Expert Systems with Applications 38 (2011) 13266–13273

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Adaptive fuzzy approach to function approximation with PSO and RLSE

Chunshien Li *, Tsunghan Wu

Laboratory of Intelligent Systems and Applications, Department of Information Management, National Central University, Taiwan, ROC

ARTICLE INFO

Keywords:
Machine learning
Fuzzy
Particle swarm optimization (PSO)
Recursive least-squares estimator (RLSE)
Function approximation

ABSTRACT

A new adaptive fuzzy approach to function approximation is proposed in the paper. A Takagi–Sugeno (T–S) type fuzzy system is used as the function approximator in the study. The proposed approach uses a hybrid learning method to train the T–S fuzzy system to achieve high accuracy in function approximation. The hybrid learning method combines both the particle swarm optimization (PSO) and the recursive least squares estimator (RLSE) to update the parameters of the fuzzy approximator. The PSO is used to update the premise part of the fuzzy system while the consequent part is updated by the RLSE. The PSO–RLSE learning method is very efficient in learning convergence. The proposed approach is compared to other methods. Three benchmark functions are used for the performance comparison. The proposed approach shows superior performance to compared approaches, in terms of approximation accuracy and learning convergence.

1. Introduction

Function approximation has played an important role in many applications, such as signal processing, pattern recognition, computer vision, communications, controls, and many others. In general, a fuzzy system can be viewed as a function approximator (Rovatti, 1998; Zeng & Singh, 1994). Basically, it has the potential of universal approximation. Based on input–output data, it can provide a solution for the identification of a complex and/or unknown system (Dickerson & Kosko, 1996; Haykin, 1994; Hopfield & Tank, 1986; Kosko, 1992, 1994). In the study, we use a Takagi–Sugeno (T–S) fuzzy system to serve as the function approximator. T–S fuzzy systems are suitable for the identification of nonlinear systems from observed data, for its mapping capability is better than Mamdani fuzzy model in approximation capability. Using an appropriate learning algorithm, the T–S fuzzy approximator can update its parameters so that the performance for function approximation can be improved to desired accuracy. In the paper, we use two well known learning algorithms in a hybrid way, which are the particle swarm optimization (PSO) (Angeline, 1998; Clerc & Kennedy, 2002; Eberhart & Kennedy, 1995; Feng, 2005; Wang, Kang, Qiao, & Wu, 2005) and the recursive least squares estimator (RLSE). PSO is a population-based optimization method and it can find the optimal solution for multi-dimensional non-linear application problems (Ying, 1994). The PSO learning algorithm mimics the search behavior of bird flocking or fish schooling for food. It has been proved that PSO can successfully apply to evolve the parameters of fuzzy model.

In this study, we focus on the proposed adaptive approach to function approximation, which is a central task in numerous applications, such as dynamic system identification, fault detection, classification and optimization (Takagi & Sugeno, 1985; Wang, 1992). For a given set of input–output data, finding the underlying relationship is a fundamental problem in real world applications. Many applications are closely related to the utility of function approximation using fuzzy systems (Hao, Qiao, Y, Liu, & Qiang, 2006). Thus, fuzzy systems can be viewed as rule-based nonlinear function approximator.

An adaptive fuzzy system can uniformly approximate a real continuous function on a compact domain to any degree of accuracy, which is known as the fuzzy approximation theorem (FAT) (Sousa & Kaymak, 2002). The training and optimization for fuzzy system to perform tasks involving function approximation is well documented in literature (Kuncheva, 2000; Mamdani & Assilian, 1999). In this study, we choose the T–S type fuzzy system as the function approximator for the problem of function approximation. To train a T–S fuzzy system, the parameters of the T–S fuzzy system are separated into two subsets, the part of premise parameters and the other part of consequent parameters. The PSO–RLSE hybrid learning method works on the two subsets of parameters that the premise set of parameters is updated first by the PSO and then the consequent set of parameters is updated later by the RLSE.

