[1,2], LI Xu-yu(李旭宇)[2], ZHONG Jue(钟 掘)[2]
(1. Department of Mechanical and Electronic Engineering,
Changsha Communications University, Changsha 410076, China;
2. College of Mechanical and Electronic Engineering, Central South University,
Changsha 410083, China)

[Abstract] On the basis of genetic algorithm, an intelligent search approach to determination of parameters of ARMAX (Autor Regressive Moving Average model with external input) processes was proposed. By representing the system with pole and zero pairs and repairing illegal chromosomes, the search space is limited to stable schemes. In calculation of objective function the "shifted data window" was designed, so that every input-output pair is used to guide the evolution and the "Data Saturation" is avoided. To prevent premature convergence, the adaptive fitness function was introduced, the conventional crossover and mutation operator was modified and the "catastrophic mutation" which is based on Metropolis mechanism was adopted. So the performance of convergence to the global optimum is improved. The validity and efficiency of proposed algorithm were illustrated by simulated results.

[Key words] system identification; genetic algorithm; ARMAX process; optimum
[CLC number] TP271.61 **[Document code]** A

1 INTRODUCTION

System identification is the foundation of control engineering. In most cases, the parameters are estimated by least squares based algorithms[1]. All these techniques are based upon the assumption of a smooth search space with ever present derivatives, and are in essence local search techniques that search for the optimum by using a gradient-following technique. In actual system identification, the power spectrum of a test input is sometimes localized in a low frequency range or somewhat limited frequency range, then the least squares (LS) estimation tends to be ill-conditioned. So, they often fail in search for global optimum.

Genetic algorithm (GA) has been established as a variable search technique which is based on natural evolution and selection[2]. A genetic algorithm is a parallel, global search technique that emulates natural genetic operators. It needs not assume that the search space is differentiable or continuous. Some researchers applied GA to linear or nonlinear dynamic system identification[3~8]. All these researches assume that the structure of system and the interval of model parameters are known. But in many cases, it is difficult to obtain the parameter intervals. In this paper, the poles and zeros of ARMAX transfer function are represented by the average of pairs (α) and their deviation from average (β), so, for stable systems, the search space should be limited to $\alpha(\beta)\in(-1, +1)$.

GA has been theoretically and empirically proven to provide robust search in complex spaces. However, there is still a pressing need to improve our understanding of the foundations of GA. One of primary complaints toward GA is the occurrence of premature convergence. Premature convergence can be subdued through variation of crossover, selection and mutation, and through alterations such as population size, crossover rate, and mutation rate[9,10]. The work presented here introduces adaptive fitness function and dynamic crossover rate, adopts the "catastrophic mutation" which is based on Metropolis mechanism, and modifies the regular crossover and mutation operation. So the performance of convergence to the global optimum is improved.

2 PRINCIPLE OF SYSTEM IDENTIFICATION

Consider a system described by an ARMAX model[1]

$$A(q^{-1})y(t) = B(q^{-1})u(t-d) + C(q^{-1})e(t) \quad (1)$$

where $A(q^{-1}) = 1 + a_1q^{-1} + a_2q^{-2} + \cdots + a_nq^{-n}$, $B(q^{-1}) = b_0(1 + b_1q^{-1} + b_2q^{-2} + \cdots + b_mq^{-m})$, $C(q^{-1}) = c_0(1 + c_1q^{-1} + c_2q^{-2} + \cdots + c_lq^{-l})$. $A(q^{-1})$ and $B(q^{-1})$ are irreducible and $A(q^{-1})$ is asymptotic stable, $e(t)$ is a normally distributed random sequence with zero mean and variance σ_e^2.

The true parameters are $\theta = [a_n\ a_{n-1}\ a_{n-2} \cdots a_1\ b_m\ b_{m-1} \cdots b_1\ b_0\ d]^T$.

① [Foundation item] Project (59835170) supported by the National Natural Science Foundation of China, project (00JJY2051) supported by the Natural Science Foundation of Hunan, China [Received date] 2001- 01- 22; [Accepted date] 2001- 05- 14

The estimated parameters are $\theta^* = [a_n^* \ a_{n-1}^* \ a_{n-2}^* \cdots a_1^* \ b_m^* \ b_{m-1}^* \cdots b_1^* \ b_0^* \ d^*]^l$.

