Research Article

Forecasting Nonlinear Chaotic Time Series with Function Expression Method Based on an Improved Genetic-Simulated Annealing Algorithm

Jun Wang,[1] Bi-hua Zhou,[1] Shu-dao Zhou,[2] and Zheng Sheng[2]

[1]*National Key Laboratory on Electromagnetic Environmental Effects and Electro-Optical Engineering, PLA University of Science and Technology, Nanjing 210007, China*
[2]*College of Meteorology and Oceanography, PLA University of Science and Technology, Nanjing 211101, China*

Correspondence should be addressed to Jun Wang; wangjun19007@163.com

The paper proposes a novel function expression method to forecast chaotic time series, using an improved genetic-simulated annealing (IGSA) algorithm to establish the optimum function expression that describes the behavior of time series. In order to deal with the weakness associated with the genetic algorithm, the proposed algorithm incorporates the simulated annealing operation which has the strong local search ability into the genetic algorithm to enhance the performance of optimization; besides, the fitness function and genetic operators are also improved. Finally, the method is applied to the chaotic time series of Quadratic and Rossler maps for validation. The effect of noise in the chaotic time series is also studied numerically. The numerical results verify that the method can forecast chaotic time series with high precision and effectiveness, and the forecasting precision with certain noise is also satisfactory. It can be concluded that the IGSA algorithm is energy-efficient and superior.

1. Introduction

Chaos is a universal complex dynamical phenomenon that exists in various natural and social systems, such as communication, atmosphere, economics, and biology. Chaos phenomenon is generated by determinate equations but the appearance follows an apparently unpredictable nonperiodic stochastic pattern, and it is aperiodic, bounded, deterministic, and sensitive to the initial state. So the prediction of chaotic time series is very useful to evaluate characteristics of dynamical systems and is important to the research of chaos. The prediction of chaotic time series has been widely studied over the years. It is proved that short-term prediction of chaotic time series is possible by exploiting the deterministic dynamics in chaotic systems [1–6]. In general, the forecast of chaotic series implies two processes. The first step is to use the immediate past behavior of the time series to reconstruct state space [7]. Estimating the proper embedding dimensions and delaying time is the main work of state space reconstruction.

The self-correlation method [8], mutual information method [9], false nearest neighbors (FNN) algorithm [10], and C-C algorithm [11] have been introduced to reconstruct the state space. The second step is to build the forecasting model. Lots of techniques are proposed to build various models in many literatures, such as artificial neural networks [12–16] and polynomial fitting [17, 18]. Each of them has drawback and advantage; no method is superior to all other methods under every evaluating criterion. The paper proposes a novel and simple predictive model named the function expression method to forecast chaotic time series. In fact, there are many methods to establish the proper function expression. Zhang and Xiao [17] proposed a continued fractions method to give explicit expression. Zhou et al. [18] presented a multivariate local polynomial kernel estimator to approximate polynomial. Although those mathematical methods can predict accurately, they are too complex to predict the complex high dimensional chaotic systems, and the convergence speed is not high enough. The paper proposes an improved

FIGURE 1: The general procedure of chaotic time series forecasting.

genetic-simulated annealing algorithm to establish the best approximation to the real dynamic equation. The genetic algorithm has a strong capability of global optimization and has been used in many forecasting problems [19–23]. However, it is easily trapped into the local-best solution, and the quality of solutions is decreased a lot. We find that the simulated annealing algorithm [24, 25] has strong ability to jump out of the local-best solution and search for the best solution, but the local search ability is relatively poor. Therefore, incorporating the simulated annealing algorithm into the genetic algorithm is an ideal way that combines the global optimization ability of GA with the local search ability of SA; GA is developed to rapidly search for an optimum or near-optimum among the solution space, and then SA is utilized to seek a better one on the basis of that solution. In addition, the fitness function and the genetic operators are also improved to further improve the performance of optimization. The performance of the proposed method is verified by some simulations, and the results demonstrate the outstanding optimization capability and higher forecasting precision compared with other methods such as traditional genetic algorithm (GA) [1], continued fractions (CF) method [17], and neural network (NN) [16].

The remaining sections of this paper are organized as follows. In Section 2, the general forecasting principle of chaotic time series is presented. Section 3 elaborates the IGSA algorithm and the detailed forecasting procedure. The simulated numerical results are given in Section 4. The effect of noise in the chaotic time series is presented in Section 5. In Section 6, the discussions upon the proposed method are given. The paper ends with conclusions in Section 7.

2. General Forecasting Principle

The system state of a chaotic system and its delayed versions can be described in (1), where t is a scalar index for the time series, τ is the time delay, and m is the embedding dimension. Consider

$$\vec{X}(t) = [x(t), x(t-\tau), \ldots, x(t-(m-1)\tau)]. \quad (1)$$

We want to determine the dependence of the state value $x(t)$ on its previous state values. Takens Embedding Theorem [7] guarantees that the system's state information can be recovered from a sufficiently long observation of the output time series. According to the theorem, the system state follows the existence of a smooth map $T : R^m \rightarrow R$ satisfying

$$x(t) = T(x(t-\tau), x(t-2\cdot\tau), \ldots, x(t-m\cdot\tau)). \quad (2)$$

Thus, once the state space has been reconstructed from the time series, the chaotic time series can be forecasted by

establishing the functional relation $T(\cdot)$ with the proposed IGSA algorithm.

