Hindawi
Computational Intelligence and Neuroscience
Volume 2017, Article ID 8942394, 9 pages
https://doi.org/10.1155/2017/8942394

Research Article

Extracting T–S Fuzzy Models Using the Cuckoo Search Algorithm

Mourad Turki and Anis Sakly

Research Unit of Industrial Systems Study and Renewable Energy (ESIER), National Engineering School of Monastir (ENIM), 5019 Monastir, Tunisia

Correspondence should be addressed to Mourad Turki; mouradessturki@yahoo.fr

Received 18 March 2017; Revised 15 May 2017; Accepted 23 May 2017; Published 6 July 2017

Academic Editor: Paolo Gastaldo

A new method called cuckoo search (CS) is used to extract and learn the Takagi–Sugeno (T–S) fuzzy model. In the proposed method, the particle or cuckoo of CS is formed by the structure of rules in terms of number and selected rules, the antecedent, and consequent parameters of the T–S fuzzy model. These parameters are learned simultaneously. The optimized T–S fuzzy model is validated by using three examples: the first a nonlinear plant modelling problem, the second a Box–Jenkins nonlinear system identification problem, and the third identification of nonlinear system, comparing the obtained results with other existing results of other methods. The proposed CS method gives an optimal T–S fuzzy model with fewer numbers of rules.

1. Introduction

To control any system, it is necessary to obtain an exact model of it but in many cases, it has not enough information to get an acceptable mathematical model, and it is required to use modelling techniques based on input–output data.

Fuzzy models are used due to their excellent performance in the modelling of nonlinear systems and being easy to implement. A fuzzy model is constructed from a basis of rules formed by inputs and outputs of a system [1].

The Takagi–Sugeno (T–S) fuzzy model is a type of fuzzy models which is able to give a local linear representation of the nonlinear system. Such a model is able to approximate a wide class of nonlinear systems because they are considered powerful in modelling and control of complex dynamic systems.

A T–S fuzzy model is powerful if it allows obtaining highly accurate models from a small number of rules but the majority of works in literature provide a large number of rules.

Many works concerning the T–S fuzzy models especially discrete-time type are done in literature such as in [2, 3]. The optimization of T–S fuzzy models is to determine the structure and parameters of model. The methods used to tune the antecedent and consequent parameters are the clustering algorithms [4] and the linear least squares [5–7]. The design of a fuzzy model is a search problem where each point represents a possible fuzzy model with different structures and parameters [1, 8]. In the aim to obtain optimal fuzzy models, many evolutionary algorithms (EAs) such as genetic algorithms (GAs) [9], genetic programming [10], evolutionary programming [11], evolution strategy [12], and differential evolution (DE) [13] have been used [14]. These methods tune the parameters and the structure is predefined.

Though, all parameters of model such as structure and parameters are linked and should be optimized simultaneously. Thereby, in [1] the authors have presented the optimization of rule structure where all the information is encoded into a chromosome.

The particle swarm optimization (PSO) is a novel metaheuristic algorithm used recently in many domains [15]. The PSO algorithm is used to elicit fuzzy models such as in [16] in which PSO optimize the structure, the number of membership functions, and the singleton consequent parameters. In [17], the results of PSO and GA are compared for the same method given in [16] with fixed number of rules and membership function for the same example of simulation.

The structure of the fuzzy model is identified using an online fuzzy clustering method and the parameters are optimized by PSO [20]. In [21], the fuzzy model is extracted using PSO with the recursive least-squares method. The ant colony and PSO were used to obtain T–S fuzzy model in [22]. Lin [23] used immune-based symbiotic PSO to obtain T–S models for the prediction of the skin color detection. In [24], Niu et al. used multiswarm PSO to tune fuzzy systems parameters. In [25], the subtractive clustering is utilized to extract a set of fuzzy rules and a variant of PSO called CPSO algorithm in the aim to find the optimal membership functions and the consequent parameters. In [19], the CRPSO is employed to tune all parameters of the fuzzy models. In [26], the GA is used for learning the T–S fuzzy model from data with a new encoding scheme.

DE and quantum inquired differential evolution is utilized to learn the T–S fuzzy model in [27]. T–S fuzzy models are also developed for modelling industrial systems such as the moving grate biomass furnace in [28].

Other metaheuristic named hunting search (HUS) is used in [18] to determine the parameters of the T–S fuzzy model.

A recent metaheuristic algorithm named the cuckoo search (CS) is proposed in 2009 by Yang. The CS algorithm has given best results compared to other metaheuristics such as PSO and GA. The CS is used to learn neural networks [29] and in the reliability optimization [30]. In [31], the CS is used to tune parameters of Sugeno-Type Fuzzy Logic Controller for liquid level control. Optimizing fuzzy controller using CS in the case study of computer numerical control of a steam condenser is done in [32]. In [33], a prediction of academic performance of student based on the CS is proposed. In [34, 35], the cuckoo search is used in the reduction of high order. In [36], Hammerstein model trained by the cuckoo search algorithm is proposed to identify nonlinear system. In [37], CS is used in the structural damage identification.

The goal and the motivation of this contribution is to obtain a T–S model with minimum number of rules by using a CS method. The objective is to have a model with less complexity able to be easily implemented in embedded systems and with a minimum of errors which proves its efficiency and precision.

This paper describes the use of CS to obtain the optimal structure in terms of a number of rules and also the parameters premises and consequents of T–S model simultaneously in the aim to explore the advantages of CS in optimization which are better compared to other metaheuristics in many examples in literature. The optimal T–S fuzzy models extracted are compared on the same examples with other methods in terms of MSE and number of rules.

