ELSEVIER Computational Statistics & Data Analysis 38 (2002) 315–330

COMPUTATIONAL STATISTICS & DATA ANALYSIS

www.elsevier.com/locate/csda

# Using genetic algorithms to parameters $(d, r)$ estimation for threshold autoregressive models

Berlin Wu[a,*], Chih-Li Chang[b]

[a] *Department of Mathematical Sciences, National Chengchi University, Taiwan*
[b] *Department of Mathematics, Hsiuping Institute of Technology, Taiwan*

Received 1 November 1999; received in revised form 1 February 2001; accepted 1 February 2001

**Abstract**

Threshold autoregressive model (TAR model) has certain characteristics due to which linear models fail to fit a nonlinear time series, while the problem of how to find an appropriate threshold value still attracts many researchers' attention. In this paper, we apply the genetic algorithms to estimate the threshold and lag parameters $r$ and $d$ for TAR models. The selection operator is formulated following Darwin's principle of survival of the fittest to guide the trek through a search space. The crossover and mutation operators have been inspired by the mechanisms of gene mutation and chromosome recombination. © 2002 Elsevier Science B.V. All rights reserved.

*Keywords:* Genetic algorithms; Threshold autoregressive models; Fitness function; Exchange rate

## 1. Introduction

Nonlinear time series analysis is a popular research topic that the scholars of time series and econometrics are intent on and the search for structure change for nonlinear time series has been attracting attention. For instance, Saikkonen and Luukkonen (1988) and Guegan and Pham (1992) used the Lagrange multiplier test (*LM test*) to test the threshold autoregressive models (TAR models); *autoregressive* conditional

[*] Corresponding author. Tel.: +2-938-7387; fax: +2-939-0064.
*E-mail address:* berlin@math.nccu.edu.tw (B. Wu).

PII: S 0167-9473(01)00030-5

heteroskedasticity models (ARCH models) and bilinear time series models. Tsay (1989) also used arranged autoregression to establish the parameters of TAR models. Chen and Lee (1995) proposed Bayesian inference of the threshold autoregressive model. Byers and Peel (1995) used TAR models to estimate and forecast industrial production for six countries and found that TAR models give better explanations and robust predictions than those of linear ARMA models. Recently, Wu and Chen (1999) applied fuzzy techniques in the detection of change period for nonlinear time series.

Practically, facts about structure changes often exist among many economic time series, and these phenomena of structure changes are generated by the variables, which are themselves based upon some change points. Exchange rate, for example, is dynamic floating for most countries, having a nonlinear tendency. If we employ the concept of transaction strategy from the filter rule: Buy foreign currency when the exchange rate keeps rising by a certain percent from its recent lowest point, and, on the contrary, sell foreign currency when it keeps falling by a certain percent from its recent highest point; then, this strategy will depend totally on its previous prices, regardless of the variations of economic factors (Boothe and Longworth, 1986). Therefore, we should construct a TAR model according to the previous prices of the exchange rate.

The purpose of this paper is to explore the TAR models, especially to find an alternative technique for the estimation of parameters $(d, r)$. This model separates the time series into several regions, each following a different AR model as a subsystem with respect to the change points considered (similar to the "recent lowest/highest points" of the filter rule). Then, we can use this nonlinear model to analyze the tendency of the underlying time series. However, regarding the TAR model itself, there still exist some problems that hinder its feasibility in applications. The main reasons are: (1) it is difficult to determine the number of change points in the time series, (2) it is difficult to find the threshold values during the establishing process of a TAR model. Besides, the number of parameters to be searched is too large for the traditional searching process.

Focusing on the stated problems of TAR models construction, this research is going to introduce the genetic algorithms (GA) by Holland (1975). Moreover, it has been frequently combined with fuzzy theories and the neural net and has become one of the most important research areas in the artificial intelligence field, for instance, see Goldberg and Deb (1991), Kennedy (1993), Bauer (1994), Oliver (1994), Yao and Sethares (1994), Loraschi et al. (1995), Herrera et al. (1997) and Wong and Yen (1999).

GA is a target-oriented parallel-searching technique, mainly applied to the optimization process searching for the universal or nearly universal extreme values. It processes a population of individuals, which represent search space solutions, employing three operations: selection, crossover and mutation. We hope that we can utilize the optimization searching techniques of genetic algorithms through certain algorithm rules and procedures, to find the optimized parameters automatically for the TAR model, to improve the weakness of conventional TAR models, and then complete the model establishment quickly and accurately.

To sum up, this research will set up a time series genetic algorithm using the establishing flow for TAR models together with the optimization searching techniques of genetic algorithms to find the optimized TAR models universally. We will use the "Macro" instruction of the statistical software, Minitab12.X, to compose a practical program for this purpose. An empirical study on Thai baht to US dollar exchange rate model construction as well as forecasting will be investigated.

## 2. TAR models and genetic algorithms

### 2.1. TAR models

TAR models have several obvious characteristics that can be used for general judgement and identification, such as periodic cycling, up-and-down oscillations and sudden rising or fallings. These characteristics are difficult to describe by the conventional linear ARIMA models, see Wu (1995). Take the example of using the TAR model for analyzing Lynx trappings and sunspot numbers by Tong and Lim (1980), where we find it easier to catch the periodic cycling by the TAR model rather than by the conventional analysis.