Measurements from a chaotic system are not restricted to a unique variable, but situations in which several variables are observed from the same system are common. In this case, we need to deal with multivariate time series $\{[x(t_i), y(t_i), \ldots, z(t_i)]\}$, $i = 1, \ldots, N$. The model of connection between the different variables can be written in (3), where $H(\cdot)$ is the model to be determined. Consider

$$x(t) = H(y(t-\tau), \ldots, y(t-m\cdot\tau), \ldots, z(t-\tau), \ldots, \\ z(t-m\cdot\tau)). \quad (3)$$

The general principle of using IGSA method to establish the optimum function expression to forecast chaotic series time is shown in Figure 1.

3. Function Expression Established by IGSA

In this section, the IGSA algorithm is proposed to establish the functional relation. The genetic algorithm is an evolutionary method that mimics the process of natural selection. GA generates solutions to optimization problems using techniques inspired by natural evolution, such as selection, crossover, and mutation. During each generation, a proportion of the existing population is selected to breed a new generation through a fitness-based process. Crossover is a process of taking more than one parent solution and producing a child solution from them. It is analogous to reproduction and biological crossover. Mutation is used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. The purpose of mutation is preserving and introducing diversity. Hence GA can come to better solution by using mutation.

The processes of establishing the best equation by GA can be described as follows. The GA algorithm selects the initial population of potential equations from the initial population of individuals that best fit the real data. The strongest strings choose a mate for reproduction whereas the weaker strings become extinct. The newly generated equations are subject to mutations that change fractions of information. The evolutionary steps are repeated until the optimum equation is established. Although GA can search the optimization solution quickly, the quality of the solution needs to be improved because it is liable to be trapped in local optima. The IGSA algorithm gets over the shortcomings of GA and incorporates the simulated annealing algorithm. The annealing process is usually simulated using a Monte Carlo procedure. In this procedure, the thermal motion of atoms in contact with a heat bath at a given temperature is simulated. The procedure is simply stated here [24]: Given

TABLE 1: Encoding rule of equation strings.

	Second coordinate	First coordinate	Pair
Operator term	1	1	$+$
	1	2	$-$
	1	3	$\times$
	1	4	$\div$
Constant term	2	$k \in [-\lambda, \lambda]$	k
Variable term	3	1	$x(t - 1 \cdot \tau)$
		$\vdots$	$\vdots$
		i	$x(t - i \cdot \tau)$
	4	1	$y(t - 1 \cdot \tau)$
		$\vdots$	$\vdots$
		i	$y(t - i \cdot \tau)$
	$\vdots$	$\vdots$	$\vdots$

a configuration of the elements of the system, randomly displace the elements on a time by a small amount and calculate the resulting change in the energy, ΔE. If $\Delta E < 0$ then accept the displacement and use the resulting configuration as the starting point for the next iteration. If $\Delta E \geq 0$ then the displacement is accepted with probability shown in (4), where T is the current temperature and k_b is Boltzmann's constant. Repetition of this step continues until equilibrium is achieved. For more detailed information on the simulated annealing, please refer to [24, 25]. Consider the following:

$$P(\Delta E) = \exp\left(-\frac{\Delta E}{k_b T}\right). \quad (4)$$

In addition, the fitness function and genetic operators are also improved to further improve the efficiency of the exploration. The whole graphical procedure of the IGSA algorithm is illustrated in Figure 2.

The detailed procedures of using IGSA algorithm to establish the best functional relation are described as follows.

Step 1 (set the encoding rules). In order to encode the symbolic form of the equation strings into a numerical structure, the encoding utilizes coordinate pairs, and the detailed rules are listed in Table 1. The second coordinate

indicates an argument or operator in the equation. If the second coordinate is 1, then the pair represents operator. For example, the pair $(p, 1)$ represents an operator in the equation string, and the values of p range from 1 to 4 to represent the four basic arithmetic operators, addition, subtraction, multiplication, and division, respectively. If the second coordinate is 2, then the pair represents constant term. For example, the pair $(k, 2)$ represents the real argument k of the equation. If the second coordinate is greater than 2, then the pair represents the variable of the equation. For example, the pair $(i, 3)$ represents the element $x(t - i \cdot \tau)$ in the time series, and $(i, 4)$ represents the element $y(t - i \cdot \tau)$ in the time series.

Step 2 (initialize). The precursor process is the generation of initial population of individuals as a basis for future generations. As the encoding rule of the equation is special, in order to make the decoded individual form the equation, there are several rules when building the initial individuals.

The first two elements of the individual must be constant or variable terms, and the last one must be an operator.

Given one location in the individual the number of nonoperators (constant or variable terms) on the left must be greater than the number of operators.

The total number of nonoperators in the individual must be the total number of operators plus one.

Step 3 (calculate the fitness value). In order to make the difference of the individual fitness obvious, the simulated annealing stretching operation is introduced to calculate the fitness value of individual. The improved fitness value of the i-individual is computed as

$$\text{Newfitness}(i) = \frac{\exp\left(\text{Fitness}(i)/T\right)}{\sum_{i=1}^{N} \exp\left(\text{Fitness}(i)/T\right)} \cdot N, \quad (5)$$

where N is the total length of the training set, $T = G(T_0)$, T_0 is the initial temperature, and G is the annealing mode. With the iteration going on, the temperature decreases and the superiority of the better individual is reinforced, so that the probability of selecting the better individuals is higher and the probability of selecting the worse individuals is lower. For the short-term predictability of unique and multivariate time series, Fitness(i) is computed in (6) and (7), respectively. Consider