This paper is organized as follows. Section 2 describes the structure of T–S fuzzy system. The CS algorithm is introduced in Section 3. Section 4 explains the encoding scheme of T–S model method used. Section 5 presents the different metaheuristic algorithms. Section 6 presents the application examples with results and discussions. Finally, conclusions are given in Section 7.

2. T–S Fuzzy Model

T–S fuzzy model presented in [38] is given by the following basis of rules.

Rule i. If x_1 is A_i^1 and ... and x_{N_I} is $A_i^{N_I}$, then

$$y_i = \alpha_i^0 + \alpha_i^1 x_1 + \cdots + \alpha_i^{N_I} x_{N_I}, \quad (1)$$

where $i = 1, \ldots, N_R$, N_R is the number of rules, $x = [x_1, \ldots, x_{N_I}]$ represents the input, N_I is the size of input, $\alpha_i^0, \alpha_i^1, \ldots, \alpha_i^{N_I}$ are the consequent parameters, y_i is the ith fuzzy rule output, and A_i^j is a fuzzy subset.

The output of the model y is obtained as follows:

$$\widehat{y} = \frac{\sum_{i=1}^{N_R} \omega_i y_i}{\sum_{i=1}^{N_R} \omega_i}, \quad (2)$$

where ω_i of the ith rule is computed as

$$\omega_i = \prod_{j=1}^{n} \mu\left(A_i^j\right) \quad (3)$$

with $\mu(A_i^j)$ being the membership function's grade of A_i^j and is characterized by a Gaussian function as

$$\mu_{A_i^j}\left(x_j\right) = \exp\left(-\frac{1}{2}\left(\frac{x_j - c_i^j}{\sigma_i^j}\right)^2\right). \quad (4)$$

c_i^j and σ_i^j are, respectively, the mean and the deviation of the MF. The premise parameters c_i^j and σ_i^j are adjusted [20].

3. Cuckoo Search

CS is developed by Yang and Deb in 2009 [39–41]. CS imitates the parasitism of cuckoo and used Lévy Flights which are better than random walks [32].

The CS has the specificity that the cuckoos lay their eggs in the nests of host birds. Some cuckoos can mimic the properties of the host eggs.

As a result, the number of the eggs abandoned is reduced and their reproductivity increases [42].

The CS models are used in many optimization problems. In [40, 43], it is demonstrated that Lévy Flights are better than random walk in CS algorithm.

In CS algorithm, the solution is given by an egg in a nest, and a new solution is represented by a cuckoo egg. The goal is to use the newer cuckoos or solutions to substitute worst solutions. In the classical algorithm, each nest has one egg; however the algorithm can be extended to complex problem [40, 43].

In the cuckoo search, the rules are as follows:

(i) Every cuckoo lays a single egg and throws it in a random nest.

(ii) The best nests or solutions will be transferred to the next generations.

```
begin
    Choose an objective function $f(x)$, $x = (x_1, \dots, x_d)^T$;
    Generate initial population of $n$ host nests $x_i$ $(i = 1, 2, \dots, n)$;
    While (stop criterion);
        Get a cuckoo (say $i$) randomly by Lévy Flights;
        Calculate its fitness $F_i$;
        Choose a nest among $n$ (say $j$) randomly;
        If $(F_i > F_j)$,
            Change $j$ by the new solution;
        end if
        A probability $(p_a)$ of worse nests is abandoned and new solutions are made;
        Conserve the best solutions;
        Classify the solutions and find the current best;
    end while
end
```

Pseudocode 1

(iii) The number of host nests is predefined, and a host can detect a stranger egg with probability $p_a \in [0, 1]$. In that event, the host bird skips the egg or abandons the nest and builds a new nest in another place [43].

The CS algorithm is described in Pseudocode 1 [43].

In our work, $f(x)$ is the MSE and nests x_i are possible T–S fuzzy models with cuckoos being the parameters of the T–S fuzzy model.

This algorithm uses a combination of a local random walk and the global random walk, controlled by parameter p_a. The local random walk can be written as

$$x_i^{t+1} = x_i^t + \alpha s \otimes H(p_a - \varepsilon) \otimes (x_j^t - x_k^t), \quad (5)$$

where x_j^t and x_k^t are two different solutions, $H(u)$ is a Heaviside function, ε is a random number drawn from a uniform distribution, and s is the step size. The global random walk is carried out by using Lévy Flights.

$$H(u) = \begin{cases} 0, & \text{if } u < 0 \\ 1, & \text{if } u \geq 0 \end{cases} \quad (6)$$
$$x_i^{t+1} = x_i^t + \alpha L(s, \lambda),$$

where

$$L(s, \lambda) = \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \cdot \frac{1}{s^{1+\lambda}} \quad (7)$$

with

$$\Gamma(\lambda) = \int_0^{+\infty} t^{\lambda-1} e^{-t}\, dt. \quad (8)$$

Here, $\alpha > 0$ is the step size scaling factor, which should be related to the scales of the problem of interest [44]. In fact, we use the Lévy Flights to obtain other T–S fuzzy models in the next generation which can be a solution.

Recent studies utilize cuckoo search such as Walton et al. by formulating a modified cuckoo search algorithm [45]; Yang and Deb improve it to multiobjective optimization [39].