When dealing with the structure changes of the nonlinear time series, Tong (1983) presented the characteristics of the delay parameter and the threshold parameter (i.e. "change point"), and so he described them by the switch mechanism. Since then, the TAR model has been established completely.

Basically, the TAR model comprises several linear AR models and the switch mechanism, and the switch mechanism switches AR models according to the comparison of the delay output and the threshold value. For example, a TAR$(2; k_1, k_2)$ model is expressed in the following form:

$$
\begin{aligned}
\mathrm{Y}_t &= c_1 + \sum_{i=1}^{k_1} \phi_{1,i} \mathrm{Y}_{t-i} + a_{1,t} \quad \text{if } \mathrm{Y}_{t-d} \leqslant r \\
&= c_2 + \sum_{i=1}^{k_2} \phi_{2,i} \mathrm{Y}_{t-i} + a_{2,t} \quad \text{if } \mathrm{Y}_{t-d} > r,
\end{aligned} \tag{2.1}
$$

where, $k_1, k_2$ are the orders of two AR models; $\phi_{1,i}, \phi_{2,i}$ are autoregressive coefficients; $\varepsilon_{1,t}, \varepsilon_{2,t}$ are the white noise terms; $d$ is the delay parameter (where the controlled threshold occurs) and $r$ is the threshold value.

Tong (1983) proposed the algorithm for determining TAR models to determine the lag parameter $d$, the threshold value $r$, and the parameters of the two autoregressive models. The key point is to use Akaike's Information criterion (AIC)

$$
\mathrm{AIC} = n \log \hat{\sigma}^2 + 2(p+q), \tag{2.2}
$$

as the standard for model determination. The method is given as follows:

(1) First fix the delay parameter $d$ and the threshold value $r$ on $d_0$ and $r_0$, and let $L$ be the largest order of the model. Then divide the observations into two regional

systems, and assign proper autoregressive models, respectively. Calculate

$$\mathrm{AIC}(d_0, r_0) = \mathrm{AIC}(\hat{k}_1) + \mathrm{AIC}(\hat{k}_2), \tag{2.3}$$

where $\hat{k}_1, \hat{k}_2$ are the minimum AIC estimator (MAICE) of $k_1, k_2$, satisfying

$$\mathrm{AIC}(\hat{k}_i) = \min_{0 \leqslant k_i \leqslant \mathrm{L}} \{\mathrm{AIC}(k_i)\}, \quad i = 1, 2. \tag{2.4}$$

(2) Keep fixing $d$ on $d_0$. Considering all possible values $\{\tau_1, \tau_2, \ldots, \tau_s\}$ of the threshold value $r$, and then repeating Step 1, we can get

$$\mathrm{AIC}(d_0, \hat{r}) = \min_{r \in \{\tau_1, \tau_2, \ldots, \tau_S\}} \{\mathrm{AIC}(d_0, r)\}, \tag{2.5}$$

(3) Find the best-fitted delay parameter $d$. Consider all possible values $\{1, 2, \ldots, T\}$ of $d$, and repeat the above two steps. Due to the difference in $d$ value, the number of the effective data, $N$–$d$, will change. Therefore, we must first normalize AIC with respect to the number of data:

$$\mathrm{NAIC}(d) = \mathrm{AIC}(d, \hat{r})/(N - d). \tag{2.6}$$

(4) Finally we find $\hat{d}$, the minimum AIC estimator of $d$, satisfying

$$\mathrm{NAIC}(\hat{d}) = \min_{d \in \{1, 2, \ldots, T\}} \{\mathrm{NAIC}(d)\}. \tag{2.7}$$

After these three steps, we can get $\hat{k}_1, \hat{k}_2, \hat{r}, \hat{d}$ as the final MAICE for the orders $(k_1, k_2)$, threshold value $r$, and delay parameter $d$ of the two autoregressive models, respectively. Thus, we have accomplished the determination of the TAR model: $Y_t$ is a nonlinear model and its parameter structures change with respect to its previous values.

We find from the described algorithm for determining TAR model that: if the number of the autoregressive models is $M$ ($M \geqslant 2$), the largest order is $L$, the number of threshold values is $S$ ($S \geqslant (M-1)$), and the number of delay parameters is $T$, then the number of models to be computed must be

$$L^M \times \binom{S}{M-1} \times T = L^M \times \frac{S(S-1)\cdots(S-M+2)}{1 \times 2 \cdots (M-1)} \times T. \tag{2.8}$$

When $S$ and $M$, the number of threshold values and the number of the autoregressive models, increase, the number of models to be searched will be huge. This is a significant weakness within the TAR model itself.

The genetic algorithm is a method triggered by the basic structure of organism evolution, first proposed by John Holland in 1975. It combined Charles Darwin's principle of "natural selection" and "survival of the fittest" with the computer-constructed evolution mechanism to select better species from the original population. Then the information was exchanged randomly among them, expecting a superior offspring. Besides, in order to avoid missing some good species and becoming a local optimization, we must process several mutations.