The output of a deterministic system driven by the actual input $u(t)$ and $A^*(q^{-1})$ and $B^*(q^{-1})$ are

$$y^*(k) = -a_n^* y(k-n) - a_{n-1}^* y(k-n+1) - \cdots - a_1^* y(k-1) + b_0^*[b_m^* u \cdot (k-m-d^*) + b_{m-1}^* u(k-m-d^*+1) + \cdots b_1^* u(k-d^*-1) + u(k-d^*)] \quad (2)$$

Define optimal function

$$J = \sqrt{\sum_{k=1}^{w} [y^*(k) - y(k)]^2} \quad (3)$$

where $y(k)$ is actual output, and w is the length of data window.

By minimizing J, the parameters are estimated.

3 ANALYSIS OF SEARCH SPACE

The genetic algorithm is used for searching optimal deterministic system according to optimal function (3). In many cases, the search range of parameters is defined through prior knowledge about actual ARMAX model. If the search space is too wide, many illegal schemes will be in search space, and if the search space is too narrow, it is possible to miss optimal scheme. So, for enhancing the efficiency of search program, we should restrict the search space as narrow as possible, but ensure the estimated models are stable. From polynomials $A(q^{-1})$ and $B(q^{-1})$, it is difficult to obtain the feasible search space directly unless we have prior knowledge about coefficients of $A(q^{-1})$ and $B(q^{-1})$. So, $A(q^{-1})$ and $B(q^{-1})$ should be represented as poles and zeros:

$$A(q^{-1}) = (1-p_1q^{-1})(1-p_2q^{-1})\cdots(1-p_nq^{-1}) \quad (4)$$

$$B(q^{-1}) = b_0(1-z_1q^{-1})(1-z_2q^{-1})\cdots(1-z_mq^{-1}) \quad (5)$$

where $p_i(i=1,2,\cdots n)$ and $z_j(j=1,2,\cdots m)$ are poles and zeros of system respectively.

For a stable minimum phase system, the poles and zeros are inside the unit circle (Fig. 1), therefore the search space of poles and zeros should be limited to be the unit circle. If the poles and zeros are real, the search space will be limited to $-1 < p_i(z_j) < 1$, which is convenient for GA. But in many cases, p_i and z_j may be complex, which must appear in conjugate pairs. For implementation of GA, p_i or z_j must be represented as real.

Taking the poles as example, if n is even, $p_i(i=1,2,\cdots n)$ can be represented as pairs $p_{j,j+1}(j=\frac{i}{2}, i=2,4,\cdots,n-2,n)$, so

$$p_{j,j+1} = (p_i + p_{i+1})/2 \pm \left|\frac{p_i - p_{i+1}}{2}\right|$$

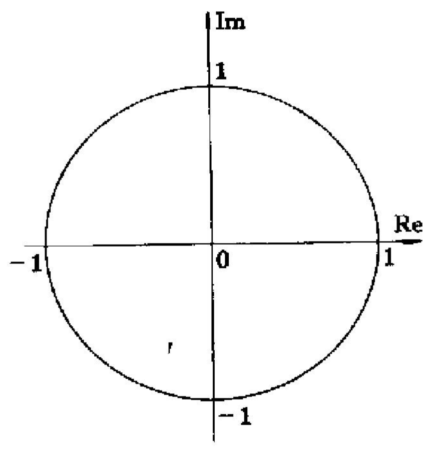

Fig. 1 Complex plane

(real pole) (6)

$$p_{j,j+1} = (p_i + p_{i+1})/2 \pm \left|\frac{p_i - p_{i+1}}{2}\right| j$$

(complex pole) (7)

Define

$$\alpha_j^{(p)} = (p_i + p_{i+1})/2 \quad (8)$$

$$\beta_j^{(p)} = \pm\left|\frac{p_i - p_{i+1}}{2}\right| \quad (9)$$

So

$$p_{j,j+1} = \alpha_i^{(p)} \pm \beta_i^{(p)} \quad \text{(real pole)} \quad (10)$$

$$p_{j,j+1} = \alpha_i^{(p)} \pm \beta_i^{(p)} j \quad \text{(complex pole)} \quad (11)$$

When $\beta_j^{(p)} \geqslant 0$, $p_{j,j+1}$ is real pair, otherwise $p_{j,j+1}$ is conjugate complex pair.

