International Journal of Hybrid Information Technology
Vol. 10, No.1 (2017), pp. 99-110
http://dx.doi.org/10.14257/ijhit.2017.10.1.09

Modified Comprehensive Learning Particle Swarm Optimization for Numerical and Takagi-Sugeno Fuzzy System Modeling

Guohan Lin[1,2] and Kuiyin Zhao[2]

[1]*College of Electrical and Information, Hunan Institute of Engineering, 411101 Xiangtan, Hunan, P.R.China*
[2] *Key Laboratory of Electrical control, University of Hunan Province, Hunan Institute of Engineering, 411101 Xiangtan, Hunan , P.R.China*
lgh@hnu.edu.cn, zidonghua04@126.com

Abstract

Modifications for comprehensive learning particle swarm optimization (M-CLPSO) is proposed for numerical problems and modeling Takagi-Sugeno(T-S) Fuzzy System. A self-adaptive strategy is adopted to adjust the value of acceleration coefficient dynamically. In the late stage of the evolution, Gaussian disturbance is hydride with algorithm to help the stagnant particles to escape from standstill state. The effectiveness of the proposed algorithm is verified by numerical experiments and T-S fuzzy system modeling. The experiments results show that the proposed method can provide comparable improvement in solving function optimization problems and can generate good fuzzy system model with high accuracy.

Keywords*: Particle swarm optimization (PSO), Gaussian disturbance, Self-adaptive strategy, Takagi-Sugeno fuzzy.*

1. Introduction

Particle swarm optimization algorithm(PSO) is one of the most popular evolutionary algorithm , which is inspired by behavior of bird flocking and is proposed firstly by Kennedy and Eberhart in 1995[1,2]. As a kind of global optimization evolutionary algorithm, PSO have gained widely used in science and engineering fields due to its simple implementation and excellence optimize performance [3, 4, 5]. However, similar to others population-based algorithms, PSO algorithm converges faster in the early stage and prone to loss diversity which will lead to the particles trapped in local optima and fall into premature convergence. To improve the performance of PSO, researchers have proposed many improved PSO algorithm. Although these improved algorithms can find better optimization results, many of these methods can be trapped in local optimum when solving complex multimodal function. Liang[6] put forward an improved PSO algorithm named comprehensive learning particle swarm optimization (CLPSO), which used a new strategy to update the particles' velocity. CLPSO has been proved to be one of the best performance algorithms, especially for complex multimodal function optimization. However, on some functions which fluctuate more intensively and have much more local optima, CLPSO does not work well [6]. Some modification has been presented to improve the performance of CLPSO. Adaptive comprehensive learning particle swarm optimizer with history learning (AH-CLPSO) was proposed by Liang to solve unimodal function problems [7]. CLPSO combined with Broyden-Fletcher-Goldfarb-Shanno method is proposed for tuning the parameter of support vector machine. In [8], WU proposed an improved CLPSO to guide antenna design.

In this paper, a novel modified comprehensive learning particle swarm optimization, called M-CLPSO is proposed. The algorithm introduced the growth operator with which

ISSN: 1738-9968 IJHIT

dynamically changes the value of acceleration coefficient. When the particles accumulate to a certain degree, Gaussian disturbance is used to reactive the stagnant particles. Simulation results on benchmark functions showed that the M-CLPSO algorithm can solve function optimization problems more efficiently compare with CLPSO and other improved PSO. The M-CLPSO's performance is further verified by its application to model the T-S fuzzy system.

2. Modified CLPSO

2.1. PSO Algorithm

In the PSO, a swarm is generated randomly in *D*-dimension search space. Each particle in the swarm is represented as a candidate solution and represent by a velocity vector $V_i=(v_{i1},v_{i2},...,v_{iD})$ and a position vector $X_i=(x_{i1},x_{i2},...,x_{iD})$. Each particle adjusts it position and velocity by learning from its individual best position denotes by *pbest*, and the global best position denoted by *gbest*. The velocity and position of particle *i* are altered by the following recursive equation.

$$V_{id}(t+1)=V_{id}(t)+c_1*r_1(t)(p_{best}(t)-X_{id}(t))+c_2*r_2(t)(g_{best}(t)-X_{id}(t)) \quad (1)$$

$$X_{id}(t+1)=X_{id}(t)+V_{id}(t+1) \quad (2)$$

where $X_{id}(t)$ and $V_{id}(t)$ is the current position and current velocity of the particle *i* respectively.

c_1 and c_2 are cognitive and social acceleration factor respectively.

r_1 and r_2 are uniformly distributed random numbers between 0 and 1.