In Section 2, the methodology of the proposed approach is specified. In Section 3, three examples are used to illustrate the performance of the proposed approach. The proposed approach is compared to other approaches for performance comparison, and the experimental results for function approximation are given. Finally, a discussion is given and the paper is concluded.

* Corresponding author.
E-mail address: jamesli@mgt.ncu.edu.tw (C. Li).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.04.145

2. Methodology of the proposed approach

2.1. Design of fuzzy system as function approximator

T–S fuzzy system is used to be the function approximator in the study. A T–S fuzzy approximator is composed of several T–S fuzzy rules. In general, a fuzzy rule has M inputs and one output, given as follows.

$$\mathrm{R}^i: \text{IF } x_1 \text{ is } A_1^i(h_1) \text{ and } x_2 \text{ is } A_2^i(h_2) \text{ and} \ldots \text{and } x_M \text{is } A_M^i(h_M)$$
$$\text{THEN } Z^i = a_0^i + a_1^i h_1 + \cdots + a_M^i h_M, \quad (1)$$

where x_i is the ith input linguistic variable, h_i is the ith base variable and $A_j^i(\cdot)$ is the jth fuzzy set for the premise of the ith rule. Each input variable has few fuzzy sets. In the study, Gaussian-type membership functions are used. The definition of Gauss membership function is given as follows.

$$\mathrm{gaussmf}(h;\delta,c) = e^{\frac{-(h-c)^2}{2\delta^2}}, \quad (2)$$

where h is a base variable, c is the center and δ is the spread. The T–S fuzzy approximator is assumed to have K rules. Based on fuzzy inference, the input–output relationship for the T–S fuzzy approximator can be expressed as follows.

$$y^* = \frac{\sum_{i=1}^K \beta^i Z^i}{\sum_{i=1}^K \beta^i} = \sum_{i=1}^K \lambda^i \times Z^i = \sum_{i=1}^K \lambda^i \times \left(a_0^i + a_1^i h_1 + \cdots + a_M^i h_M\right), \quad (3)$$

$$\lambda^i = \frac{\beta^i}{\sum_{i=1}^K \beta^i}, \quad (4)$$

where y^* is the T–S fuzzy system output, β^i is the firing strength of the ith rule, and λ^i is the normalized firing strength of the ith rule.

To model a unknown target system (or function), we first have to sample (observe) the input–output behavior of the target system to collect a set of input–output data pairs, which will be used as training data (TD), denoted as follows.

$$\mathrm{TD} = \{(u_i, y_i), i = 1, 2, \ldots, m\}. \quad (5)$$

With the observed input–output data, the proposed T–S fuzzy system can be applied to approximate the unknown target system. The diagram for system modeling or function approximation is shown in Fig. 1.

In Fig. 1, the output of the T–S fuzzy approximator is compared to the corresponding target to produce an error, denoted as follows.

$$e_i = y_i - y_i^*, \quad (6)$$

where $i = 1,2,\ldots,m$. For the set of training data, the concept of root mean square error (RMSE) can be used as the error norm, defined as follows.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^m e_i^2}{m}}. \quad (7)$$

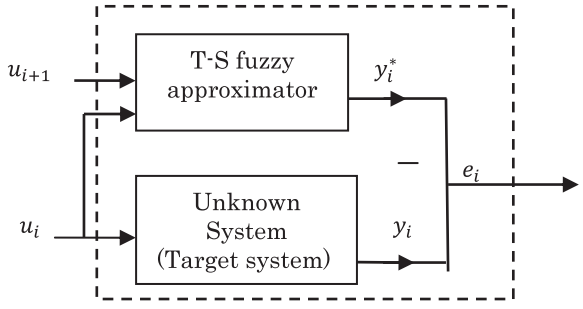

Fig. 1. Diagram of function approximation. The problem of function approximation is viewed as the same problem of system modeling.