$$\text{Fitness}(i) = 1 - \frac{\sum_{t=m\cdot\tau+1}^{N} \left[x(t) - T_i(x(t-\tau), x(t-2\cdot\tau), \ldots, x(t-(m-1)\cdot\tau))\right]^2}{\sum_{t=m\cdot\tau+1}^{N} \left[x(t) - (1/(N-m\cdot\tau)) \cdot \sum_{t=m\cdot\tau+1}^{N} x(t)\right]^2}, \quad (6)$$

$$\text{Fitness}(i) = 1 - \frac{\sum_{t=m\cdot\tau+1}^{N} \left[x(t) - H_i(y(t-\tau), \ldots, y(t-(m-1)\cdot\tau), \ldots, z(t-\tau), \ldots, z(t-(m-1)\cdot\tau))\right]^2}{\sum_{t=m\cdot\tau+1}^{N} \left[x(t) - (1/(N-m\cdot\tau)) \cdot \sum_{t=m\cdot\tau+1}^{N} x(t)\right]^2}. \quad (7)$$

Step 4 (make selection, crossover, and mutation genetic operation). The three genetic operations are described in detail as follows.

The Selection Operation. The selection operator chooses individuals that are used for crossover and mutation based on the fitness of the individuals. Proportional selection operator is
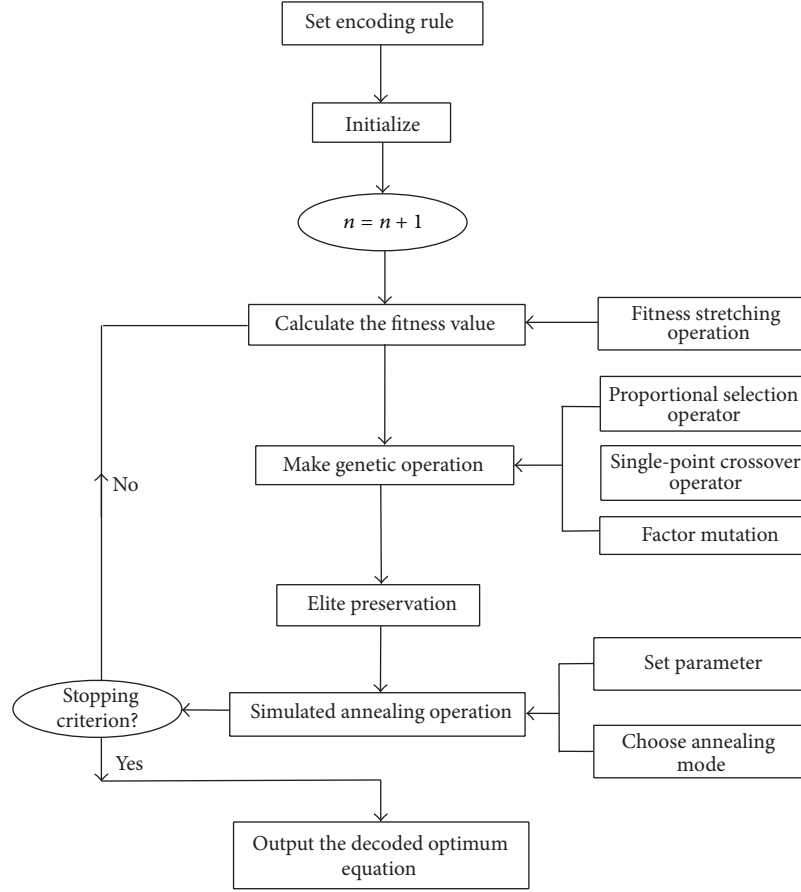
FIGURE 2: The whole graphical procedure of the proposed IGSA algorithm.

used to select the individuals, and the probability of selecting i-individual is proportional to its fitness value. The probability is computed as

$$\text{prob}\,(i) = \frac{\text{Newfitness}\,(i)}{\sum_{k=1}^{N} \text{Newfitness}\,(k)}. \tag{8}$$

The Crossover Operation. Once the mates are selected according to their strength, the crossover operation between the two parent individuals is carried out to generate two new offspring. The crossover procedure starts determining randomly one of the arguments, constant terms, or variables terms of the time series in the first individual. If the next element to the right of this randomly determined argument is an operator, then only this argument is considered for interchanging. If the next pair to the randomly selected argument represents another argument, the part of the string used for the crossover is that which is limited between the randomly selected argument and that element of the string where the numbers of nonoperators, N_S, and the number of operators, N_O, between the randomly selected argument and the element verify the relation $N_O + 1 = N_S$. Notice that the object of this procedure is to interchange self-consistent parts between the equation strings in order to avoid inconsistent mathematical expressions in the offspring.

The same operation is carried out for the second equation string. Take the crossover procedure of the following equation strings F_1, F_2 of two parents for example; strings (2, 3) and (3.6, 2) are the crossover points; the possible pairs of offspring are $F_{1\text{offspring}}$, $F_{2\text{offspring}}$, where the bold pairs represent the parts that have been crossed between the parent equations:

$$F_1 = \{(1.5, 2)\,,(1, 3)\,,(2, 3)\,,(1, 1)\,,(3, 1)\}\,,$$

$$F_2 = \{(3.6, 2)\,,(1, 3)\,,(1, 1)\,,(2, 3)\,,(4, 1)\}\,,$$

$$F_{1\text{offspring}} = \{(1.5, 2)\,,(1, 3)\,,(\mathbf{3.6, 2})\,,(\mathbf{1, 3})\,,(\mathbf{1, 1})\,,(1, 1)\,, \tag{9}$$