According to (10) and (11), we can represent poles by a plane where the horizontal axis represents the average value and the vertical axis the deviation from the average. Every point in this new plane will represent two points, complex conjugate poles if it is in the lower half plane and two real poles if it is in upper half plane. That turns the search space in Fig. 1 into S_1 in Fig. 2. If n is odd, the last pole p_n should be excluded, and be satisfied by $-1 < p_n < 1$.

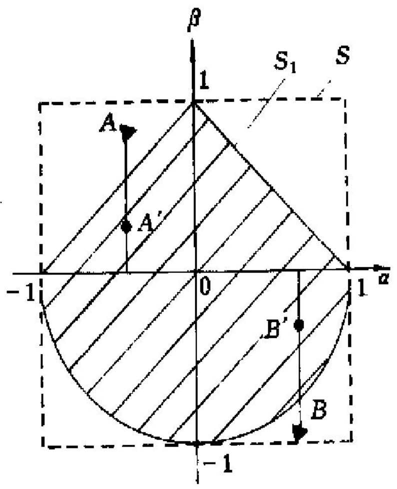

Fig. 2 Pole (zero) pair plane

For minimum phase stable system, the zeros should be treated as poles. We can identify system model by searching optimum pole and zero pairs instead of coefficients of polynomials $A(q^{-1})$ and $B(q^{-1})$. So, we need not determine the interval of coefficients of $A(q^{-1})$ and $B(q^{-1})$.

Nonlinear search space S_1 in Fig. 2 is difficult to GA. For convenience, we use rectangle S in Fig. 2 as search space of GA in which $S_1/S = 0.6425$. All schemes in $S-S_1$ which represent unstable models will be illegal. By repairing illegal schemes in $S-S_1$, the search space can be limited to S_1.

4 IMPLEMENTATION OF GA

The basic element processed by genetic algorithm is the string formed by concatenating substrings, each of which is coding of a parameter of the search space. Thus each string represents a possible solution to the problem. The genetic algorithm works with a set of strings, called the population. With the guide of the objective function, this population then evolves from generation to generation through the application of genetic operators. A genetic algorithm in its simplest form uses three operators: Reproduction, Crossover, and Mutation[2].

4.1 Coding of chromosome

GAs require the natural parameter set of the optimization problem to be coded as a finite-length alphabet[2,11]. We must face the trade-off problem between the length of coded string and the resolution of the parameter value. When evaluating the chromosome, the string should be encoded, so the coding and decoding process will waste the computation time and slow down the convergent rate. In this paper, we code chromosomes Gen by real value of parameters, which is pole or zero pair parameters $\alpha^{(p)}(\alpha^{(z)})$ and $\beta^{(p)}(\beta^{(z)})$, remanent pole p_n and z_m zero (n or m is odd), b_0 and delay d.

$$\begin{aligned} Gen = \; & \alpha_1^{(p)}\beta_1^{(p)}\ldots\alpha_j^{(p)}\beta_j^{(p)}\ldots\alpha^{(p)}_{\frac{(n-1)}{2}}\beta^{(p)}_{\frac{(n-1)}{2}}p_n \\ & \ldots\alpha_1^{(z)}\beta_1^{(z)}\ldots\alpha_i^{(z)}\beta_i^{(z)}\ldots \\ & \alpha^{(z)}_{\frac{(m-1)}{2}}\beta^{(z)}_{\frac{(m-1)}{2}}z_m b_0 d \end{aligned} \tag{12}$$

The chromosome can be viewed as a vector including all parameters, so the decoding is not necessary. Primitive chromosomes are generated randomly in viable search space S_1.