2.2. PSO algorithm

The general form of PSO algorithm is given below.

$$\boldsymbol{V}_i(k+1) = \boldsymbol{V}_i(k) + c_1 \times \xi_1 \times (\boldsymbol{pbest}_i(k) - \boldsymbol{L}_i) + c_2 \times \xi_2 \times (\boldsymbol{gbest}(k) - \boldsymbol{L}_i), \quad (8)$$

$$\boldsymbol{L}_i(k+1) = \boldsymbol{L}_i(k) + \boldsymbol{V}_i(k), \quad (9)$$

where k is to indicate the kth iteration, V_i is the velocity of the ith particle, L_i is the location of the ith particle, $\boldsymbol{pbest}_i$ is the best location of the ith particle, $\boldsymbol{gbest}$ is the best location of the swarm, $\{c_1, c_2\}$ are the PSO parameters, and $\{\xi_1, \xi_2\}$ are random numbers uniformly distributed in [0,1]. The swarm size is assumed to be S. Suppose the problem space is with Q dimensions. The particle locations and velocities are expressed as follows.

Locations of particles:

$$\{\boldsymbol{L}_i = (l_{i,1}, l_{i,2}, l_{i,3}, \ldots l_{i,Q}), i = 1, 2, \ldots, S\}.$$

Velocities of particles:

$$\{\boldsymbol{V}_i = (v_{i,1}, v_{i,2}, v_{i,3}, \ldots v_{i,Q}), i = 1, 2, \ldots, S\}.$$

In the PSO algorithm, the velocity of each particle is limited by a given value in magnitude V_{max}. Once the particle velocity is excess of the velocity limit, the particle velocity is set to V_{max}. If the particle velocities are large, the PSO learning is fairly easy to be trapped at a local minimum. If the velocities are small, the PSO converges slowly, but it is relatively hard to fall into a local minimum. Based on the above concepts, we set larger velocity limit at the early training stage to accelerate the convergence speed. As the training process proceeds, the velocity limit gets narrowed gradually. In this study, the initial particle velocities are limited from −10 to 10. If the cost is not improved, a counter is incremented. This learning situation is known as search failure. Once the counting of search failure equals to 300, the velocity limit is updated to $V_{max} = \frac{V_{max}}{10}$. With the concept of RMSE in (7), the fitness values for the particles can be calculated, and the PSO can evolve the fuzzy approximator.

2.3. RLSE algorithm

In this study, the RLSE is used for the identification of the parameters of consequent part (Astrom & Wittenmark, 1994). For a general least-squares problem (Jang, Sun, & Mizutani, 1997), the output of a linear model y, is specified by the linearly parameterized expression, given as follows.

$$y_i = \sum_{i=1}^n \theta_i f_i(u_i), \quad (10)$$

where u_i is the model's input, $f_i(\cdot)$ is known function of u_i, and θ_i, $i = 1,2,\ldots,n$, represent free parameters to be estimated. Here θ_i, $i = 1,2,\ldots,n$, can be viewed as the parameters of the consequent part of the proposed T–S fuzzy approximator. With the training data in (5), substituting each data pair into (10), we can obtain a set of linear equations as follows.

$$f_1(u_1)\theta_1 + f_2(u_1)\theta_2 + \cdots + f_n(u_1)\theta_n = y_1,$$
$$f_1(u_2)\theta_1 + f_2(u_2)\theta_2 + \cdots + f_n(u_2)\theta_n = y_2,$$
$$\vdots$$
$$f_1(u_m)\theta_1 + f_2(u_m)\theta_2 + \cdots + f_n(u_m)\theta_n = y_m.$$

In matrix notation, the least-squares problem can be rewritten as follows.

$$\boldsymbol{A}\boldsymbol{\theta} = \boldsymbol{y}, \tag{11}$$

where $\boldsymbol{A}$ is an $m \times n$ matrix, $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_n]^{\mathrm{T}}$ is an $n \times 1$ parameter vector, $\boldsymbol{y} = [y_1, y_2, \ldots, y_m]^{\mathrm{T}}$ is an output vector. The least-squares estimator (LSE) can be given as follows.

$$\boldsymbol{\theta} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}. \tag{12}$$

For the training of the proposed fuzzy approximator, using data pairs in (5) to update the consequent parameters appropriately, the RLSE can be given as follows.