The genetic algorithm not only avoids the trap of local optimization, but also reduces much computational time to find the optimum. Therefore, it is quite eligible for solving the optimization problems. This, of course, is also due to the aid of computer programs. So one of the great contributions by Holland (1975) is to turn the “evolution” concept into real computer programs.

In order to survive the complicated environment, organisms need the ability to fit themselves to the environment. Genes in the organism determine different fitting abilities when facing different environments. And different genes compose the chromosome. Thus, organisms can produce better gene combinations (chromosome) through many generations of the evolution process to equip themselves with the best fitting ability.

We can express the chromosome by the binary string in the program computation, that is, the encoding process for the entire parameter value range. For example, 11010 and 01101 are two chromosomes, representing two parameters respectively, and 0 and 1 are the genes in the chromosome. Then pick out a certain amount of chromosomes (i.e., the strings), at random to be the primary population, and decode the chromosomes (strings) of this population. Then compute the fitness of the chromosome (string) by the fitness function defined for optimization. This fitness is equivalent to the fitting ability of the chromosome facing complicated environment, only the one with better fitting ability having a better chance to survive and reproduce its offspring. In the same sense, the chromosome (string) with higher fitness has a higher chance of being selected for reproduction. After the reproduction, perform random crossover and mutation on the chromosomes according to the crossover rate and mutation rate to produce their new generation. Follow this procedure from generation to generation, repeat the “evolution” process (selection, reproduction, and mutation) over and over again until the most competitive and best fitted optimum is found. Then we will finish the job of the genetic algorithm.

## 2.2. Principle of the genetic algorithm

### 2.2.1. Encoding and decoding the parameters

The first step of the genetic algorithm is to encode the parameter values into an appropriate range with finite-digit digital strings (i.e. the chromosomes, usually binary strings). A widely used formula for decoding is

$$c = L + \frac{A}{2^B - 1} \times (U - L), \tag{2.9}$$

where $c$ is the parameter value of the chromosome; $U$ and $L$ are the upper and the lower bounds in the parameter; $A$ is the number in decimal form that is being represented in binary form of the chromosome; and $B$ is the digits of the chromosome.

For example, if $X$ is an integer-valued parameter, lying in [1,8], encoded by 3-digit strings, then the chromosome range will be $[000, 111]$, totally $2^3 = 8$ values. In which, chromosome 110 represents the value

$$c = 0 + \frac{2^2 + 2^1}{2^3 - 1} \times (8 - 1) = 7.$$

If $X$ is a real-valued parameter, lying in $[0, 10]$, encoded still by 3-digit strings, then the chromosome range will be [000, 111], totally $2^3 = 8$ values in which, chromosome 110 represents the value

$$c = 0 + \frac{2^2 + 2^1}{2^3 - 1} \times (10 - 0) = 8.57.$$

### 2.2.2. *Fitness function, selection, and reproduction*

The fitness function is the index to evaluate the fitting ability of the chromosome, and the selection and reproduction are similar to the asexual reproduction of organisms. The probability of selection and mutation is determined by the fitness function value, computed by the genetic algorithm, of each chromosome. Chromosomes with higher fitness will have a higher probability of being selected and reproduced, while chromosomes with lower fitness will be eliminated. So the next generation must have the chromosomes with higher fitness, that is, "evolution" has increased the overall quality of the next generation. As for the number of chromosome reproduction, it depends on the selecting method.

### 2.2.3. *Crossover*

Crossover resembles the sexual reproduction of organisms. It provides an information-exchanging function allowing different chromosomes (strings) in this population to exchange their genes (digits) for reproduction in a random crossover process. The new chromosomes will retain some characteristics (genes) of their parents but will still show differences. Their fitness might be higher or lower than that of their parents but chromosomes with low fitness will be eliminated after competition. Only chromosomes with high fitness survive.

Crossover enables the parents with some superior genes to reproduce the new generation with superior genes from both sides. And these more superior genes are able to reproduce more similar, even better, offspring to increase the fitness of the chromosomes in the whole population through this "evolution" process. That is why the genetic algorithm can find the optimum through this "evolution" mechanism quite efficiently.

The result of the crossover process depends on the crossover rate. The higher the crossover rate is, the more rapidly new chromosomes enter the population, and thus speed up the optimum searching. But if the crossover rate is too high, then the speed with which good chromosomes are taken out of the population might be faster than the speed of improving the chromosomes by crossover, and the crossover is thus in vain. On the contrary, if the crossover rate is too low, the optimum searching will slow down or even stop. Therefore, the crossover rate must be carefully selected. Usually, people take 0.5–0.9 as the appropriate range.

Although random factors exist in the crossover process, all the preserved chromosomes are of higher fitness because chromosomes of the crossover source have undergone the selection and reproduction process in advance. That is also another reason why the searching technique of the genetic algorithm is more effective than that of general random methods.