4.2 Adaptive fitness function

The fitness function needs to be maximized so it is chosen as[9]

$$F(j) = C_{\max} - J(j) \tag{13}$$

where $J(j)$ is the objective function of individual j defined by (3). $C_{\max}$ is a bias term needed to ensure a positive fitness, which can be determined empirically or chosen as maximum of $J(j)$ of accumulated generations.

When calculating $J(j)$ by (3), the window size w and the position of w in input-output sequence play an important role in calculation of fitness. It has been shown via simulation that the variance of the parameter estimates reduces as the window size w increases. However, execution slows down as the window size increases. So the window size must be compromised. On the other hand, in order to use every input-output datum to guide the evolution process, the position of window shifts a time step when evolving several generations, with an effect akin to that of forgetting factor in LS based algorithms[1], so the "Data Saturation" and abrupt disturbance are avoided.

$C_{\max}$ is another important factor of $F(j)$. When $C_{\max}$ is chosen to be too high, the fitness of chromosomes inclines to uniform, so the search process will slow down. If $C_{\max}$ is too small, the fitness of chromosomes will differ remarkably, so the GA leads to dominant allele fixation in the loci prior to the discovery of optimal or near optimal solution. To ensure variance of chromosomes in population and high searching speed, we introduce adaptive fitness function as follow:

$$\begin{aligned} F(j) = \; & \max[J(j)] - J(j) + \\ & q(k)\{\max[J(j)] - \min[J(j)]\} \end{aligned} \tag{14}$$

where $q(k)$ is the adaptive factor which is the function of generation k. The ratio of offspring number of optimal chromosome to worst one in population is: $\frac{\max|F(j)|}{\min|F(j)|} = \frac{1+q(k)}{q(k)}$. $q(k)$ is limited to $q_1 \leqslant q(k) \leqslant q_2$, so $\frac{1+q_2}{q_2} \leqslant \frac{\max|F(j)|}{\min|F(j)|} \leqslant \frac{1+q_1}{q_1}$. At the beginning of evolution, $\frac{\max|F(j)|}{\min|F(j)|}$ should be smaller, the selection pressure is lower and the variability of population is ensured. As the evolution advances, $\frac{\max|F(j)|}{\min|F(j)|}$ must increase to enhance the selection pressure to converge at the end. So we define $q(k)$ as linear decreasing function of generation k

$$q(k) = d_1 k + d_2 \tag{15}$$

where $d_1 = \frac{q_1 - q_2}{k_{\max} - 1}$, $d_2 = q_2 - d_1$, $k_{\max}$ is the determined number of generations, q_1 and q_2 are minimum and maximum factor respectively.

In simple GA, the fitness is scaled to ensure proper distinction in populations[2,9]. The proposed adaptive fitness function defined in (14) not only adjust the selection pressure as evolution proceeding but also scale the fitness function.

4.3 GA operators

4.3.1 Reproduction

Reproduction is based on the principle of the "survival of the fittest". A fitness, $F(j)$, is assigned to each individual in the population where high numbers mean good fit. The number of offspring for each individual is proportional to normalized fitness. The strings with higher-than-average fitness will have more than one offspring, and those with below-average fitness will have less than one offspring on the average. To achieve this, the strings are selected according to "roulette wheel"[2].

In this paper, the algorithm keeps track of the best string in the population and the best one is put into the new population by removing the worst string. This procedure is called "elitism"[9].

4.3.2 Crossover and mutation

Reproduction directs the search toward the best existing individuals but does not create any new individuals. In nature, an offspring is rarely an exact clone of a parent, it usually has two parents and inherits genes from both. The main operator to work on the parents is crossover, which is applied with a certain probability, called crossover rate (p_c). This operator takes valuable information from both parents and combines it to find a highly fit individual. In the paper, numerical crossover is adopted[9]. Two strings from the reproduced population are mated at random, the parents υ_1 and υ_2 give two new offsprings υ'_1 and υ'_2 as follow

$$\upsilon'_1 = \gamma\upsilon_1 + (1-\gamma)\upsilon_2 \qquad (16)$$

$$\upsilon'_2 = \gamma\upsilon_2 + (1-\gamma)\upsilon_1 \qquad (16')$$

where γ is a uniformly distributed random number between 0 and 1.