Rule 1	⋯	Rule i	⋯	Rule Nmax
↓	↓	↓	↓	↓
Premise parameters	⋯	Premise parameters	⋯	Premise parameters
Consequent parameters	⋯	Consequent parameters	⋯	Consequent parameters
Label 1	⋯	Label i	⋯	Label Nmax

Figure 1: The particle structure of encoding a fuzzy rule base.

4. Encoding Scheme for T–S Fuzzy Model

In this paper, the fuzzy system is given by a particle formed by the premise and the consequent parameters and also the labels which are used to choose the rules to construct the fuzzy system [20]. The fuzzy model's particle is presented in Figure 1.

In Figure 1, each rule i is formed by premise and consequent parameters and the label.

Figure 2 shows that the particle i is given by a vector composed of the premise parameters $\sigma_i^1, c_i^1, \dots, \sigma_i^{N_I}, c_i^{N_I}$, consequent parameters $\alpha_i^0, \alpha_i^1, \dots, \alpha_i^{N_I}$, and the label l_i of all rules.

The fuzzy rules are selected using the labels in fact. If $l_i > 0$, then the rule i is selected where $i = 1, \dots, N_{\max}$ is the index of the rule. The T–S fuzzy system is composed of the active rules.

The CS algorithm is used to elicit T–S fuzzy model and presented as follows:

(1) Encoding all the parameters premise and consequent of all rules with a predefined maximum number of rules $N_{\max}$.

(2) Defining a fitness function and the bounds of parameters.

(3) Randomizing an initial swarm of nests. Initializing all the parameters of particles representing fuzzy models with the lower and upper bounds chosen of parameters. Every nest is a fuzzy model with different structures and parameters.

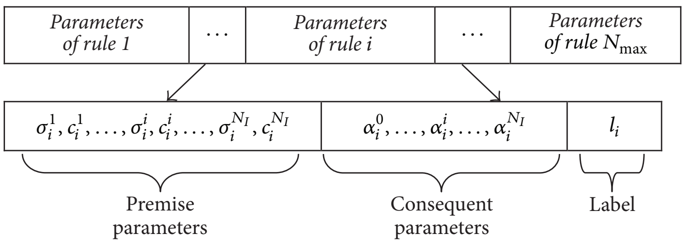
Parameters of rule 1
...
Parameters of rule i
...
Parameters of rule $N_{\max}$
$\sigma_i^1, c_i^1, \ldots, \sigma_i^i, c_i^i, \ldots, \sigma_i^{N_I}, c_i^{N_I}$
$\alpha_i^0, \ldots, \alpha_i^i, \ldots, \alpha_i^{N_I}$
l_i
Premise parameters
Consequent parameters
Label

FIGURE 2: The encoding scheme of a fuzzy model.

(4) Calculating the fitness of initials nests which is MSE given by this equation:

$$\mathrm{MSE} = \frac{1}{n}\sum_{k=1}^{n}\left(y_{\mathrm{ref}}(k) - y(k)\right)^2. \tag{9}$$

n is the number of input prototypes, $y_{\mathrm{ref}}(k)$ is the desired output, and $y(k)$ is the model output.

(5) Using the CS to search the optimal T–S fuzzy model.

Step 1. Get a fuzzy model (cuckoo) FM_1 from the swarm of nests (TS fuzzy models generated randomly) randomly by using Lévy Flights, calculate its fitness, and select another fuzzy model (nest) FM_2 randomly among the n nests of the swarm.

Step 2. If $\mathrm{MSE}(\mathrm{FM}_1) > \mathrm{MSE}(\mathrm{FM}_2)$, replace FM_2 by the new solution FM_1; otherwise pass to the next step.

Step 3. p_a worst fuzzy models or nests (each nest is a fuzzy model in our work) are abandoned and new ones are built.

Step 4. Test the stopping criterion; if it is verified, keep the best fuzzy model (the optimal premise, consequent, and number of rules) and otherwise return to Step 1.

5. Metaheuristic Algorithms

There are many metaheuristic algorithms in literature such as the particle swarm optimization (PSO), the cooperative random learning PSO called CRPSO, the hunting search (HUS), the genetic algorithms (GA), and the differential evolution (DE).

5.1. The Particle Swarm Optimization (PSO). The particle swarm optimization (PSO) imitates the movement of birds flocking or fish schooling looking for food [19]. The research of optimal solution is given by two equations:

$$\begin{aligned} v(t+1) &= w \cdot v(t) + c_1 r_1 \left[p - x(t)\right] \\ &\quad + c_2 r_2 \left[p_g - x(t)\right] \\ x(t+1) &= x(t) + v(t+1), \end{aligned} \tag{10}$$

where x is the position of a particle, v is the velocity, w is the inertia weight, c_1 and c_2 are constants, r_1 and r_2 are random numbers between 0 and 1, p is the best position of the particle, and p_g is the best position of all particles in the swarm.

Another version of PSO is the cooperative random learning PSO (CRPSO) which used subswarms and the equation of velocity is given as follows:

$$\begin{aligned} v_j(t+1) &= w \cdot v_j(t) + c_1 r_1 \left[p_j - x_j(t)\right] \\ &\quad + c_2 r_2 \left[p_g(j) - x_j(t)\right] \\ &\quad + c_3 r_3 \left[p_g(r) - x_j(t)\right], \end{aligned} \tag{11}$$

where c_3 is constant, r_3 is random number, j is the index of subswarm, r is the number between 1 and the number of subswarms, and p_g is the global best position of all subswarms [19].