The following are three schemes of the crossover:

(i) One-point crossover

Randomly select one crossover point from one pair of the parental chromosomes (strings) and divide the chromosome into two parts, then switch the genes (digits) after the crossover point. Demonstrate by the following example (string length is 6, crossover point is 4):

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Father chromosome | 0 | 1 | 0 | 1 | : | 0 | 0 |
| Mother chromosome | 1 | 1 | 1 | 0 | : | 1 | 1 |
| ⇓ crossover | | | | | | | |
| Son chromosome | 0 | 1 | 0 | 1 | : | 1 | 1 |
| Daughter chromosome | 1 | 1 | 1 | 0 | : | 0 | 0 |

From this example, we know that although one-point crossover can produce different children chromosomes from their parental chromosomes, the last digits will definitely be switched. It will damage certain high-fitness chromosomes whose fixed digit of the schema is at the last position.

(ii) Two-point crossover

Randomly select two crossover points from one pair of the parental chromosomes (strings) and divide the chromosome into three parts, then switch the genes (digits) between the two crossover points. Demonstrate by the following example (string length is 7, crossover point is 4,7):

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Father chromosome | 1 | 0 | 0 | 1 | : | 1 | 0 | : | 0 |
| Mother chromosome | 0 | 0 | 1 | 0 | : | 0 | 0 | : | 0 |
| ⇓ crossover | | | | | | | | | |
| Son chromosome | 1 | 0 | 0 | 1 | : | 0 | 0 | : | 0 |
| Daughter chromosome | 0 | 0 | 1 | 0 | : | 1 | 0 | : | 0 |

Two-point crossover replaces the single point in the one-point crossover with two crossover points. It prevents the schemas from damage in the one-point crossover, but there are still many characteristics of many schemas that this two-point crossover cannot preserve effectively.

(iii) Uniform crossover

In addition to the frequent use of the one- and two-point crossovers, many relevant researchers of the genetic algorithm are eager to find methods with more crossover points to fit a variety of problems. The most successful method is the uniform crossover proposed by Syswerda (1991), that is: whether one digit is going to switch is determined by a random index; if the index is 1, then switch; if its 0, then do not. Demonstrated by the following example (string length is 7):

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Father chromosome | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| Mother chromosome | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| ⇓ crossover | | | | | | | |
| Son chromosome | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Daughter chromosome | 1 | 0 | 1 | 1 | 0 | 0 | 0 |

The consequence of uniform crossover is irrelevant to the encoding characteristic position of the chromosomes, unlike one- and two-point crossovers in which children chromosomes have higher possibilities of being incomplete if there are more fixed digits in the schema. But uniform crossover still has the possibility of damaging the superior genes. As a result, it depends on the situation of the problem to judge the crossover methods (Bauer, 1994).

### *2.2.4. Mutation*

If certain genes (digits) of all chromosomes (strings) in the population are identical, then their values will never change after any reproduction and crossover. This will limit the chance for some new chromosomes to enter this population, and possibly fall into the trap of local optimization. To avoid this situation, we have to perform a little mutation. We select, at random, a few genes of certain chromosomes to perform the mutation, i.e. turn 0 into 1 and 1 into 0, according to the mutation rate.

The possibility of the occurrence of mutation in the natural world is very low, so we shall also keep the mutation rate rather small in the genetic algorithm.

All the above are the fundamental principles of the genetic algorithm. According to the principles, first we have to encode the parameter value range of the target before the genetic algorithm is started, and then set up the controlled parameters and the terminating criterion of the genetic algorithm to execute the job of optimum searching successfully. These essential parameters are explained in Table 1.

## 3. Simulation study on the estimation of $(d, r)$ in TAR models with GA

The genetic algorithm is a tool for solving the optimization problems, being able to find a set of optimized parameters. Especially, when the problem we face has an enormous parameter space, which means lots of choices, the genetic algorithm demonstrates its efficiency to search and solve the problem. There are many differences between it and other conventional optimization solving methods.

(1) The genetic algorithm computes strings by encoded and decoded discrete points instead of the original parameter value set. That is why those discontinuity or

Table 1
Essential parameters in the genetic algorithm

| Parameters | Explanation |
|---|---|
| Encoding method and the chromosome length | Design the encoding method according to the parameter value range of the target. If the needed chromosome (string) length is $B$, then all parameter value combinations will be$2^B$ |
| Size of the population | The size of the population is the number of chromosomes participating in the evolution. Generally, it appears that the larger the population is, the closer to the optimum the result of the searching will be. But Syswerda (1991) pointed out that it would not sustain forever because the reproduction and crossover method and design of other controlled parameters would all influence the final outcome. Therefore, we can execute the genetic algorithm with a large population size to achieve evolution; we can also use a smaller size to speed up the convergence, but with a number of executions, to compare their results |
| Fitness function | Set up a fitness function of every target, being capable of evaluating the chromosome efficiency, to continue the evolution process |
| Selecting method | After the fitness function is settled, we must settle the selecting method. Then we can determine the number of each chromosome to be reproduced following the fitness function and the selecting method |
| Crossover rate and method | Crossover rate is the probability as to whether two chromosomes will crossover, ranging from 0 to 1, normally set between 0.6 to 1. At the same time, the crossover method must also be settled to continue the evolution process |
| Mutation rate | Mutation rate is the probability of gene mutation in every chromosome, usually set very small |
| Terminating criterion | The searching process of the genetic algorithm described in Section 2.3 is a cycling evolution process. So we have to set up a judging rule to decide whether or not to continue the process. This criterion is usually set to terminate the execution when the optimum (degree of convergence) or a certain generation (maximum terminating generation) is reached |

indifferentiability function problems that calculus cannot handle can be solved, thus escaping the additional limits in analysis. This is not only more flexible and efficient but also fits computer logic operations better, because of the adaptation of binary strings.