Inertia weight is introduced by Shi in [9] to balance the exploration and exploitation ability of PSO. The most widely used form of velocity update equation in PSO algorithms is

$$V_{id}(t+1)=wV_{id}(t)+c_1*r_1(t)(p_{best}(t)-X_{id}(t))+c_2*r_2(t)(g_{best}(t)-X_{id}(t)) \quad (3)$$

where ω is the inertia weight.

2.2. CLPSO

The basic PSO is easily trapped into local optima due to the diversity of the particles decrease rapidly. To maintain the diversity of particle and avoid of the premature convergence, new learning strategy is adopted in CLPSO, that is the particle learns from the other particles' historical best position *pbest* other than *gbest*. The velocity updating equation in CLPSO is

$$V_{id}=\omega V_{id}+c_1*r_1(Pbest_{f_i(d)}-X_{id}) \quad (4)$$

where $Pbest_{f_i(d)}$ indicates which particles' *pbest* will be chosen to learn from according to a learning probability $P_{ci.}$ The value of P_{ci} for particle *i* is set at the beginning of searching process according to following equation:

$$P_{ci}=0.05+0.45*\frac{\left(\exp(\frac{10(i-1)}{N-1})-1\right)}{(\exp(10)-1)} \quad (5)$$

where N is the population size.

2.3. Modified CLPSO

As an improved version of PSO algorithm, CLPSO algorithm has been proved to perform better than other PSO variants on multimodal function optimization problems. However, CLPSO has the drawback of slow convergence rate and unreliable if the local optima is not good enough. There are some potential disadvantages exist in CLPSO [10].

1) When the $Pbest_{f_i(d)}$ of all the particles are very close to each other, which may make the particles trap in local optima.

2) As the solution represent by $Pbest_{f_i(d)}$ are generally poorer than that of $gbest$, which may cause the problem of slow convergence.

3) The solution quality depend largely on the $Pbest_{f_i(d)}$, and the $Pbest_{f_i(d)}$ is initiated blindly, which may make the algorithm unstable.

In the late stage of CLPSO, the particle velocity gradually tends to stop, result to converge slowly, to make particles have sufficient energy searching in the solution space, an event-trigger approach is applied to the CLPSO algorithm.

Definition 1. Suppose $f_{best}^{i}(k)$ and $f_{best}^{i}(k-k_1)$ are the current best fitness value of particle i in k generation and k-k_1 generation respectively, a factor name as growth rate for each particle is defined by:

$$g_i = \frac{f_{best}^{i}(k) - f_{best}^{i}(k-k_1)}{f_{best}^{i}(k-k_1) + \varepsilon} \tag{6}$$

where $\varepsilon > 0$ is the smoothing coefficient.

When the fitness value of a particle i at the k generation is higher than that of the particle at the k-k_1 generation, the g_i will increase and that mean the particles can search effectively in the searching space, otherwise, when the value of g_i is smaller than a preset threshold value , the particles tend to be standstill. In proposed algorithm, Gaussian disturbance is applied to these particles to help them keep active.

In proposed algorithm, the formula for updating the velocity of the particles is modified as follow

$$\begin{cases} V_{id} = \omega v_{id} + c_1 * r_1 (Pbest_{f_i(d)} - x_{id}) + \left(x_{id}^{\max} - x_{id}^{\min}\right) \text{Gaussian}\left(\mu, \sigma^2\right) & \text{when } g_i < \lambda \\ V_{id} = \omega v_{id} + c_i * r_1 (Pbest_{f_i(d)} - x_{id}) & \text{otherwise} \end{cases} \tag{7}$$

where λ is the preset threshold value, $\text{Gaussian}\left(\mu, \sigma^2\right)$ is a random number of a Gaussian distribution function with zero mean and standard deviation.