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_k - \frac{\boldsymbol{P}_k\boldsymbol{b}_{k+1}\boldsymbol{b}_{k+1}^T\boldsymbol{P}_k}{1 + \boldsymbol{b}_{k+1}^T\boldsymbol{P}_k\boldsymbol{b}_{k+1}}, \tag{13}$$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{P}_{k+1}\boldsymbol{b}_{k+1}(y_{k+1} - \boldsymbol{b}_{k+1}^T\boldsymbol{\theta}_k), \tag{14}$$

where $\boldsymbol{P}_k$ is known as the projection operator or gain matrix, $\boldsymbol{\theta}_k$ is the parameter vector at the kth iteration.

To implement the RLSE, we initialize $\boldsymbol{\theta}_0$ as a zero matrix, and $\boldsymbol{P}_0$ identified as follows.

$$\boldsymbol{P}_0 = \alpha\boldsymbol{I}, \tag{15}$$

where α is a large number, $0 < \alpha < \infty$, and $\boldsymbol{I}$ is the identity matrix. In this study, we set $\alpha = 10^9$.

2.4. PSO–RLSE hybrid algorithm

The methods of PSO and RLSE can be combined in a hybrid way to train the fuzzy approximator, as shown in Fig. 2. The PSO is used to update the premise parameters of the T–S fuzzy system. Each location by ***gbest*** of the PSO provides a potential premise solution. Based on the premise parameters, the normalized firing strengths are calculated. After the premise structure is determined, next task is to identify the parameters of the consequent part. The RLSE is used to update the consequent parameters, with the normalized firing strengths. According to (13) and (14), the consequent parameters are estimated, the input vector $\boldsymbol{b}_{k+1}$ and the vector $\boldsymbol{\theta}$ are arranged as follows.

$$\boldsymbol{b}_{k+1} = [\boldsymbol{b}\boldsymbol{b}^1(k+1)\boldsymbol{b}\boldsymbol{b}^2(k+1)\cdots\boldsymbol{b}\boldsymbol{b}^k(k+1)], \tag{16}$$

$$\boldsymbol{b}\boldsymbol{b}^i(k+1) = [\lambda^i h_1(k+1)\lambda^i \cdots h_M(k+1)\lambda^i], \tag{17}$$

$$\boldsymbol{\theta} = [\tau^1 \quad \tau^2 \quad \cdots \quad \tau^K], \tag{18}$$

$$\tau^i = [a_0^i \quad a_1^i \quad \cdots \quad a_M^i], \tag{19}$$

$$i = 1, 2, \ldots, K,$$

$$k = 0, 1, \ldots, m-1.$$

With (3), calculate the output of the approximator. The error between the target and the proposed PSO–RLSE is calculated by (6). The error is RMSE defined in (7), which is used as the cost function in the study. The cost in RMSE is used as the performance index in the learning process for the proposed PSO–RLSE. The PSO–RLSE learning method is described in steps as follows.

Step 1: Generate training data pairs from the target function.
Step 2: Design a fuzzy approximator.
Step 3: Initialize the particle swarm for the premise part of the fuzzy approximator.
Step 4: Input training data pair to the approximator in Fig. 1. Use RLSE to calculate the optimal parameters $\boldsymbol{\theta}$ for the T–S consequent part.
Step 5: Use $\boldsymbol{\theta}$ and training data pairs to calculate the output of the approximator.
Step 6: Use RMSE given in (7) to calculate fitness values of particles. Update the velocities and locations of the particles in the swarm.
Step 7: Go back to Step 4 until any of stopping criteria is satisfied.