(2) There is no need of any extra information, such as prior information, but the fitness function is used completely for evaluation, because the primary population is created at random. From this, we can know whether the fitness function

is properly chosen, has become the key of the genetic algorithm and directly determines how well the consequence is.

(3) The genetic algorithm uses random processing to find the optimum more objectively. This random processing combines characteristics of both natural systems and artificial systems, which gives it high plasticity so as to improve the system.

(4) The searching process of the genetic algorithm depends on the selection, reproduction, crossover, and mutation, and all these steps are accompanied by random factors. Therefore, even under an identical parameter setting, every single execution of the genetic algorithm will be a stand-alone searching process, and thus it might end up with different answers.

Owing to its diverse characteristics described above, the genetic algorithm has diverse applications such as engineering, social sciences, and medicine. As Goldberg (1989) said, "Maturation of the genetic algorithm lies in its broaden application fields, not its theory". As a result, anyone can do the optimum-approaching calculation with the powerful searching ability of the genetic algorithm, if he or she can clearly transform the problem into a genetic algorithm form.

A practical GA procedure of executing the searching rule for $(d,r)$ of a TAR model is described as follows.

*Step* 1: *Encode the parameters* $d,r,k_1$, *and* $k_2$ *from an assigned range.*

*Step* 2: *Initialize a random population with N digits of chromosome.*

*Step* 3: *Decode the chromosome and construct the TAR models according to the* $d,r,k_1$ *and* $k_2$ *values.*

*Step* 4: *For each candidate TAR model, calculate its NAIC value and rank these values by the integral order.*

*Step* 5: *Delete undesirable members from population, evaluate the new chromosomes and insert them into the population.*

*Step* 6: *If the stopping criterion is satisfied, then stop and return the best chromosome; otherwise, go to step* 5.

### 3.1. Simulation study 3.1

We simulate 200 time series data from the following TAR model (3.1). The plot is shown in Fig. 1

$$\begin{aligned} Y_t &= 1 + 0.2Y_{t-1} + \varepsilon_{1,t} \quad \text{if } Y_{t-2} \leqslant 0, \\ &= -1 + 0.2Y_{t-1} + \varepsilon_{2,t} \quad \text{if } Y_{t-2} > 0. \end{aligned} \tag{3.1}$$

From model (3.1) we know that the values of parameters $d,r$ and $k_1,k_2$ are $2,0,1,1$, respectively. Given the possible range of parameters $d=1,2,\ r=0,1$ and $k_1=1,2,\ k_2=1,2$, there will be $2^4=16$ possible combinations. The following steps illustrate the whole searching process.

(1) Let $d,r$ and $k_1,k_2$ be codes as a 4-bit of strings. The bit strings consist of 0s and 1s, which represent the value of a number in binary form. For instance, here, the number 11 would look like 1011, i.e., $2^3+2^1+2^0=11$.

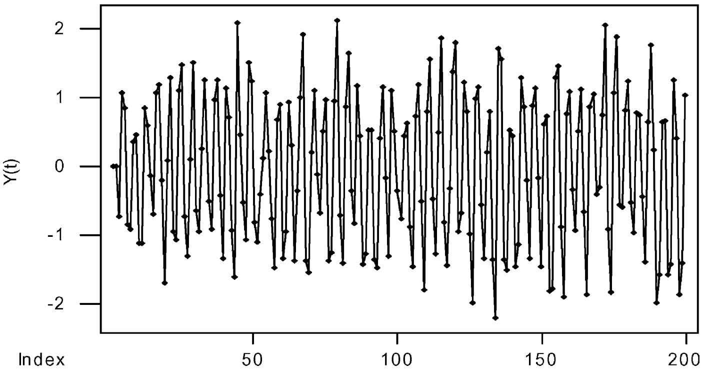

Fig. 1. Simulation data for model (3.1).

Table 2
The initial generation

| Family | Coding | $d$ | $r$ | $k_1$ | $k_2$0 | NAIC | Fitness | Probability of selection | Numbers of reproduction |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1101 | 2 | 1 | 1 | 2 | −0.081 | 3 | 0.3 | 1 |
| 2 | 1110 | 2 | 1 | 2 | 1 | −0.038 | 2 | 0.2 | 1 |
| 3 | 1001 | 2 | 0 | 1 | 2 | −0.996 | 4 | 0.4 | 2 |
| 4 | 0110 | 1 | 1 | 2 | 1 | 0.037 | 1 | 0.1 | 0 |

Table 3
The process for the first generation on reproduction

| Family | Coding | $d$ | $r$ | $k_1$ | $k_2$ | NAIC | Fitness | Probability of selection | Numbers of reproduction |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1101 | 2 | 1 | 1 | 2 | −0.081 | 2 | 0.2 | 1 |
| 2 | 1110 | 2 | 1 | 2 | 1 | −0.038 | 1 | 0.1 | 0 |
| 3 | 1001 | 2 | 0 | 1 | 2 | −0.996 | 3.5 | 0.35 | 1.5 |
| 4 | 1001 | 2 | 0 | 1 | 2 | −0.996 | 3.5 | 0.35 | 1.5 |

(2) Randomly generate a population with each chromosome containing 4 digits. For each chromosome, set up a TAR model according to $d, r$, and $k_1, k_2$.