The acceleration coefficient c_i in Equ. (7) is modified as:

$$c_i = a\left(1 - \frac{g_i}{ST_i}\right) \tag{8}$$

where a is constant, ST_i is defined as :

$$ST_i = \sqrt{\sum_{i=1}^{N} g_i^2} \tag{9}$$

The various steps of the proposed modified CLPSO are as follows:

Step 1 Generate the particles with random positions and velocities within pre-defined ranges, initial all the particles learning probability according to equ. (5)

Step 2 Evaluate all the particles' fitness, update $pbest_1$, $pbest_2$,…, $pbest_N$ and $gbest$, let current generation t=0, the max iteration generation $T_{\max}$=10000.

Step 3 For each particle, do the following sub steps.

1) $w(k) = w_{start}(\frac{w_{start}-w_{end}}{t_{\max}})k$,where, $w_{start} = 0.9$, w_{end} =0.4.

2)if $flag_i \geq$ *refreshing gap*, update velocity and position of particle according to basic PSO algorithm, calculate the $pbest_1$, $pbest_2$,…, $pbest_N$ and *gbest*, let *flagi*=0.

If *flagi*< refreshing gap ,for *j*=1,2,…,*N*, update the velocity and position

$$\begin{aligned} V_i^j &= w_i V_i^j + crand1_i^j \left(pbest_{r_1(j)}^j - X_i^j \right) \\ V_i^j &= \min(V_{\max}^j, \max(V_{\max}^i, V_i^j)) \\ X_i^j &= X_i^j + V_i^j \end{aligned} \tag{10}$$

3)if $x_i^j \in \left[x_{\min}^j, x_{\max}^j \right]$, judge whether update the $pbest_i$ and *gbest*, $flag_i=flag_i+1$.

Step 4 calculates the growth rate g_i and the acceleration coefficient according to (5), (7)and (8).

Step 5 if $g_i < \lambda$, then update the particles' velocity using Equ.(7).

Step 6 $t=t+1$, *if* t< T_{max} , go to step 3 ,else go to step 7.

Step 7 The algorithm stops and returns the gbest.

3. Benchmark Tests and Discussions

A set of benchmark functions are used to evaluate the performance of the proposed M-CLPSO, all functions are tested on 30 dimensions, and the details of these test functions are given in table I. Among this functions, f_1 and f_2 are unimodal function, the rest are multimodal problems.

Simulation is conducted in M-CLPSO algorithm and compared with APSO [11], CLPSO in terms of solution quality, convergence speed and success rate.

To make a fair comparison between different algorithms, thirty independent runs of all algorithms were implemented. The initial population is the same for all the algorithms, the population size is 50. The maximal number of fitness evaluations is set to 10000 and used as the stop criteria for all algorithms on each function, the threshold λ is set to 0.01, a in (7) is set to 2. The experiments are carried out on a PC with AMD Athlon (tm) II X2 250 processor, 3.00 GHz and 3.25GB memory, and Windows XP3 operating system.

3.1 Comparison regarding the solution accuracy

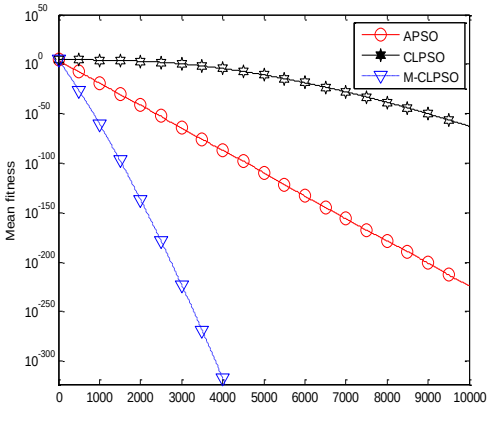
The mean value and the standard deviation (Std. Dev) of the solutions are the key criteria to measure the solution accuracy of a algorithm. The simulation results are given in Table II. The results mark in bold are the best among those obtained by all algorithms. We can observe that the M-CLPSO algorithm surpasses the other two algorithms on all the test functions, and especially significantly improves the results on functions f_1, f_5, and f_7.