$$(3, 1)\}\,,$$

$$F_{2\text{offspring}} = \{(\mathbf{2, 3})\,,(2, 3)\,,(4, 1)\}\,.$$

The Mutation Operation. A mutation operation is taken to yield solutions with new information. In order to preserve the information of the top ranked individual, the mutation is not applied to the best individual with the minimal fitness value of the current iteration. Each element of a determined string is changed by a mutation process with some probability. An element is randomly selected and, depending on whether it is a number, a variable, or an operator, if the element is an operator, then the element is self-mutated; if the element is a number or variable, then the element is self-mutated or
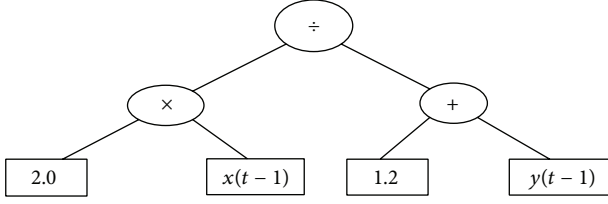
FIGURE 3: The binary tree structure of the equation strings.

mutually mutated. Take $F_{2\text{offspring}}$ for example; the possible mutated string is $F_{2\text{ off \_mut}}$, where the bold pairs represent the mutations. Consider

$$F_{2\text{ off \_mut}} = \{(\mathbf{1}, \mathbf{2}), (\mathbf{1}, \mathbf{3}), (\mathbf{2}, \mathbf{1})\}. \tag{10}$$

Step 5 (simulated annealing operation). After the mutation operation, preserve the best individual of the current nth iteration, and compare the fitness Fitness_new of the best individual of the nth iteration with the fitness Fitness_old of the best individual of the $(n-1)$th iteration. If Fitness_new is better than Fitness_old, then replace Fitness_old with Fitness_new and accept the new best individual; the worst individual of current iteration is also replaced with the best individual; otherwise, displacement is accepted with probability shown in (4), and the worst individual of the current iteration is also replaced with the best individual of the current iteration. Set the proper parameters related to SA, and then the local search performance can be well performed to avoid falling into a local optimum.

Step 6 (determine the best individual of the current iteration and check whether the stopping criterion is met). If the criterion is met then decode the optimum individual and obtain the final equation expression; otherwise, continue the iteration processes until the criterion is met or the maximum iteration number is achieved. The decoding rule is defined as follows to assemble the equation strings.

The operator term is placed after its two operands; taking the following coded individual $\{(2.0, 2), (1.0, 3), (3, 1), (1.2, 2), (1, 4), (1, 1), (4, 1)\}$ for example, the decoded equation strings are assembled in the binary tree structure shown in Figure 3. The corresponding equation expression is decoded as $(2 \cdot x(t-1))/(1.2 + y(t-1))$.

4. Numerical Simulations of Forecasting by the Function Expression

When the optimum function expression is established, the forecasting model is built. In order to verify the feasibility and effectiveness of the prediction of chaotic time series by the function expression, the nonlinear chaotic time series obtained from Quadratic map and Rossler map are adopted as testing objects.

First take the simple unique time series Quadratic map for example. The equation of Quadratic map is

$$x_{n+1} = c - x_n^2, \tag{11}$$

where c is a constant; here $c = 1.5$, and the initial value is $x_1 = 0.8$; 400-step values are generated from (11), and the previous 100-step values are abandoned to diminish the influence of the starting value; the 101–300 values are used to train and establish the equation; the remaining 100-step values are taken to test the forecasting performance. The delaying time τ and embedding dimension m are determined as 1, 2 by use of the method of minimum mutual information [9] and false nearest neighbors algorithm [10], respectively. So the mapping model is

$$x(n) = T(x(n-1), x(n-2)). \tag{12}$$

The IGSA algorithm is performed to establish the function equation, and the parameters of the algorithm are set as follows. The number of generations $N_{\max} = 5000$. The number of initial individuals $N_{\text{pop}} = 100$. The length of the individual is 19. The probability of mutation $P_{\text{mut}} = 0.1$. The initial temperature $T_0 = 1000$ (the initial temperature should be set high enough to make the initial accepting probability high). The annealing rate function is the chosen classical annealing function [25] shown in

$$T(n) = \frac{T_0}{\ln(1+n)}. \tag{13}$$

After iterations the final optimum individual is obtained: $\{(2.000000, 3), (2.000000, 3), (1.000000, 3), (2.000000, 3), (2.000000, 3), (2.000000, 3), (1.000000, 1), (2.000000, 1), (2.000000, 3), (2.000000, 3), (2.000000, 3), (1.000000, 3), (3.000000, 1), (2.000000, 1), (4.000000, 1), (3.000000, 1), (4.000000, 1), (2.000000, 1), (3.000000, 1)\}$, and the simplified form of the corresponding decoded function expression is

$$x(n) = x^2(n-2) + x(n-1) - x^2(n-1). \tag{14}$$

It can be found that $x^2(n-2) + x(n-1) = 1.500000$ (calculated accurately to sixth place of decimal) from the Quadratic dynamic equation, so that (14) can be written as follows:

$$x(n) = 1.500000 - x^2(n-1). \tag{15}$$

It is obvious that (15) is very close to the real map given by (11). Using the independent 100-step values to test the forecasting performance, the real time series generated by (11) and the forecasting time series obtained from (15) are both plotted in Figure 4. Figure 5 indicates the convergence process of the method of IGSA during the iterations. It can be seen from Figure 4 that the forecasting values are almost the same as the real values. The mean square error (MSE) calculated by (16) is $3.089628E - 11$, and the value is very close to zero which verifies the high forecast precision of the method. Consider

$$\text{MSE} = \frac{1}{100} \sum_{n=301}^{400} \left[x(n) - \left(1.500000 - x(n-1)^2\right)\right]^2, \tag{16}$$