(3) Calculate the values of adaptive function NAIC for each family. Then process the reproduction and crossover according to their fitness (rank).

Tables 2 and 3 illustrate the result of the process

(3) Let the crossover rate be 0.6 and the mutation rate be 0.001. Using the uniform crossover scheme, the random combinations of crossover are families 2 and 3. We illustrate the result in Table 4. Note that the crossover occurred at the 4th digit and that the mutation did not occur. The minimum NAIC occurred at the second family. We find that the second family of the second generation has a smaller NAIC value than the minimum NAIC of the first generation.

Table 4
The second generation

| Family | Coding | $d$ | $r$ | $k_1$ | $k_2$ | NAIC |
|---|---|---|---|---|---|---|
| 1 | 1101 | 2 | 1 | 1 | 2 | −0.081 |
| 2 | 1111 | 2 | 1 | 2 | 2 | −0.068 |
| 3 | 1000 | 2 | 0 | 1 | 1 | −1.044 |
| 4 | 1001 | 2 | 0 | 1 | 2 | −0.996 |

## 4. Empirical studies

Since the economics of APEC countries relies broadly on international trading, the exchange rate plays a crucial role in government policy. In response to the regional economics promotion, liberation and internationalization have become the index of a nation's goal. Therefore, the investigation of changing exchange rate with respect to other regional countries become indispensable.

In this empirical study, we will use the GA technique to search for the changing exchange rate of Thailand. We consider 300 daily data, from July 2, 1997 to September 15, 1998, which comes from FSM Data Bank, EPS, Computer Center, Ministry of Education. Fig. 2 plots the daily time series.

Let $k_1, k_2 \in \{1, 2, \ldots, L\}$. According to Tong (1983), the order of AR models had been suggested as $B = N^{\alpha}$, $\alpha < 1/2$, where $N$ is the data size. For model simplification and computational efficiency, in this study we choose $L = 9$, i.e. $k_1, k_{2 \in \{1,2,3,4\}}$.

Since the delay parameter $d \in \{1, 2, \ldots, T\}$, but $T \leqslant L$, we choose the range of $d$ as $1 \leqslant d \leqslant 4$. For the threshold value $r$, let $S = 5$. Since the median of the data is 40.2, we take the candidate values of $r$ to be $\{38.0, 38.5, 39.0, 39.5, 40, 40.5, 41.0, 41.5\}$.

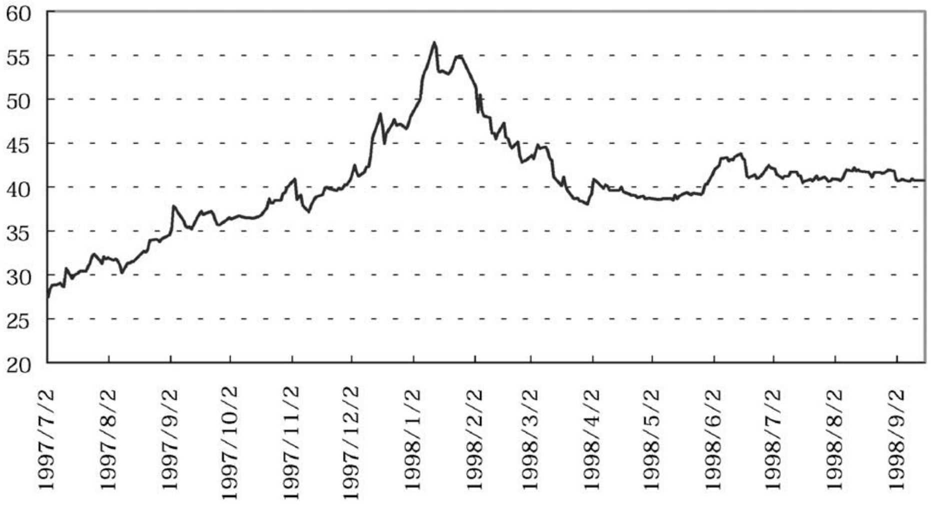

Fig. 2. Changing of exchange rate for Thailand/US$.

The boundary of $r$ approximately touches the $Q1$ (1/4) and $Q3$ (3/4) quintiles of the data.