5.2. The Hunting Search (HUS). The hunting search (HUS) algorithm imitates the social behavior of animals when catching a prey in the way wolves hunt. The algorithm is based on approaching the leader having the best position in the group and reorganizing if the hunters are close to each other but still cannot find the optimum solution [18].

The research of new solution obeys this equation:

$$x_i = x_i + r \cdot \mathrm{NML} \cdot \left(x_i^l - x_i\right), \tag{12}$$

where r is a random number between 0 and 1, NML is the maximum number of movements toward the leader, and x_i^l is the position of leader in the ith variable.

Another step is the reorganization of hunters which is given by this equation:

$$\begin{aligned} x_i = x_i^l &\pm \mathrm{rand} \cdot \left(\max(x_i) - \min(x_i)\right) \cdot \alpha \\ &\cdot \exp(-\beta \cdot \mathrm{EN}), \end{aligned} \tag{13}$$

where EN is the number of past reorganizations and α and β are positive constants, with the aim to avoid falling into a local minimum and obtain a globally optimal solution.

5.3. The Genetic Algorithm (GA). The genetic algorithm (GA) is a random search technique which imitates natural evolution with Darwinian survival of the fittest approach. In this algorithm, the variables are represented as genes in a chromosome, and the chromosomes are evaluated according to their fitness values. The chromosomes with better fitness

Table 1: CS parameter values.

Parameter	Value
Number of nests	20
p_a	0.25
Number of iterations	500
λ	1.5
Step size α	1
Maximum number of rules Nmax	5

are found through the three basic operations of GA: selection, crossover, and mutation [46].

The algorithm of GA is described as follows:

(1) Initialization of initial population called chromosomes.

(2) Evaluation of each element in the population by calculating its fitness function.

(3) Selection of the chromosomes.

(4) Generation of new chromosomes using the chromosomes selected and the GA's operations such as crossover and mutation.

(5) Test of the stopping criterion: if validated, then the parameters are kept; otherwise return to Step (2).

5.4. The Differential Evolution (DE). The differential evolution (DE) is a search algorithm that is similar to GA; it deals with a real coded population and devises its own crossover and mutation in the real space [13]. DE creates x^0, a mutated form of any individual x, using the vector difference of randomly picked individuals called x^* and x° using this equation:

$$x^0 = x + \gamma \left(x^* - x^\circ \right), \tag{14}$$

where γ is a scaling factor between 0 and 2. Then, the crossover is applied between any individual member of the population and the mutated vector x^0 and the best element is kept in the last iteration.

6. Application Examples and Discussions

In this part, the T–S models optimized are used to identify three systems: a nonlinear plant modelling problem, the Box–Jenkins gas furnace benchmark, and identification of nonlinear system.

The performance of CS is compared with other metaheuristic algorithms. The parameters used in the examples are presented in Table 1.

6.1. Application to Nonlinear Plant Modelling Problem. In [1, 5, 6], the nonlinear dynamic plant given by this nonlinear difference equation has been modelled.

$$y(k) = g\left(y(k-1), y(k-2)\right) + u(k), \tag{15}$$

Table 2: Performance comparison of different methods for nonlinear plant modelling.

Method	Number of rules	MSE (training)		MSE (testing)	
	Mean	Mean	Std	Mean	Std
CS	*3.3*	*0.0016*	*0.0005*	*0.0009*	*0.0003*
HUS [18]	4.84	0.0017	0.0015	0.0029	0.0016
CRPSO [19]	5.6	0.0020	0.0016	0.0035	0.0044
PSO [19]	5.64	0.0055	0.0055	0.0063	0.0072
GA [19]	3.62	0.0038	0.0053	0.0045	0.0051
DE [19]	5.32	0.0081	0.0041	0.0086	0.0062

where

$$g\left(y(k-1), y(k-2)\right) = \frac{y(k-1)\,y(k-2)\left(y(k-1)-0.5\right)}{1 + y(k-1)^2 + y(k-2)^2}. \tag{16}$$

The aim of this application is to identify the nonlinear component $g(y(k-1), y(k-2))$ using the presented fuzzy model with CS algorithm. The example has two inputs and one output. The simulated data are formed by 400 points which are generated from the plant model: 200 data points are calculated using a random input signal $u(k)$ uniformly distributed in $[-1.5, 1.5]$, and other 200 data points are computed using a sinusoidal input signal $u(k) = \sin(2\pi k/25)$ [1, 5, 6].

The number of generations chosen is 500 and was iterated 50 times on a Pentium Core 2 Duo (1.8 GHz CPU) and 2 GB memory personal computer in the same computing environment (MATLAB 2007a). The consequent parameters are chosen in $[-10, 10]$ and the width of the premises parameters is given in $[0, 5]$.

As those in [1, 5, 6], we select $y(k-1)$ and $y(k-2)$ as inputs in the aim to predict the nonlinear component $g(y(k-1), y(k-2))$ and 5 is the maximum number of rules. Table 2 gives the best results of 50 experimental trials.

According to Table 2, CS method gives the best results in terms of a number of rules (mean) fewer than the HUS method in [18], MSE (mean), and standard deviation (Std) in both training and testing stages compared with other methods. Also, the CS method gives good performances with the smaller number of evaluations than the result in [19]. In fact, in [19] 1000 generations and 20 particles are used with CRPSO algorithm and 2000 generations and 30 particles with PSO, GA, and DE; however in our work we use 500 generations and 20 particles which are less than the previous algorithms.

The optimal fuzzy model gives an MSE $8 * 10^{-4}$ in training and $4 * 10^{-4}$ in testing.