Fitness (best)

$$= 1 - \frac{\sum_{n=301}^{400} \left[x(n) - \left(1.5000000 - x(n-1)^2\right)\right]^2}{\sum_{n=301}^{400} \left[x(n) - (1/100) \cdot \sum_{n=301}^{400} x(n)\right]^2}. \tag{17}$$
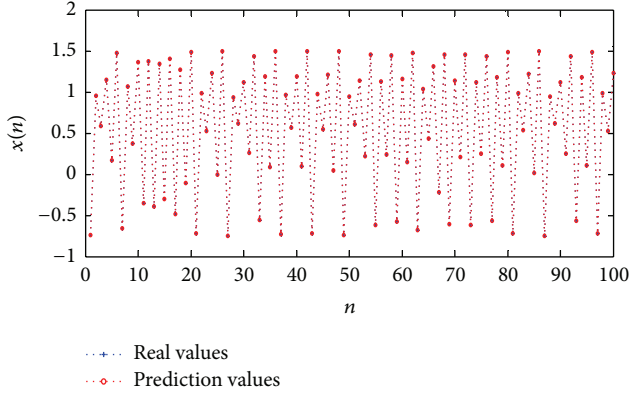
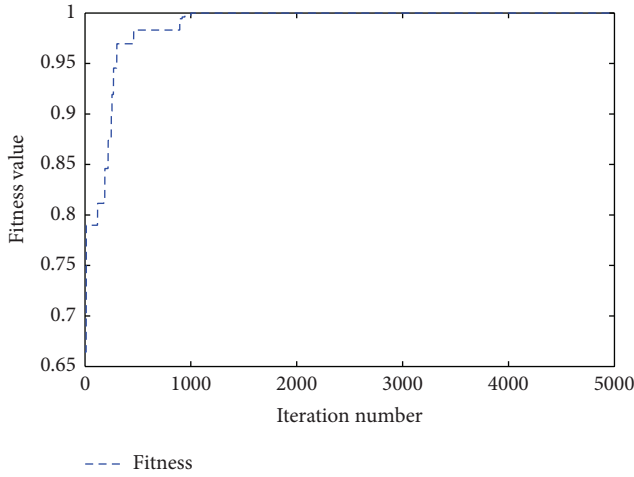FIGURE 4: Real values of Quadratic time series and short-term prediction values.

FIGURE 5: Convergence process of IGSA during the iterations.

Figure 5 indicates that the fitness value converges to the best value with a high speed and the final best fitness value is calculated as $\text{Fitness}_{\text{test}} = 1.000000$ (accurately to sixth place of decimal) by (17).

The complex multivariate chaotic Rossler system is also used to test the performance of the proposed method. Rossler map is described as

$$\frac{dx}{dt} = -(y + z),$$

$$\frac{dy}{dt} = x + 0.2y, \tag{18}$$

$$\frac{dz}{dt} = 0.2 + z \cdot (x - 5).$$

Time series of all variables (x, y, z) are obtained from solving (18) via 4th-order Runge-Kutta method. The integral step of 4th-order Runge-Kutta method is 0.01, the previous 10000-step values are abandoned, and the following 3000-step values of all variables are adopted to be the sample data, and the 1–2800 values are used to train and establish the equation;
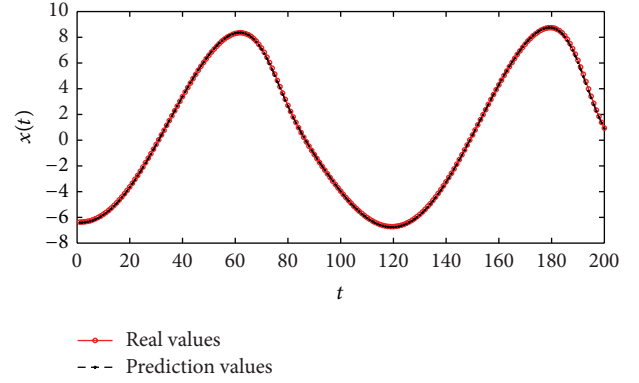

FIGURE 6: Real values of Rossler time series and short-term prediction values.

the remaining 200-step values are taken to test and validate the forecasting performance. The initial values of the variables are $x_1 = -1$, $y_1 = 0$, and $z_1 = 1$.

Take the x-variable for example, and the processes of forecasting y-variable and z-variable are so similar that they are not given here. The delaying time τ and embedding dimension m are determined as 1, 4 by use of the minimum mutual information method and false nearest neighbors algorithm, respectively. So the mapping model of x-variable is

$$x(t) = H(y(t - u), \dots, y(t - 4 \cdot u), \dots, z(t - u), \dots,$$
$$z(t - 4 \cdot u)). \tag{19}$$

The parameters of the IGSA method are set as follows. The number of generations $N_{\max} = 5000$. The number of initial individuals $N_{\text{pop}} = 120$. The length of the individual is 19. The probability of mutation $P_{\text{mut}} = 0.1$. The initial temperature $T_0 = 1000$. The annealing rate function is (13).