The variability of population is effected by Crossover rate (p_c). As crossover rate increases, the variability is increased, and the search process converges more slowly. As crossover rate decreases, the variability is decreased, so the probability of premature convergence is increased. The crossover rate should decrease as the generation number is increased.

$$p_c = (p_{c\max}k_{\max} - p_{c\min})/(k_{\max} - 1) + [(p_{c\min} - p_{c\max})/(k_{\max} - 1)]k \qquad (17)$$

where $p_{c\min}$ and $p_{c\max}$ are minimum and maximum crossover rate, $k_{\max}$ is the determined number of generation, and k is generation number. To explore the search space, when the optimal fitness function does not change in several successive generations, which is called "Evolutionary Stagnation", the crossover rate should be higher than one calculated by (17).

When crossover can not ensure the variability in population because of dominant allele fixation in population prior to the discovery of optimal or near optimal solution, the new genetic strings should be introduced. Mutation is thought of as disruptive tool when the population converges prematurely to a local optimum, in which case a high mutation rate is helpful. However a high level of mutation yields an essentially random search. The difficulty is in seeking the balance between exploration and fine-tuning. In normal GA, mutation is a successive procedure after crossover. In this paper, mutation and crossover are implemented at the same time. When the two parents of crossover have equal fitness, in which case crossover will be void, we mutate any of them completely. When the fitness of parents are different remarkably, mutation is not necessary. If the fitness of two parents is approximately equal, the crossover can be thought of as a local search in an area surrounding that individual in a multimensional space. In this way, mutation can prevent inbreeding in population and insure higher efficiency of search process.

In variable coding, mutation operator simply replace any allele (x_k) of individuals with new allele (x'_k) as follow[9]:

$$x'_k = u^k_{\min} + r(u^k_{\max} - u^k_{\min}) \qquad (18)$$

where $u^k_{\min}$ and $u^k_{\max}$ is minimum and maximum value of gene respectively, r is a uniformly distributed random number in [0, 1].

It is shown via simulation that although we take so much measures, it is unavoidable that the premature convergence occurs. When the ratio of average fitness of population to maximum fitness is more than 0.95, which is defined to be premature convergence, the population becomes more homogeneous. In such situation, the schemata of population should be disrupted violently.

When premature convergence occurs we implement "catastrophic mutation" to population. As to any individual in premature population, we generate a new individual in search space S_1 randomly, but the new individual is accepted at the probability rate (p) described by Metropolis mechanism[12]

$$p = \begin{cases} 1 & \text{when} \quad J(j) \leqslant J(i) \\ \dfrac{1}{\exp\left(\dfrac{J(j) - J(i)}{T}\right)} & \text{otherwise} \end{cases} \qquad (19)$$

where $J(j)$, $J(i)$ are fitness of old and new individual respectively, T is the annealing temperature which reduces at linear function of generation number k.

$$T = (T_{\max}k_{\max} - T_{\min})/(k_{\max} - 1) + [(T_{\min} - T_{\max})/(k_{\max} - 1)]k \qquad (20)$$

where $T_{\min}$, $T_{\max}$ are minimum and maximum annealing temperature, $k_{\max}$ is the maximum generation number, k is the generation number.

With Metropolis rule, as search process approach to the end, the probability of replacement of old individual with new inferior one generated by (18) is decreased. As to any individual in population, the more the fitness of new inferior individual close to the old one, the higher the probability of acceptance is.

4.3.3 Mending of illegal chromosome

As to search space S of GA, the ratio $(S - S_1)/S$ is 0.3575, individuals in $S - S_1$ are illegal chromosomes, which cut down the efficiency of search process. In some literature, illegal individuals are punished, so they have lower probability of survival[9]. In this paper, we turn those individuals into legal ones by repairing some of schemata, so the efficiency of the algorithm is increased.