If we use the above initial parameters settings and execute the procedure Tong (1983) proposed, we get the following result:

$$\begin{aligned} Y_t &= 1.0444Y_{t-1} - 0.00444Y_{t-2} + \varepsilon_{1,t} \quad \text{if } Y_{t-1} \leqslant 40.5 \\ &= 1.0256 + 1.2245Y_{t-1} - 0.2483Y_{t-2} + \varepsilon_{2,t} \quad \text{if } Y_{t-1} > 40.5. \end{aligned} \tag{4.1}$$

From Eq. (4.1) we can find that the threshold values of $d, r, k_1$ and $k_2$ are $1, 40.5, 2$ and 2, respectively, with the NAIC=$-189.78$+($-54.52$)=$-244.29$. Since the number of the autoregressive models $M = 2$, the largest order $L = 4$, the number of threshold values $S = 8$, and the number of delay parameters $T = 4$ from (2.8) the number of models to be computed must be

$$L^M \times \binom{S}{M-1} \times T = 4^2 \times \binom{8}{1} \times 4 = 512.$$

For comparison, if we use the ARIMA family to fit this data, we can find that the best-fitted ARIMA model is ARIMA(1, 1, 0):

$$Y_t = 1.1607Y_{t-1} + 0.1607Y_{t-2} + \varepsilon_t, \quad \text{AIC} = -191.44. \tag{4.2}$$

Next, to incorporate GAs into our scheme, we encode the rule base into a binary string. Let $d \in \{1,2,3,4\}$, $r \in \{38.0, 38.5, 39.0, 39.5, 40, 40.5, 41.0, 41.5\}$ $k_1, k_2 \in \{1, 2, 3, 4\}$; then, coding with length 9 is illustrated as

$$\overbrace{\bullet\ \bullet}^{d}\ \overbrace{\bullet\ \bullet\ \bullet}^{r}\ \overbrace{\bullet\ \bullet}^{k_1}\ \overbrace{\bullet\ \bullet}^{k_2}$$

These 9 bits represent the TAR characteristics. For example, the chromosome (string) with coding $\underline{01}\,\underline{011}\,\underline{10}\,\underline{11}$ stands for $d = 1$, $r = 39.0$, $k_1 = 2$, $k_2 = 3$. The total number of combination of parameters is $2^9 = 512$.

In executing the processing, we take the following parameters:

The population size $N = 20$.
The fitness function NAIC $(d, r, k_1, k_2) = [\text{AIC}(k_1) + \text{AIC}(k_2)]/(N - d)$.
The crossover probability $P_c = 0.8$ with uniform crossover.
The mutation probability $P_m = 0.01$.
The selecting rule: rank order selection.

Stopping rule: If the NAIC is less than $-245.98$ or the evolution reaches the 20th generation, then stop. The reason is that in order to compare the efficiency of our proposed method with that of Tong's (1983), we set the stopping value at $-245.98$, which is the NAIC for Tong's method.

We execute our GAs procedure three times with different initial populations. The processes are illustrated in Tables 5–7, respectively. All three processes reached the goal NAIC = $-245.98$. The numbers of iteration are 6, 4, and 10. Note that two of them converge very fast, while the third process met a bottleneck at the 9th generation. The average number of iterations is 6.67.

Note that there are 20 chromosome getting into the evolution. If we execute the process for 4 generations, then we need to search 80 TAR models by traditional

Table 5
GA in TAR model construction: first process

| Generation | Best chromosome | $d$ | $r$ | $k_1$ | $k_2$ | NAIC |
|---|---|---|---|---|---|---|
| 1 | 000101010 | 1 | 39 | 2 | 2 | −213.29 |
| 2 | 000100110 | 1 | 39 | 1 | 2 | −224.46 |
| 3 | 000111010 | 1 | 39.5 | 2 | 2 | −226.25 |
| 4 | 001001010 | 1 | 40 | 2 | 2 | −232.89 |
| 5 | 001010110 | 1 | 40.5 | 1 | 2 | −239.51 |
| 6 | 001011010 | 1 | 40.5 | 2 | 2 | −245.98 |

Table 6
GA in TAR model construction: second process

| Generation | Best chromosome | $d$ | $r$ | $k_1$ | $k_2$ | NAIC |
|---|---|---|---|---|---|---|
| 1 | 001110101 | 1 | 41 | 1 | 1 | −213.45 |
| 2 | 001111001 | 1 | 41 | 2 | 1 | −238.07 |
| 3 | 001011001 | 1 | 40.5 | 2 | 1 | −239.51 |
| 4 | 001011010 | 1 | 40.5 | 2 | 2 | −245.98 |

Table 7
GA in TAR model construction: third process

| Generation | Best chromosome | $d$ | $r$ | $k_1$ | $k_2$ | NAIC |
|---|---|---|---|---|---|---|
| 1 | 000001001 | 1 | 38 | 2 | 1 | −193.60 |
| 2 | 000001010 | 1 | 38 | 2 | 2 | −199.72 |
| 3 | 000011010 | 1 | 38.5 | 2 | 2 | −202.09 |
| 4 | 000011010 | 1 | 38.5 | 2 | 2 | −202.09 |
| 5 | 000011010 | 1 | 38.5 | 2 | 2 | −202.09 |
| 6 | 001011010 | 1 | 38.5 | 2 | 2 | −202.09 |
| 7 | 000011010 | 1 | 38.5 | 2 | 2 | −202.09 |
| 8 | 000110110 | 1 | 39.5 | 1 | 2 | −227.05 |
| 9 | 001001010 | 1 | 40 | 2 | 2 | −232.89 |
| 10 | 001011010 | 1 | 40.5 | 2 | 2 | −245.98 |

methods. If we execute the process 7 times (the average number of iterations), we need to search 140 TAR models. Therefore, it appears that our proposed GA is more efficient than that of Tong's.