Figure 3 indicates outputs of target and model in the training and testing stages of the optimal model and the errors between them can be seen in Figure 4. As follows in Figure 3, the CS method gives the output with small errors.

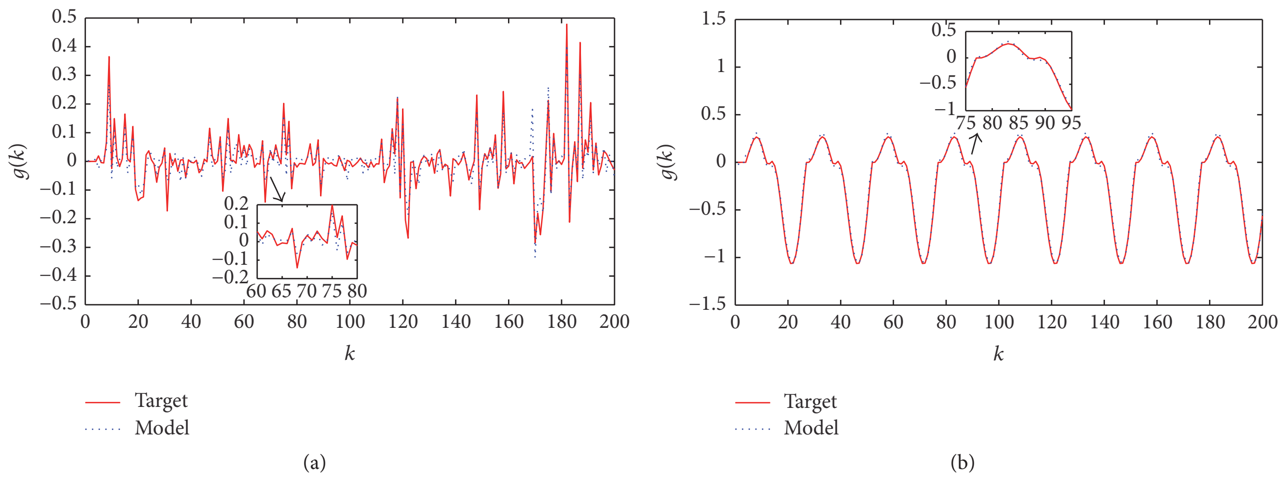

FIGURE 3: The target output and model output for nonlinear plant modelling problem: (a) training stage and (b) testing stage.

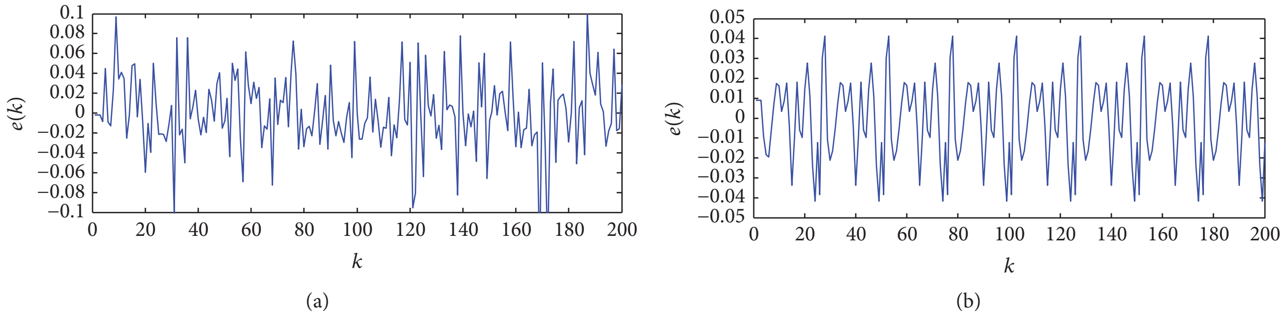

FIGURE 4: The errors between target output and model output: (a) training stage and (b) testing stage.

6.2. Application to Box–Jenkins Gas Furnace Data. The Box–Jenkins gas furnace data [1, 11, 34–36] was recorded from a combustion process of a methane–air mixture [44]. The data set originally consists of 296 data points $[y(t), u(t)]$. The input $u(t)$ is the gas flow rate; however the output $y(t)$ was the carbon dioxide (CO_2) concentration. The aim is to elicit a model to predict $y(t)$ using this data. The first step is to determine the appropriate inputs to be used. The initial fuzzy inputs are $y(t-1)$, $y(t-2)$, $y(t-3)$, $y(t-4)$, $u(t-1)$, $u(t-2)$, $u(t-3)$, $u(t-4)$, $u(t-5)$, and $u(t-6)$, and the output is $y(t)$. Due to many studies in literature, $y(t-1)$ and $u(t-4)$ are chosen as inputs. All the coefficients of the consequent parameters are chosen in $[-100, 100]$ and the width of the Gaussian function is limited to $[0, 10]$.

All the simulations are executed 50 times. The mean number of rules, the mean, and standard deviations of the MSE are listed in Table 3.

From Table 3, we conclude that CS has minimum mean MSE compared to PSO, HUS, and DE and less standard deviation MSE than PSO, CRPSO, HUS, and DE. The mean number of the rules of CS is much smaller than HUS, PSO, CPSO, GA, and DE. In conclusion, the CS-based method can give a fuzzy model with less number of rules. The optimal fuzzy model found by CS during 50 runs has an MSE of 0.139 and 3 rules.

TABLE 3: Performance comparison of different methods for Box–Jenkins gas furnace data set.