The final optimum individual is got after iterations, shown as follows: {(4.914803, 2), (−4.694566, 2), (2.000000, 1), (2.000000, 4), (4.000000, 4), (2.000000, 1), (3.000000, 1), (2.000000, 4), (4.000000, 4), (4.000000, 4), (1.000000, 1), (2.000000, 1), (4.000000, 4), (1.000000, 5), (1.000000, 1), (2.000000, 1), (5.713432, 2), (4.000000, 1), (1.000000, 1)}, and the simplified form of the corresponding decoded equation is

$$x(t) = 9.784395\, y(t - 2) - 10.134447\, y(t - 4)$$
$$- 0.175026\, z(t - 1). \tag{20}$$

It can be seen from (20) that the values of x-variable can be extrapolated by the previous values of y-variable and z-variable. The forecasting performance is verified by taking the remaining 200-step values. The prediction values and real values of Rossler time series are shown in Figure 6, and it can be seen from Figure 6 that the forecasting values are very close to the real values; the mean square error (MSE) is $2.755001E - 03$. Figure 7 indicates that the fitness value converges to the best value with a high speed and the final best fitness value is calculated as $\text{Fitness}_{\text{test}} = 0.999903$ (accurately to sixth place of decimal).

TABLE 2: Fitness values for noisy time series with different SNR.

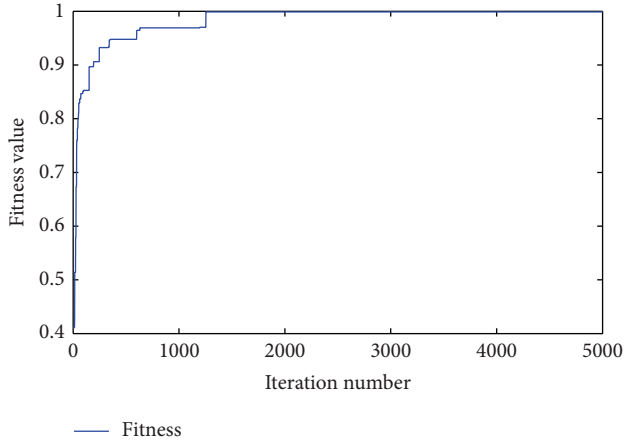| Chaotic system | Fitness value | | | | | | | |
| | SNR value (dB) | | | | | | | |
	50	40	35	30	25	20	15	10
Quadratic map	0.9908	0.9012	0.8452	0.8023	0.7473	0.6115	0.4396	0.2618
Rossler map	0.9862	0.8443	0.7629	0.7119	0.6592	0.5361	0.3587	0.2130

FIGURE 7: Convergence process of IGSA during the iterations.

5. Effect of Noise in the Chaotic Time Series

As the actual chaotic systems always contain noise, in order to test the influence of noise on the forecasting precision, the noise-free time series of Quadratic and Rossler maps were corrupted using Gaussian white noise with the different signal-to-noise ratios (SNR) values (see Table 2). The fitness values obtained by taking the same procedure described earlier were also listed in the table.

It can be observed from Table 2 that the fitness values are all decreased, and none of the forecasting precisions matched the precision when noise-free data were used. However, it can be found that the precisions are generally satisfactory when the values of SNR are larger than 25 dB, and when the SNR decreased, the fitness value was decreased a lot. It can be concluded that the proposed method possesses a certain capability of noise immunity and can effectively perform in the noisy time series with large SNR values (in this case, the SNR value should be larger than 25 dB). However, the limit is also obvious that the precisions are not acceptable and the method is not effective in a highly corrupted time series.

6. Discussion

In order to demonstrate that the proposed method is robust to parameter variations and to evaluate the effect of parameter variations, different parameters are also set to test the performances of the method, such as the number of generations and initial individuals and the probability of mutation and the initial temperature. The results are varied slightly. However,

the following rules of setting the parameters may prove helpful to improve the efficiency and accuracy of the method.

The initial temperature should be set high enough to make the initial accepting probability high. The temperatures from 50 to 3000 (50, 100, 200, 300, 400, . . . , 3000) are all used to perform the method, and the results indicate that the performance is improved when the temperature increased, but the performance is kept constant until the temperature reaches 1000, so in the paper the temperature is set to be 1000.

The number of initial individuals should be moderate; if the number is too large, then the time consumption is huge; if the number is too small, then diversity of the initial solutions is limited which reduces the quality of solutions.

The number of the generations should also be moderate; if the number is too large, then the time consumption is huge; if the number is too small, then optimal solution may not be searched out during the iterations.

The probability of mutation is typically kept small, and then the diversity of the solutions is improved and the speed of convergence is also kept high. If the probability is large, then the number of iterations increases a lot and the performance is reduced. The probability values from 0.01 to 0.3 are all used to perform the method, and the results are varied slightly with the probability from 0.01 to 0.2. Meanwhile, the performance is reduced when the probability is larger than 0.2.

In order to assess the advantage and improvement of the IGSA method, three representative methods (the traditional genetic algorithm (GA) [1], continued fractions (CF) method [17], and neural network (NN) method [16]) were chosen to compare with the IGSA method. The GA method was applied by Alvarez et al. [1] to forecast chaotic time series; though the precision was high, the accuracy can still be improved a lot as the traditional GA is easily trapped into the local-best solution and the genetic operator can be also improved. The CF method [17] was used by Zhang and Xiao to forecast chaotic time series; it is a pure mathematical method of polynomial approximants, and the main drawback is that it is too complex to predict the complex high dimensional chaotic systems. de Oliveira et al. [16] used the NN method to forecast chaotic systems; the results were almost satisfactory; however, the performance of NN was highly related to the topology structure and initial parameters, and it was easy to fall into local optimal solution.