3.2. Comparison Regarding the Convergence Speed and Success Rate

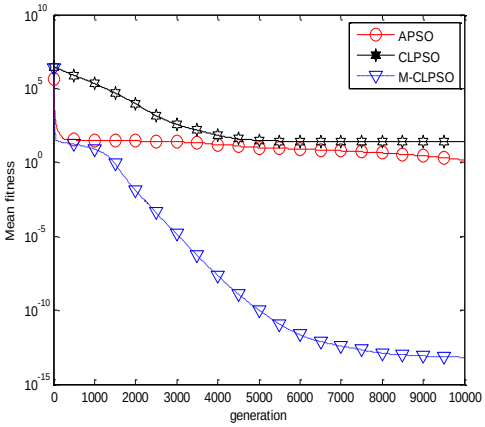
Figure 3 shows the convergence graphs of the different algorithms in testing the benchmark functions. The average number of function evaluations (FEs) to reach acceptable solution and the success rate are listed in Table III. It can be seen from table III that the average numbers of FEs needed to reach an acceptable solution using the M-CLPSO algorithm is smaller than that of other two algorithms. For example, tests on f_3 show that the average numbers of FEs of 1600, 1605, 582 are needed by the APSO, CLPSO and M-CLPSO algorithms, respectively. Table III also reveals that the CLPSO and M-CLPSO reach the acceptable solutions with a success rate of 100% on all the test functions, with that APSO did not converge on function f_7.

Table I. Benchmark Functions Used In Comparison

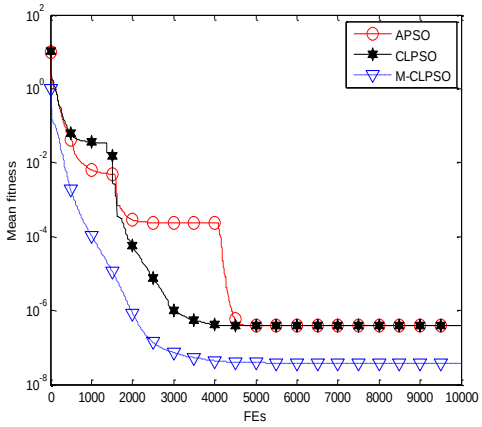
Benchmark function(n=30)	Search space	Acceptance	Global *optimal*
$f_1(x) = \sum_{i=1}^{n} x_i^2$	$-100 \le x_i \le 100$	0.01	0
$f_2(x) = \sum_{i=1}^{n} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$-10 \le x_i \le 10$	100	0
$f_3(x) = 418.9829 * n - x_i \sin\left(\sqrt{x_i}\right)$	$-500 \le x_i \le 500$	0.001	0
$f_4(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$-5.12 \le x_i \le 5.12$	0.01	0
$f_5(x) = \sum_{i=1}^{n} \left[y_i^2 - 10\cos(2\pi y_i) + 10\right]$ where, $y_i = \begin{cases} x_i & ....\lvert x_i\rvert<0.5 \\ round(2x_i)/2 & ....\lvert x_i\rvert\ge 0.5 \end{cases}$	$-5.12 \le x_i \le 5.12$	50	0
$f_5(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{n} x_i}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	$-32 \le x_i \le 32$	0.000001	0
$f_7(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-600 \le x_i \le 600$	0.0001	0
$f_8(x) = \frac{\pi}{n}\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ where, $y_i = 1 + \frac{1}{4}(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	$-50 \le x_i \le 50$	0.001	0