Method	Number of rules (mean)	MSE (mean)	MSE (Std)
CS	*2.34*	*0.1527*	*0.0129*
HUS	2.68	0.1796	0.0300
PSO [19]	3.64	0.1550	0.0427
CRPSO [19]	3.96	0.1428	0.0138
GA [19]	4.32	0.1474	0.0109
DE [19]	4.96	0.1768	0.0602

Figure 5 shows the target and the model outputs and Figure 6 gives the errors between them. The optimal fuzzy model can identify the output with small errors.

Table 4 gives the parameters of the optimal fuzzy model.

6.3. Identification of Nonlinear System. The third example used for identification, given by Narendra and Parthasarathy, is described by the next difference equation [47]:

$$y(k+1) = \frac{y(k)}{1 + y^2(k)} + u^3(k). \tag{17}$$

Table 4: The optimal fuzzy model for identification of Box–Jenkins gas furnace problem.

Rule base	$y(t-1)(\sigma_1, c_1)$	$u(t-4)(\sigma_2, c_2)$	$y(t)$
Rule 1	(4.64, 48.36)	(2.72, 2.00)	$-0.32y(t-1) - 2.67u(t-4) + 72.61$
Rule 2	(4.77, 50.72)	(1.86, 2.49)	$1.91y(t-1) + 1.23u(t-4) - 51.66$
Rule 3	(2.67, 48.00)	(1.32, −1.91)	$2.20y(t-1) + 3.67u(t-4) - 62.00$

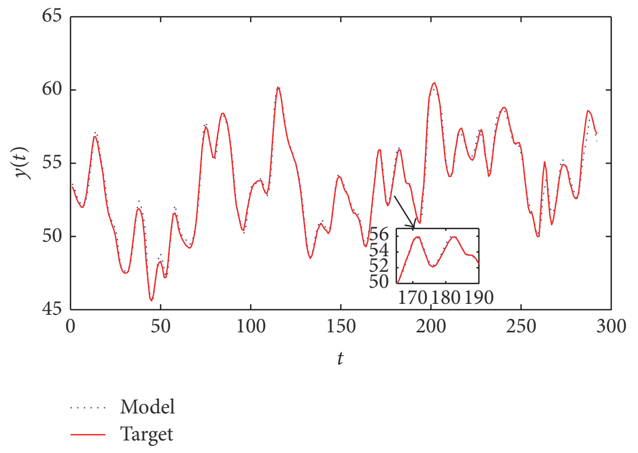

Figure 5: Fuzzy model output and real output for modelling the BJ gas furnace data.

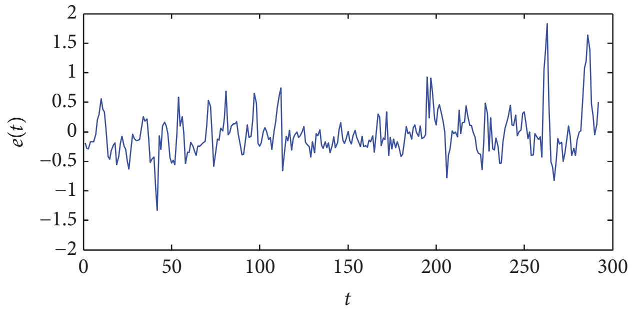

Figure 6: The errors between fuzzy model output and real output.

The input $u(k)$ is given by this equation:

$$u(k) = \sin\left(\frac{2\pi k}{25}\right). \tag{18}$$

The output of this equation depends nonlinearly on both its past values and the input. The 200 training input patterns are randomly generated in the interval $[-1 \quad 1]$ by using (17).

The aim of this application is to predict $y(k)$ by using this approach when the inputs chosen are $y(k-1)$ and $u(k-1)$.

All the coefficients of the premise and consequent parameters are limited to $[-5, 5]$. The maximum number of rules is chosen as 5. Table 5 gives the best results of 50 experimental trials.

For all methods, the number of generations is fixed to 500 and the number of particles is fixed to 20.

As shown in Table 5, CS gives a minimum number of rules, less mean of MSE, and less standard deviation MSE compared to PSO, HUS, and GA.

Table 5: Performance comparison of different methods for identification of nonlinear system.

Method	Number of rules (mean)	MSE (mean)	MSE (Std)
CS	*2.40*	*0.0263*	*0.0031*
HUS	3.08	0.0192	0.0116
PSO	2.54	0.0330	0.0059
GA	2.64	0.1236	0.1694

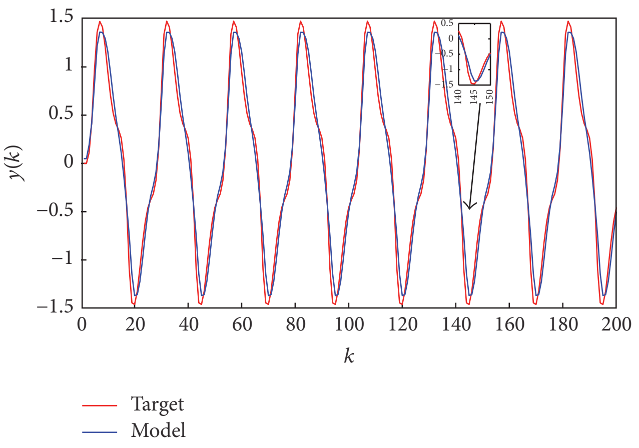

Figure 7: Fuzzy model output and target output for identification of the nonlinear system.