In Fig. 2, unstable pole pair $\alpha^{(p)}(k)$, $\beta^{(p)}(k)$ are repaired according to

$$\alpha^{(p)'}(k) = \alpha^{(p)}(k) \tag{21}$$

$$\beta^{(p)'}(k) = \begin{cases} \text{rand}\,(0,\ 1-|\alpha^{(p)'}(k)|) \\ \text{where}\quad \beta^{(p)}(k) > 0 \\ \text{rand}\,(-\sqrt{1-[\alpha^{(p)'}(k)]^2},\ 0) \\ \text{where}\quad \beta^{(p)}(k) < 0 \end{cases} \tag{22}$$

where rand$(x,\ y)$ is uniformly distributed random number in $[x,\ y]$. Unstable pair at upper plane (point "A") in Fig. 2 is repaired in upper plane in S_1 (point "A'"), and another one at lower plane (point "B") is repaicred in lower plane in S_1(point "B'"). To some extent, the modified individual is similar to the old one.

5 SIMULATED EXAMPLES

5.1 Example 1

The actual ARMAX process is assumed to be described by

$$A(q^{-1}) = 1 - 2.06q^{-1} + 1.7778q^{-2} - 0.5247q^{-3},$$

$$B(q^{-1}) = 1.0(1 + 1.5q^{-1} + 0.685q^{-2}),$$

$$d = 1,$$

$$b_0 = 1.0$$

The poles and zeros are:

Poles: $p_{1,2} = 0.75052 \pm 0.6127\text{j}$, $p_3 = 0.5590$ or pole pairs: $\alpha^{(p)} = 0.75052$, $\beta^{(p)} = -0.6127$, $p_3 = 0.5590$

Zeros: $z_{1,2} = -0.75 \pm 0.35\text{j}$ or zero pairs: $\alpha^{(z)} = -0.75$, $\beta^{(z)} = -0.35$

The input is inverse PRBS (Pseudo-Random Binary Sequence) contaminated by normally distributed random sequence with zero mean. The outputs are simulated with two kinds of noise: one is normally distributed white noise sequence, another is colored noise with the color filter: $C(q^{-1}) = 1.0 - 1.0q^{-1} + 0.2q^{-2}$. Fig. 3 shows the output with colored noise. Table 1 is the convergent results, which are evolved 5 000 generations. Fig. 4 shows the estimated parameters with generation. After about 1 200 generations, the algorithm converges to the true value for poles, but shows some bias for zeros. The reason that the