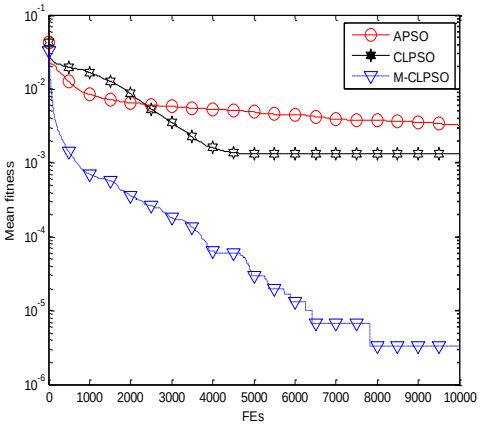
(a) Optimization figure of f_1



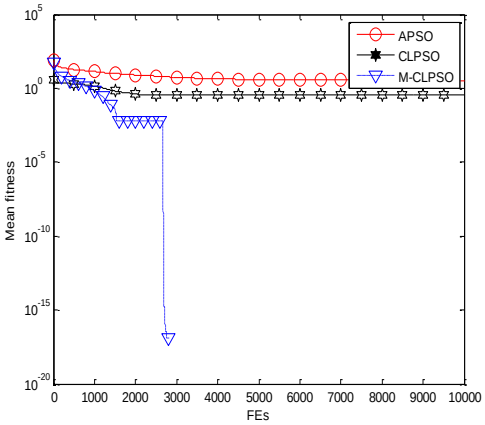
(b) Optimization figure of f_2



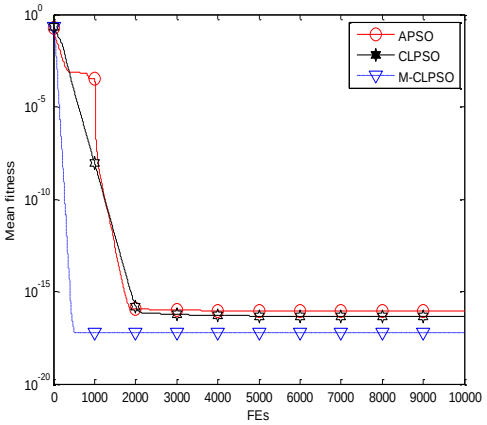
(c) Optimization figure of f_3



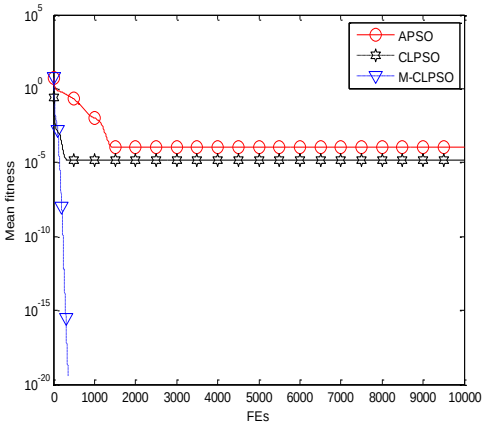
(d) Optimization figure of f_4



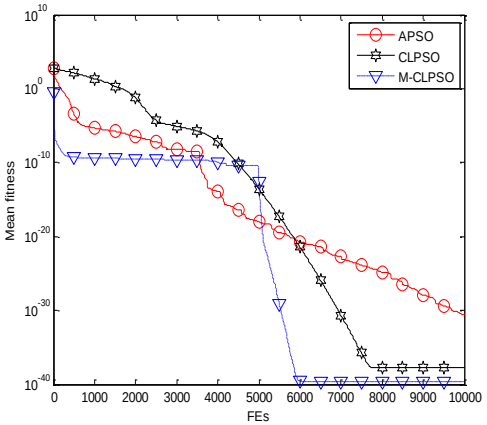
(e) Optimization figure of f_5



(f) Optimization figure of f_6



(g) Optimization figure of f_7



(h) Optimization figure of f_8

Figure 3. Convergence Performance of the Different Algorithm on the Benchmark Functions F_1~F_8

Table II. Mean Value and Standard Deviation Obtained By the Algorithms

Function		APSO	CLPSO	M-CLPSO
f_1	Mean	2.89e-224	2.94e-63	**0**
	Std.Dev	0	2.31e-63	**0**
f_2	Mean	1.436	23.89	**2.43e-10**
	Std.Dev	1.24	1.116	**1.386e-13**

Function		APSO	CLPSO	M-CLPSO
f_3	Mean	3.57e-2	3.77e-2	**4.20e-3**
	Std.Dev	3.12e-1	3.44e-1	**3.27e-2**
f_4	Mean	5.80e-3	4.90e-3	**3.98e-4**
	Std.Dev	3.40e-3	6.40e-3	**1.60e-3**
f_5	Mean	6.48	10.91	**5.48e-1**
	Std.Dev	6.10	10.56	**2.80**
f_6	Mean	2.00e-3	8.67e-4	**3.73e-4**
	Std.Dev	1.52e-2	7.50e-3	**6.10e-3**
f_7	Mean	3.67e-2	1.51e-4	**0**
	Std.Dev	1.94e-1	3.00e-3	**0**
f_8	Mean	6.42e-1	19.36	**4.30e-3**
	Std.Dev	11.17	74.78	**3.79e-1**