The best parameters of NN used to forecast Quadratic and Rossler maps are listed in Table 3 referring to [16], and the ideal architectures of NN are 2-4-2-1 and 4-8-4-1 for forecasting Quadratic and Rossler maps, respectively. The forecasting precisions of Quadratic and Rossler maps evaluated by the different methods are listed in Tables 4 and 5,

TABLE 3: Parameters of NN for forecasting Quadratic and Rossler maps, respectively.

| Chaotic system | Parameters of NN | | | | |
	Input units number	Hidden layer number	First hidden layer neurons number	Second hidden layer neurons number	Output units number
Quadratic map	2	2	4	2	1
Rossler map	4	2	8	4	1

TABLE 4: Forecasting precisions of Quadratic map calculated by IGSA, GA, CF, and NN methods.

Method	IGSA	GA	NN	CF
Fitness value	1.000000	0.999996	0.999796	0.998845
MSE value	$3.089628E - 11$	$2.695395E - 06$	$3.190952E - 05$	$5.522902E - 04$

TABLE 5: Forecasting precisions of Rossler map calculated by IGSA, GA, and NN methods.

Method	IGSA	GA	NN
Fitness value	0.999903	0.998318	0.981785
MSE value	$2.755001E - 03$	$4.755993E - 02$	0.514942

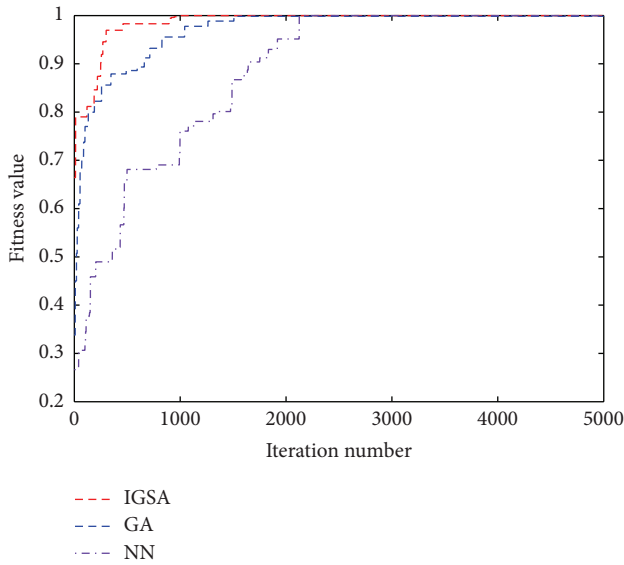FIGURE 8: Convergence process of IGSA, GA, and NN methods for Quadratic map during the iterations.

FIGURE 9: Convergence process of IGSA, GA, and NN methods for Rossler map during the iterations.

respectively. The history of convergences for IGSA, NN, and GA is illustrated in Figures 8 and 9.

By the way, Rossler map is too complex to predict by the method of CF, and then the corresponding results are not given here. It can be seen from Tables 3 and 4 and Figures 8 and 9 that the values of fitness and MSE are all satisfactory. However, it is obvious that the results obtained by IGSA are still improved; the fitness value is nearest to 1 and the MSE value is the minimum; besides, the convergence speed is also improved. In fact, it is not easy to further increase the precision when the previous precision is very high; the better values of the fitness value and MSE verifies the superior performance of the proposed algorithm.
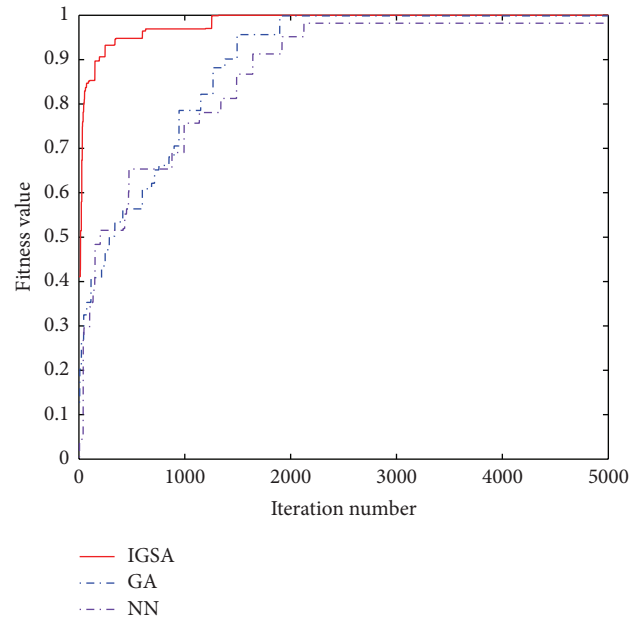
Though the proposed method can forecast chaotic time series with high precision and efficiency, the limits are also obvious: The method cannot be applied to the signals which are produced by the complex high dimension systems; it is also unsuitable to apply the method to the signals which are corrupted with low SNR values. Besides, the sample size should be large, and the accuracy and effectiveness of forecasting the chaotic time series with high dimension are decreased; the forecasting precision is decreased when the data are noise-corrupted, and the forecasting results may be ineffective when the SNR values of the noise are not high enough.