The optimal fuzzy model found by CS during 50 runs has an MSE of 0.0231 and 3 rules.

Figure 7 shows the target and the model outputs and Figure 8 gives the errors between them. The optimal fuzzy model can predict the output with small errors.

7. Conclusions

In this paper, the extracting of T–S fuzzy model using CS algorithm is described. The T–S fuzzy model tuned by CS has the rules structures and both the premises and consequents parameters optimized. The optimal T–S fuzzy model has a fewer number of rules and smaller MSE both in mean and in standard deviation. The T–S model using CS algorithm is validated by the comparison of its performance to other methods for modelling three benchmarks: the nonlinear plant modelling problem, the Box–Jenkins problem, and identification of nonlinear system and this shows that the CS algorithm gives much better accuracy in modelling nonlinear systems; in fact, CS gives a model with minimum of number of rules with better errors compared to other metaheuristics.

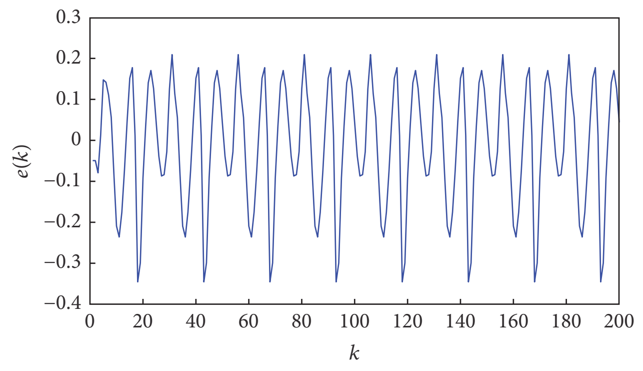

Figure 8: The errors between fuzzy model output and target output.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] M.-S. Kim, C.-H. Kim, and J.-J. Lee, "Evolving compact and interpretable Takagi-Sugeno fuzzy models with a new encoding scheme," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 36, no. 5, pp. 1006–1023, 2006.

[2] X. Xie, D. Yue, T. Ma, and X. Zhu, "Further studies on control synthesis of discrete-time T-S Fuzzy systems via augmented multi-indexed matrix approach," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2784–2791, 2014.

[3] X. Xie, D. Yue, H. Zhang, and Y. Xue, "Fault Estimation Observer Design for Discrete-Time Takagi-Sugeno Fuzzy Systems Based on Homogenous Polynomially Parameter-Dependent Lyapunov Functions," *IEEE Transactions on Cybernetics*, 2017.

[4] K. K. Singh, M. J. Nigam, K. Pal, and A. Mehrotra, "A fuzzy Kohonen local information c-means clustering for remote sensing imagery," *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, vol. 31, no. 1, pp. 75–81, 2014.

[5] H.-J. Rong, N. Sundararajan, G.-B. Huang, and P. Saratchandran, "Sequential adaptive fuzzy inference system (SAFIS) for nonlinear system identification and prediction," *Fuzzy Sets and Systems*, vol. 157, no. 9, pp. 1260–1275, 2006.

[6] L. Wang and J. Yen, "Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and Kalman filter," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 101, no. 3, pp. 353–362, 1999.

[7] P. P. Angelov and D. P. Filev, "An approach to online identification of Takagi-Sugeno fuzzy models," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 1, pp. 484–498, 2004.

[8] Y. Shi, R. Eberhart, and Y. Chen, "Implementation of evolutionary fuzzy systems," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 2, pp. 109–119, 1999.

[9] O. Cordón and F. Herrera, "A two-stage evolutionary process for designing TSK fuzzy rule-based systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 6, pp. 703–715, 1999.

[10] O. Cordon, F. Herrera, F. Gomide, F. Hoffmann, and L. Magdalena, "Ten years of genetic fuzzy systems: current framework and new trends," in *Proceedings of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, pp. 1241–1246, Vancouver, BC, Canada, 2001.

[11] S.-J. Kang, C.-H. Woo, H.-S. Hwang, and K. B. Woo, "Evolutionary design of fuzzy rule base for nonlinear system modeling and control," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 1, pp. 37–45, 2000.

[12] W. Pedrycz and M. Reformat, "Evolutionary fuzzy modeling," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 5, pp. 652–665, 2003.

[13] M. Eftekhari, S. D. Katebi, M. Karimi, and A. H. Jahanmiri, "Eliciting transparent fuzzy model using differential evolution," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 466–476, 2008.

[14] S. Mitra and Y. Hayashi, "Neuro-fuzzy rule generation: survey in soft computing framework," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 748–768, 2000.

[15] V. Mangat and R. Vig, "Dynamic PSO-based associative classifier for medical datasets," *IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India)*, vol. 31, no. 4, pp. 258–265, 2014.

[16] A. Khosla, S. Kumar, and K. K. Aggarwal, "A framework for identification of fuzzy models through particle swarm optimization algorithm," in *Proceedings of the INDICON 2005: An International Conference of IEEE India Council*, pp. 388–391, ind, December 2005.

[17] A. Khosla, S. Kumar, and K. R. Ghosh, "A comparison of computational efforts between particle swarm optimization and genetic algorithm for identification of fuzzy models," in *Proceedings of the Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS '07)*, pp. 245–250, June 2007.

[18] S. Bouzaida, A. Sakly, and F. M'Sahli, "Extracting TSK-type neuro-fuzzy model using the hunting search algorithm," *International Journal of General Systems*, vol. 43, no. 1, pp. 32–43, 2014.