Table III. Mean Fes to Reach Acceptable Solutions and Success Rate

Function	APSO	CLPSO	M-CLPSO
f_1	239(100%)	3548(100%)	95(100%)
f_2	135(100%)	3728(100%)	45(100%)
f_3	1600(100%)	1605(100%)	582(100%)
f_4	726(100 %)	1250(100%)	49(100%)
f_5	439(100%)	1760(100%)	113(100%)
f_6	740(100%)	1061(100%)	141(100%)
f_7	--	354(100%)	154(100%)
f_8	548(100%)	2254(100%)	365(100%)

-- denotes that the algorithm cannot converge to the acceptance value

4. MCLPSO for Takagi-Sugeno Fuzzy System Modeling

T-S models have recently become a powerful practical engineering tool for modeling and controlling of complex systems. A T-S fuzzy model consists of several IF-THEN rules, which include the fuzzy antecedents and the mathematical function considered as the rule consequent parts. The task to construct a T-S fuzzy model can be divided into two steps :1) extracting the fuzzy rules; 2) optimizing the parameters of the linear regression models. The main purpose of extracting fuzzy rules is to determine the model's structure, which depends on the clustering algorithm. The latter step can be achieved by optimization techniques, such as the least-squares method (LSM), gradient descent, and genetic algorithms (GA) [13]

4.1. T-S Fuzzy Model

The T-S fuzzy model was used to describe a nonlinear system. It decomposes the input space into several sub-spaces and each one is represented by a simple linear regression model. The typical T-S fuzzy rule is described as follows:

$$\begin{aligned} &R^i: \text{if} \quad x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i \text{ and...and IF } x_n \text{ is } A_n^i \\ &\text{then} \quad y^i = p_0^i + p_1^i x_1 + \cdots + p_n^i x_n \end{aligned} \tag{11}$$

where $i = 1, 2, \cdots c$, $x = [x_1, x_2 \cdots x_n]$ is the input vector, A_j^i is fuzzy set, p_j^i is the structure parameters of the ith rule, y^i is the output of the ith rule.

The membership functions can be described as:

$$A_j^i(x_j) = \exp(-(\frac{x_j - m_j^i}{\sigma_j^i})^2), i = 1, ..., c, j = 1, ...n \tag{12}$$

where m_j^i and σ_j^i are the center and width respectively.

Given a system with the input vector, using defuzzy weighted average method, the output of T-S fuzzy model can be calculated by:

$$\hat{y} = \sum_{i=1}^{c} \mu_i y^i / \sum_{i=1}^{c} \mu_i = \sum_{i=1}^{c} w_i y^i = \sum_{i=1}^{c} w_i (p_0^i + p_1^i x_1 + \cdots + p_n^i x_n) \quad (13)$$

where μ_i is the weight of premise variable and can be defined as

$$u_i = \prod_{j=1}^{c} A_j^i(x_j) \quad (14)$$

4.2. T-S Fuzzy Modeling with M-CLPSO

The modeling of T-S model includes the structure identification and parameter identification. The parameters need to identify include: the center and width of membership functions, the consequent parameters. Coding the center, width and consequent as particle, each particle represents a T-S fuzzy model. The parameter identification problem is transformed into finding the optimized value of fitness value according some criterion.

A single particle in the M-CLPSO algorithm is illustrated in Figure 4.