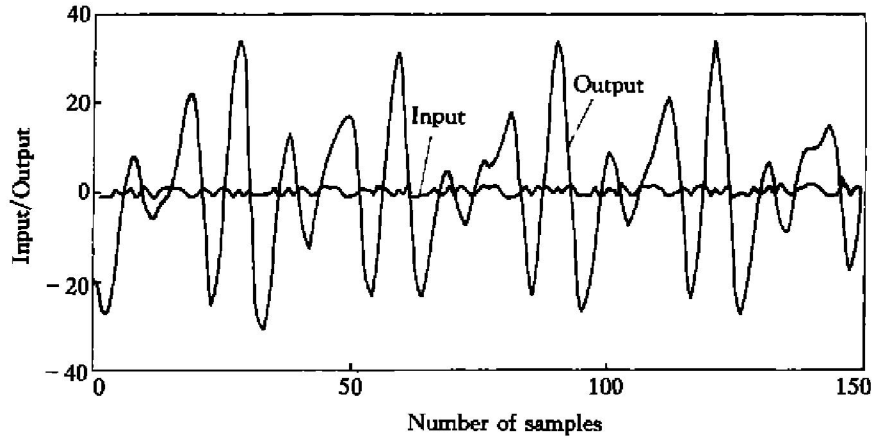

Fig. 3 Input and output sequence with colored output noise

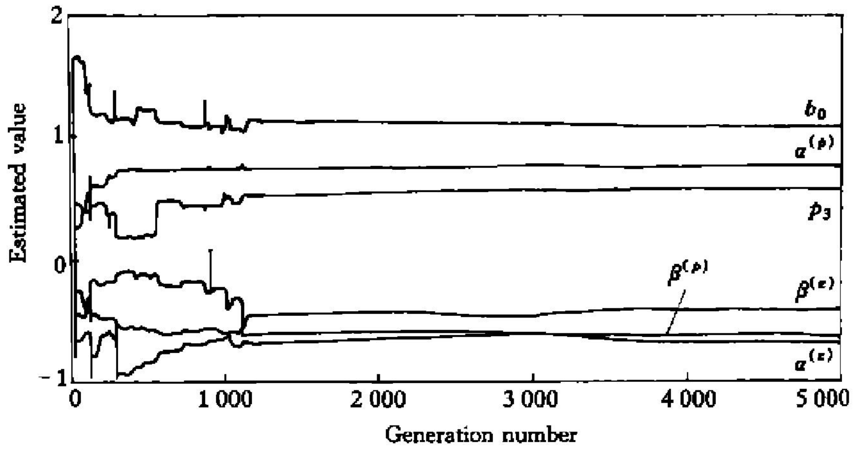

Fig. 4 GA estimation of simulated system

zeros are biased but the poles are not is that the steady-state gain is 16.5, so the objective function is less sensitive to changes in the zeros than in poles. It should emphasized that the algorithm does not necessarily converge to the optimum, but do best while learning to do better. Fig. 5 shows the objective function of optimal individual in population with generations. After about 2000 generations, the algorithm completes "rough search" process, and then turns to "fine tuning" process. It is shown that the convergent speed is excellent. Fig. 6 shows the average objective function of population. The average objective function oscillates with generations, which shows that the algorithm prematurely converges frequently, but the envelope curve of oscillating process declines with generations. That is the effect of "catastrophic mutation", which ensures that the algorithm escapes from premature convergence and converges to globe optimum effectively.

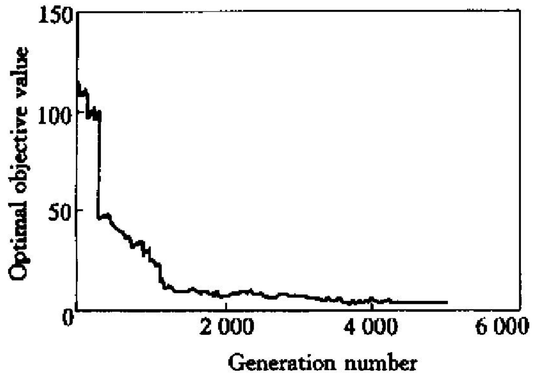

Fig. 5 Objective function of best individual in population

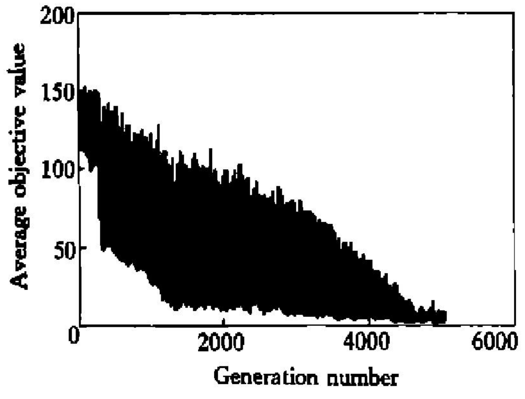

Fig. 6 Average objective function of population

5.2 Example 2

The actual ARMAX process is assumed to be described by[1]

$$A(q^{-1}) = 1 - 1.5q^{-1} + 0.7q^{-2},$$
$$B(q^{-1}) = 1 + 0.5q^{-1},$$
$$d = 1,$$
$$b_0 = 1.0$$

Poles: $p_{1,2} = 0.75 \pm 0.37j$ or pole pair: $\alpha^{(p)} = 0.75$, $\beta^{(p)} = -0.37$

Zero: $z_1 = -0.5$

The input and output noise are the same as example 1. Table 2 shows the estimates of the system with 3000 evolutionary generations, which shows that the estimated results are good agreement with true values.

6 CONCLUSIONS

In this paper it has been demonstrated how a genetic algorithm can be used to estimate ARMAX process, and how to improve the efficiency of GA. It has shown by simulated examples that

1) By turning poles (zeros) into pole (zero) pairs, we can limit the search space of GA to stable schemes. It is unnecessary to determine the scope of parameters, so the proposed algorithm is convenient for engineering applications.