[19] L. Zhao, F. Qian, Y. Yang, Y. Zeng, and H. Su, "Automatically extracting T-S fuzzy models using cooperative random learning particle swarm optimization," *Applied Soft Computing Journal*, vol. 10, no. 3, pp. 938–944, 2010.

[20] C.-F. Juang, I.-F. Chung, and C.-H. Hsu, "Automatic construction of feedfoward/recurrent fuzzy systems by clustering-aided simplex particle swarm optimization," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 158, no. 18, pp. 1979–1996, 2007.

[21] C.-C. Chen, "A PSO-based method for extracting fuzzy rules directly from numerical data," *Cybernetics and Systems*, vol. 37, no. 7, pp. 707–723, 2006.

[22] C.-F. Juang and C. Lo, "Zero-order TSK-type fuzzy system learning using a two-phase swarm intelligence algorithm," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 159, no. 21, pp. 2910–2926, 2008.

[23] C.-J. Lin, "An efficient immune-based symbiotic particle swarm optimization learning algorithm for TSK-type neuro-fuzzy networks design," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 159, no. 21, pp. 2890–2909, 2008.

[24] B. Niu, Y. Zhu, X. He, and H. Shen, "A multi-swarm optimizer based fuzzy modeling approach for dynamic systems processing," *Neurocomputing*, vol. 71, no. 7-9, pp. 1436–1448, 2008.

[25] L. Zhao, Y. Yang, and Y. Zeng, "Eliciting compact T-S fuzzy models using subtractive clustering and coevolutionary particle swarm optimization," *Neurocomputing*, vol. 72, no. 10-12, pp. 2569–2575, 2009.

[26] H. Du and N. Zhang, "Application of evolving Takagi-Sugeno fuzzy model to nonlinear system identification," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 676–686, 2008.

[27] H. Su and Y. Yang, "Differential evolution and quantum-inquired differential evolution for evolving Takagi-Sugeno fuzzy models," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6447–6451, 2011.

[28] S. Grosswindhager, L. Haffner, A. Voigt, and M. Kozek, "Fuzzy modelling of a moving grate biomass furnace for simulation and control purposes," *Mathematical and Computer Modelling of Dynamical Systems. Methods, Tools and Applications in Engineering and Related Sciences*, vol. 20, no. 2, pp. 194–208, 2014.

[29] E. Valian, S. Mohanna, and S. Tavakoli, "Improved Cuckoo Search Algorithm for Feed forward Neural Network Training," *International Journal of Artificial Intelligence & Applications*, vol. 2, no. 3, pp. 36–43, 2011.

[30] E. Valian, S. Tavakoli, S. Mohanna, and A. Haghi, "Improved cuckoo search for reliability optimization problems," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 459–468, 2013.

[31] A. Aghaei, K. Kiani, and M. Bayati, "Optimizing fuzzy controller using cuckoo optimization algorithm (COA)," *International Journal of Enhanced Research in Science Technology & Engineering*, vol. 3, no. 12, pp. 1–10, December 2014.

[32] J. F. Chen and Q. H. Do, "A cooperative cuckoo search – hierarchical adaptive neuro-fuzzy inference system approach for predicting student academic performance," *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology*, vol. 27, no. 5, pp. 2551–2561, Sep 2014.

[33] G. E. P. Box and G. M. Jenkins, *Time Series Analysis, Forecasting and Control*, Holden-day, San Francisco, Calif, USA, 1970.

[34] A. Afzal Sikander and B. Rajendra Prasad, "A novel order reduction method using cuckoo search algorithm," *IETE Journal of Research*, vol. 61, no. 2, pp. 83–90, 2015.

[35] A. Narwal and B. R. Prasad, "A novel order reduction approach for LTI systems using cuckoo search optimization and stability equation," *IETE Journal of Research*, vol. 62, no. 2, pp. 154–163, 2016.

[36] A. Gotmare, R. Patidar, and N. V. George, "Nonlinear system identification using a cuckoo search optimized adaptive Hammerstein model," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2538–2546, 2015.

[37] H. J. Xu, J. K. Liu, and Z. R. Lu, "Structural damage identification based on cuckoo search algorithm," *Advances in Structural Engineering*, vol. 19, no. 5, pp. 849–859, 2016.

[38] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[39] X.-S. Yang and S. Deb, "Multiobjective cuckoo search for design optimization," *Computers & Operations Research*, vol. 40, no. 6, pp. 1616–1624, 2013.

[40] X.-S. Yang and S. Deb, "Engineering optimisation by Cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.

[41] S. Balochian and E. Ebrahimi, "Parameter Optimization via Cuckoo Optimization Algorithm of Fuzzy Controller for Liquid Level Control," *Journal of Engineering (United States)*, vol. 2013, Article ID 982354, 2013.

[42] R. Payne, M. Sorenson, and K. Klitz, *The Cuckoos*, Oxford University Press, 2005.

[43] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.

[44] X. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.

[45] S. Walton, O. Hassan, K. Morgan, and M. R. Brown, "Modified cuckoo search: a new gradient free optimisation algorithm," *Chaos, Solitons and Fractals*, vol. 44, no. 9, pp. 710–718, 2011.

[46] M. Turki, S. Bouzaida, A. Sakly, and F. M'Sahli, "Modeling and on line control of nonlinear systems using neuro-fuzzy learning tuned by metaheuristic algorithms," *International Journal of Control and Automation*, vol. 7, no. 5, pp. 323–342, 2014.

[47] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.