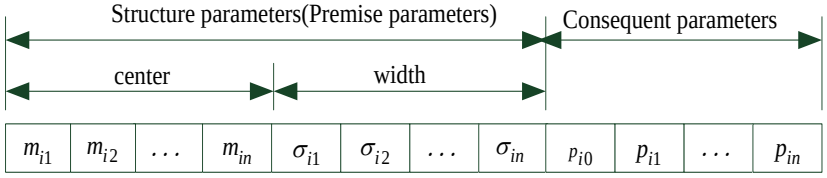


Figure 4. Rule Encoded In a Particle

To measure the performance index in fuzzy modeling, the mean squared error (MSE) between the estimated output and the true value is used as the fitness function, which is defined as:

$$MSE = \frac{1}{L} \sum_{j=1}^{L} (\sum_{i=1}^{n} \mu(p_i^T x + p_{i0}) - y_i)^2 \quad (15)$$

4.3. Experimental Results

Considering the following nonlinear static system with double-input and single-output to be a target system for fuzzy modeling, which is described as [12] .

$$y = (1 + x_1^{-2} + x_2^{-1.5})^2 \quad 1 \le x_1, x_2 \le 5 \quad (16)$$

50 data sets were used to train the M-CLPSO optimized T-S model, the number of fuzzy rules is set to 6, the maximum iteration number *max_gen* = 100. Figure5 illustrate us the iteration process of the M-CLPSO in fuzzy modeling of the static nonlinear system, Figure 6(a) shows the comparison of the outputs of the identified fuzzy model and the real system, and Figure 6(b) shows the error between the real value and the optimized model output. The comparison results between M-CLPSO optimized model and the other models are listed in Table IV.

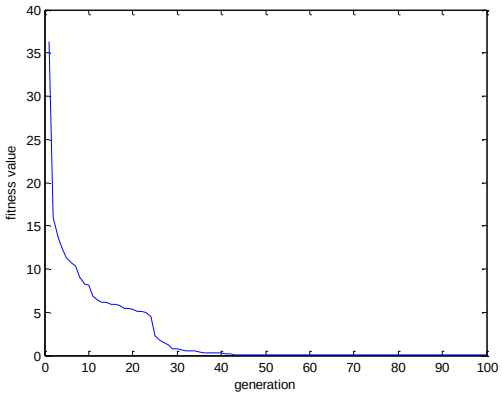
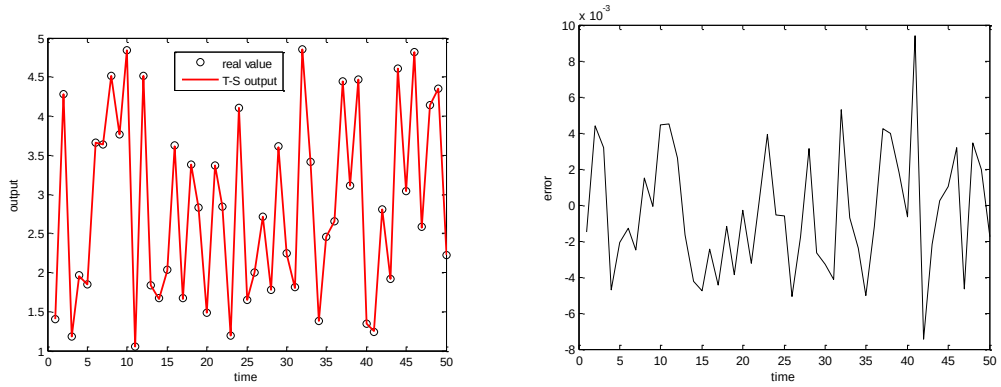


Figure 5. Iteration Process of the M-CLPSO in Fuzzy Modeling of the Static System



(a) Output for the real value and the model (b) The error between the real value and the model

Figure 6. Modeling Result of the Static Nonlinear System

Table IV. Comparison of the Different Methods on the Static Nonlinear System

model	Rule number	MSE
BPSO	6	1.8E-1
Cheung[13]	6	6.4E-3
LI[14]	6	5.0E-3
M-CLPSO	6	2.6E-4

From Figure 5 and Table IV, we can see that the optimized T-S model is the best in terms of the MSE value, which was 2.6E-4, following by LI, Cheung and BPSO**.**

4. Conclusion

A modified CLPSO algorithm names M-CLPSO is proposed in this paper. To increase the convergence speed of CLPSO algorithm, A self-adaptive strategy is adopted to change the value of acceleration coefficient. In the late stages of proposed algorithm, growth rate is defined and used to judge whether the particle is in stagnant. Gaussian disturbance is used to increase the diversity of particles and avoid the stagnant particle of standstill state. Experimental results on multimodal and unimodal benchmark functions demonstrate the superiority of the proposed algorithm to APSO and CLPSO algorithms. The experimental results on T-S

modeling demonstrate that optimized model is able to generate efficient and robust T-F fuzzy model.