7. Conclusion

In this paper, we propose a simple but energy-efficient method to predict chaotic time series. The core thought of prediction by the proposed method is adopting an

improved genetic-simulated annealing algorithm to construct the optimum function expression to approximate the original dynamic equation of chaotic time series. Then the prediction can be carried out by the constructed expression. The IGSA algorithm incorporates the simulated annealing operation into genetic algorithm to enhance the optimization performance. The simulated annealing stretching operation is introduced to calculate the fitness value and genetic operators are also improved. Finally, two kinds of chaotic time series are used as the testing objects to verify the accuracy and effectiveness of the proposed method, and the corresponding results indicate that performance of forecasting is efficient and satisfactory. The forecast precisions of the simple unique time series and complex multivariate time series are both high enough. In order to evaluate the forecast performance when the system is contaminated with noise, the white noises with different SNR values are added; the results indicate that the method possesses a certain capability of noise immunity and can effectively perform in the noisy time series with large SNR values. We also use other common methods such as traditional genetic algorithm, continued fractions, and neural network to do the same numerical simulations; the compared results demonstrate the obvious superiority and improvement of the proposed method. In general, the proposed IGSA algorithm is a feasible, energy-efficient, and promising method for chaotic time series prediction.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

References

[1] A. Alvarez, A. Orfila, and J. Tintore, "DARWIN: an evolutionary program for nonlinear modeling of chaotic time series," *Computer Physics Communications*, vol. 136, no. 3, pp. 334–349, 2001.

[2] M. Ardalani-Farsa and S. Zolfaghari, "Residual analysis and combination of embedding theorem and artificial intelligence in chaotic time series forecasting," *Applied Artificial Intelligence*, vol. 25, no. 1, pp. 45–73, 2011.

[3] Y. Zhang, "New prediction of chaotic time series based on local Lyapunov exponent," *Chinese Physics B*, vol. 22, no. 5, Article ID 050502, 2013.

[4] M. Ahmadi and H. Mojallali, "Chaotic invasive weed optimization algorithm with application to parameter estimation of chaotic systems," *Chaos, Solitons & Fractals*, vol. 45, no. 9-10, pp. 1108–1120, 2012.

[5] M. Feki, "An adaptive chaos synchronization scheme applied to secure communication," *Chaos, Solitons and Fractals*, vol. 18, no. 1, pp. 141–148, 2003.

[6] F. Sorrentino and P. Delellis, "Estimation of communication-delays through adaptive synchronization of chaos," *Chaos, Solitons & Fractals*, vol. 45, no. 1, pp. 35–46, 2012.

[7] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence*, D. A. Rand and L.-S. Young, Eds., Lecture Notes in Mathematics, pp. 898–910, Springer, New York, NY, USA, 1981.

[8] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, Cambridge University Press, Cambridge, UK, 1997.

[9] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, vol. 33, no. 2, pp. 1134–1140, 1986.

[10] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Physical Review A*, vol. 45, no. 6, pp. 3403–3411, 1992.

[11] H. S. Kim, R. Eykholt, and J. D. Salas, "Nonlinear dynamics, delay times, and embedding windows," *Physica D: Nonlinear Phenomena*, vol. 127, no. 1-2, pp. 48–60, 1999.

[12] R. An, J. Zhang, W. Kong, and B.-J. Ye, "The application of artificial neural networks to the inversion of the positron lifetime spectrum," *Chinese Physics B*, vol. 21, no. 11, Article ID 117803, 2012.

[13] H. Niska, T. Hiltunen, A. Karppinen, J. Ruuskanen, and M. Kolehmainen, "Evolving the neural network model for forecasting air pollution time series," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 2, pp. 159–167, 2004.

[14] Y.-H. Chen and F.-J. Chang, "Evolutionary artificial neural networks for hydrological systems forecasting," *Journal of Hydrology*, vol. 367, no. 1-2, pp. 125–137, 2009.

[15] W. K. Wong, M. Xia, and W. C. Chu, "Adaptive neural network model for time-series forecasting," *European Journal of Operational Research*, vol. 207, no. 2, pp. 807–816, 2010.

[16] K. A. de Oliveira, A. Vannucci, and E. C. da Silva, "Using artificial neural networks to forecast chaotic time series," *Physica A: Statistical Mechanics and Its Applications*, vol. 284, no. 1, pp. 393–404, 2000.

[17] S. Zhang and X.-C. Xiao, "A new method of global prediction for chaotic time series based on continued fractions," *Acta Physica Sinica*, vol. 54, no. 11, pp. 5062–5069, 2005.

[18] Y. D. Zhou, H. Ma, W. Y. Lu, and H. Q. Wang, "Prediction of the chaotic time series using multivariate local polynomial regression," *Acta Physica Sinica*, vol. 56, no. 12, pp. 6809–6815, 2007.

[19] J. Wang, Z. Sheng, B. H. Zhou, and S. D. Zhou, "Lightning potential forecast over Nanjing with denoised sounding-derived indices based on SSA and CS-BP neural network," *Atmospheric Research*, vol. 137, pp. 245–256, 2014.

[20] Y.-G. Tang and X.-P. Guan, "Parameter estimation of chaotic system with time-delay: a differential evolution approach," *Chaos, Solitons and Fractals*, vol. 42, no. 5, pp. 3132–3139, 2009.

[21] J.-H. Ma, Y.-S. Chen, and B.-G. Xin, "Study on prediction methods for dynamic systems of nonlinear chaotic time series," *Applied Mathematics and Mechanics*, vol. 25, no. 6, pp. 605–611, 2004.

[22] T. Suzuki, Y. Ueoka, and H. Sato, "Estimating structure of multivariate systems with genetic algorithms for nonlinear prediction," *Physical Review E*, vol. 80, no. 6, Article ID 066208, 2009.

[23] G. G. Szpiro, "Forecasting chaotic time series with genetic algorithms," *Physical Review E*, vol. 55, no. 3, pp. 2557–2568, 1997.

[24] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[25] M. S. Kim, M. R. Feldman, and C. C. Guest, "Optimum encoding of binary phase-only filters with a simulated annealing algorithm," *Optics Letters*, vol. 14, no. 11, pp. 545–547, 1989.