## 5. Conclusion

The most difficult aspect in building a TAR model is the identification and finding of the procedure used to construct the model. Since the GA procedure can simultaneously select an appropriate subset of the $(d, r)$ parameters in the TAR models, it reduces the huge searching time compared to the traditional estimation methods. The advantage of GAs lies in its flexibility. As long as one decodes the possible answer in the form of binary strings, one can get the closest result by means of the evolutionary process of GAs.

Reviewing the results of our empirical study, we can see that the searching scheme for parameters $(d, r)$ can be found effectively. The searching schemes are capable of constructing the evolutionary rule to achieve satisfactory performance in model identification.

Lastly, we have three suggestions about the limitations and the research yet to be performed:

(1) The practical part of this research only explains the dynamic structural relationship between the exchange rate and its historical information without considering other impacts from seasonal factors and other economic variables.
(2) In the genetic algorithms, the determination of the controlled parameters (such as population size, crossover rate, etc.) apparently will affect its searching efficiency. However, problems of different kinds bring different criteria for the determination. Therefore, there is no universal standard for it at present.
(3) How to decide the mutation rate? Since GAs are essentially parallel searching techniques, the mutation rate should depend on the strong or long memory dependence data.
(4) Is the AIC criterion an optimal adaptive function for constructing TAR models?
(5) The appropriateness for the constructed time series model when compared with other models.

However, in order to get answers for the above questions, we believe that the research combined with fuzzy theories and the neural networks will constitute an encouraging approach in the field of nonlinear time series analysis.

## Acknowledgements

We are grateful to the referees for many very useful comments and suggestions, which have greatly improved our presentation.

## References

Bauer, R.J., 1994. Genetic Algorithms and Investment Strategies. Wiley, New York.

Boothe, P., Longworth, D., 1986. Foreign exchange market efficiency tests: implications of recent empirical findings. J. Internat. Money Finance 5, 135–152.

Byers, J.D., Peel, D.A., 1995. Forecasting industrial production using non-linear methods. J. Forecasting 14, 325–336.

Chen, C.W.S., Lee, J.C., 1995. Bayesian inference of threshold autoregressive model. J. Time Series Anal. 16, 483–492.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York.

Goldberg, D.E., Deb, K., 1991. A comparative analysis of selection schemes used in genetic algorithms. In: Rawlins, G.J.E. (Ed.), Foundations of Genetic Algorithms. Morgan Kaufmann, San Mateo, CA.

Guegan, D., Pham, T.D., 1992. Power of the score test against bilinear time series models. Statist. Sinica 2 (1), 157–169.

Herrera, F., Lozano, M., Verdegay, J., 1997. Fuzzy connectives based crossover operators to model genetic algorithms population diversity. Fuzzy Set and Systems 92, 21–30.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Kennedy, S.A., 1993. Five ways to a smarter genetic algorithms. AI Expert 8 (12), 35–38.

Loraschi, A., Tettamanzi, A., Tomassini, M., Verda, P., 1995. Distributed genetic algorithms with a application to portfolio selection problem. In: Pearson, N.C., Steele, N.C., Albrecht, R.F. (Eds.), Artificial Neural Networks and Genetic Algorithms. Springer, Berlin, pp. 384–387.

Oliver, J., 1994. Finding decision rules with genetic algorithms. AI Expert 9 (3), 33–39.

Saikkonen, P., Luukkonen, R., 1988. Lagrange multiplier test for testing nonlinearities in time series models. Scand. J. Statist. 15, 55–68.

Syswerda, G., 1991. Schedule optimization using genetic algorithms. In: Davis, L. (Ed.), Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, pp. 332–349.

Tong, H., 1983. Threshold Models in Non-linear Time Series Analysis. Springer, New York.

Tong, H., Lim, K.S., 1980. Threshold autoregression, limit cycles and cyclical data. J. Roy. Statist. Soc. 42 (3), 245–292.

Tsay, R.S., 1989. Testing and modeling threshold autoregressive process. J. Amer. Statist. Assoc. 84, 231–240.

Wong, L., Yen, J., 1999. Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and Kalman filter. Fuzzy Sets and Systems 101, 353–362.

Wu, B., 1995. Model-free forecasting for nonlinear time series: with application in exchange rates. Computat. Statist. Data Anal. 19, 433–459.

Wu, B., Chen, M., 1999. Use fuzzy statistical methods in change periods detection. Appl. Math. Comput. 99, 241–254.

Yao, L., Sethares, W.A., 1994. Nonlinear parameter estimation via the genetic agorithm. IEEE Trans. Signal Process. 42, 927–935.