2) The proposed method of modifying illegal chromosomes, which is based on similarity principle, avoids the invalid search in search space, so it improves the efficiency of algorithm.

3) The adaptive fitness and crossover rate can adjust the variability in population with evolutionary

Table 1 Simulated results of example 1

Type of noise	$\alpha^{(p)}$	$\beta^{(p)}$	p_3	b_0	$\alpha^{(z)}$	$\beta^{(z)}$	d	Objective function
White noise	0.7498	−0.6112	0.5642	1.1354	−0.6204	−0.4612	1	5.7351
Colored noise	0.7489	−0.6118	0.5622	1.0745	−0.6744	−0.4154	1	3.1398
True value	0.7505	−0.6127	0.5590	1.0000	−0.7500	−0.3500	1	

Table 2 Simulated results of example 2

Type of noise	$\alpha^{(p)}$	$\beta^{(p)}$	z_1	b_0	d	Objective function
White noise	0.7107	−0.3894	−0.3599	0.9860	1	4.3958
Colored noise	0.7329	−0.3782	−0.4181	0.9787	1	1.3672
True values	0.7500	−0.3700	−0.5000	1.0000	1	

generations, so they can explore the search space more effectively.

4) The modified crossover and mutation prohibit inbreeding, enhance the ability of preventing premature convergence of GA.

5) The "catastrophic mutation" which is based on Metropolis mechanism makes the GA escape from premature convergence, and at the same ensures the globe convergence.

Simulated examples validate that the proposed algorithm is an efficient approach to identify ARMAX process.

[REFERENCES]

[1] FANG Chong-zhi, XIAO De-yun. Systen Identification (in Chinese), [M]. Beijing: Tsinghua University Press, 1988.

[2] Holland J H. Adaption in Natural and Artifical Systems [M]. Cambridge: MIT Press, 1992.

[3] Kristinn Kristinsson. System identification and control using gas [J]. IEEE Trans on SMC, 1992, 22(5): 1033- 1046.

[4] Jones A H, Tatnall M L. Automated on-line frequency domain identification and genetic estimation of plant transfer functions [A]. Proc 10th IFAC Sympo on Identification and System Parameter Estimation [C]. Gopenhagen: 1994, 3: 817- 822.

[5] XU Hong-ze, CHU Dong-sheng, ZHANG Fu-en. A modified genetic algorithm and its application to system identification [J]. Journal of Harbin Institute of Technology, (in Chinese), 1997, 29(4): 72- 75.

[6] JING Bo, WANG Bin-wen. Parameter estimation of nonlinear system based on genetic algorithms [J]. Control Theory and Applications, (in Chinese), 2000, 17 (1): 150- 152.

[7] XU Li-na, LI Lin-lin. Application of genetic algorithm to identification of nonlinear system [J]. Journal of Harbin Institute of Technology, (in Chinese), 1999, 31(2): 39 - 42.

[8] Baolin W, Xinghuo Y. Fuzzy modeling and identification with genetic algorithm based learning [J]. Fuzzy Sets and Systems, 2000, 113: 351- 365.

[9] Gen Mitsuo, CHENG Run-wei. Genetic Algorithms and engineering Design [M]. John Wiley & Sons, Inc, 1997.

[10] ZHAO Ming-wang. A hybrid algorithm for global nonlinear least squrae solution based on genetic algorithm and steepest decent method [J]. System Engineering and Electronics, (in Chinese), 1997, 19(8): 59- 63.

[11] YU Shou-yi, PENG Ying, ZHENG Xiao-ming. A dynamic genetic algorithm based on numeric encoding [J]. Journal of Central South University of Technology, (in Chinese), 1998, 29(1): 85- 97.

[12] KONG Li-shan, XIE Yun. Non-numerical parallel computing [A]. Simulated Annealing Algorithms [C]. Beijing: Scientific Press, 1998. 7.

(Edited by ZHU Zhong-guo)