Acknowledgments

This work was Supported by the Foundation Project of Key Laboratory of Electrical Control, University of Hunan Province (No. 10K017) and Supported by Educational Commission of Hunan Province of China (Key Project, No. 15A044).

References

[1] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory", Proceeding of the Sixth International Symposium on Micro Machine and Human Science", Nagoya, Japan, **(1995)**.
[2] Kennedy and R. C. Eberhart, "Particle swarm optimization", Proceeding of IEEE International Conference on Neural Network,Perth, Australia, **(1995)**.
[3] S. Liu, L. Xu, D. Li, Q. Li, Y. Jiang, H. Tai and L. Zeng, "Prediction of dissolved oxygen content in river crab culture based on least squares support vector regression optimized by improved particle swarm optimization", Computers and Electronics in Agriculture, vol.95, **(2013)**, pp.82-91.
[4] S. Safari, M. M. Ardehali and M. J. Sirizi, "Particle swarm optimization based fuzzy logic controller for autonomous green power energy system with hydrogen storage", Energy Conversion and Management, vol.65, **(2013)**, pp. 41-49.
[5] K. Sarath and V. Ravi, "Association rule mining using binary particle swarm optimization", Engineering Applications of Artificial Intelligence, vol.26, **(2013)**, pp.1832-1840.
[6] J. J. Liang, A.K. Qin, P.N. Suganthan and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", IEEE Transactions on Evolutionary Computation, vol.10, **(2006)**, pp.281-295.
[7] H. Wu, J. Geng, R. Jin, J. Qiu and W. Liu, "An improved comprehensive learning particle swarm optimization and its application to the semiautomatic design of antennas", IEEE Transactions on Antennas and Propagation, vol.57, **(2009)**, pp**.** 3018-3028.
[8] J. J. Liang and P. N. Suganthan, "Adaptive comprehensive learning particle swarm optimizer with history learning", Simulated Evolution and Learning, Springer Berlin Heidelberg, **(2006)**.
[9] Y. Shi and R. Eberhart, "Modified particle swarm optimizer", Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98), Anchorage, Alaska, USA, **(1998)**.
[10] S. Li, M. Tan, "Tuning SVM parameters by using a hybrid CLPSO-BFGS algorithm", Neurocomputing, vol. 73, **(2010)**, pp. 2089-2096.
[11] Z. H. Zhan, J. Zhang, Y. Li and H. S. Chung, "Adaptive particle swarm optimization", IEEE Transactions On Systems, Man, and Cybernetics, Part B: Cybernetics, vol.39, **(2009)**, pp. 1362-1381.
[12] M. Sugeno and T. Yasukawa, "Afuzzy-logic-based approach to qualitative modeling", IEEE Transaction on Fuzzy Syst., vol. 1, **(1993)**, pp. 7–31.
[13] N. J. Cheung, X. M. Ding and H. B. Shen, "OptiFel: a convergent heterogeneous particle swarm optimization algorithm for Takagi–Sugeno fuzzy modeling", IEEE Transactions on Fuzzy Systems, vol. 22, **(2014)**, pp. 919-933.
[14] C. Li, J. Zhou, B. Fu, P. Kou and J. Xiao, "T–S fuzzy model identification with a gravitational search-based hyperplane clustering algorithm", IEEE Transactions on Fuzzy Systems, vol. 20, **(2012)**, pp.305-317.

Authors

Guohan Lin, he received his B. Sc. and M. Sc. degrees from Hunan University in1996 and 2005 respectively. In 2005, he joined the School of Hunan Institute of Engineering. He is currently working towards the PhD degree in Hunan University. His research interests include evolutionary computation techniques, electric machine drives, power electronics, and intelligent control theory.

Kuiyin Zhao, he received his B. Sc. and M. Sc. degrees from Hunan University of Science and Hunan University in 1996 and 2004 respectively. He is a professor of Hunan Institute of Engineering. His research interests include electric machine drives, power electronics, and intelligent control theory.