3. Experiment for the proposed approach

There are three examples used to test the proposed approach. The proposed PSO–RLSE hybrid learning fuzzy approach is compared to other approaches.

Example 1. The following benchmark function is called the tooth function, and it is given as follows.

$$y = 0.08(1.2(u-1)\cos(3u)) + (u - (u-1)\cos(3u))\sin(u), \tag{20}$$

where $3 \leqslant u \leqslant 7$. With (20), 100 input–output training data pairs are sampled in the given range, denoted as $\{(u_i, y_i),\ i = 1, 2, \ldots, 100\}$. These samples are to be used to train the fuzzy approximator. The tooth function is shown in Fig. 3.

The T–S fuzzy system we used in this example has two inputs x_1 and x_2. With (1) the type of the fuzzy rules for the fuzzy approximator is given as follows.

$$\begin{aligned} \mathrm{R}^i &: \text{IF } x_1 \text{ is } A_1^i(h_1) \text{ and } x_2 \text{ is } A_2^i(h_2), \\ &\text{THEN } Z^i = a_0^i + a_1^i h_1 + a_2^i h_2. \end{aligned} \tag{21}$$

Each input variable has three fuzzy sets with the type of Gaussian membership function in (2).

Fuzzy sets for the first input variable are given as follows.

$$A_{1,1} = \mathrm{gaussmf}(h_1, b_1, b_2),$$
$$A_{1,2} = \mathrm{gaussmf}(h_1, b_3, b_4),$$
$$A_{1,3} = \mathrm{gaussmf}(h_1, b_5, b_6).$$

Fuzzy sets for the second input variable are given as follows.

$$A_{2,1} = \mathrm{gaussmf}(h_2, b_7, b_8),$$
$$A_{2,2} = \mathrm{gaussmf}(h_2, b_9, b_{10}),$$
$$A_{2,3} = \mathrm{gaussmf}(h_2, b_{11}, b_{12}).$$

As a result, the T–S fuzzy approximator has nine rules, given as follows.

$$\mathrm{R}^1 : \mathrm{IF}x_1 \text{ is } A_{1,1}^1(h_1) \text{ and } x_2 \text{ is } A_{2,1}^1(h_2)\mathrm{THEN}\ Z^1 = a_0^1 + a_1^1 h_1 + a_2^1 h_2,$$
$$\mathrm{R}^2 : \mathrm{IF}x_1 \text{ is } A_{1,1}^2(h_1) \text{ and } x_2 \text{ is } A_{2,2}^2(h_2)\mathrm{THEN}\ Z^2 = a_0^2 + a_1^2 h_1 + a_2^2 h_2,$$
$$\vdots$$
$$\mathrm{R}^9 : \mathrm{IF}x_1 \text{ is } A_{1,3}^9(h_1) \text{ and } x_2 \text{ is } A_{2,3}^9(h_2)\mathrm{THEN}\ Z^9 = a_0^9 + a_1^9 h_1 + a_2^9 h_2,$$

In this way, the premise part of the T–S fuzzy approximator has 12 parameters to update, and the consequence part has 27 parameters. Thus, in total there are 39 free parameters in the fuzzy approximator, and they need to be adjusted to develop good approximation capability. For the proposed approach, the PSO algorithm is used to update the parameters of the premise part and the RLSE is used to update the parameters of the consequent part. For each particle in PSO, velocities at initial stage are limited in between-10 to 10. The RMSE in (7) is used as the cost function for the PSO to measure the learning performance. The proposed fuzzy approach using the PSO–RLSE learning method is compared to the same fuzzy approximator using only the PSO learning method. The settings for the PSO and PSO–RLSE are listed in Tables 1 and 2, respectively. The learning curves and the results by PSO and PSO–RLSE are given in Figs. 4–7 for the tooth function. The error between the tooth function and the output of the fuzzy approximate by PSO–RLSE is shown in Fig. 8. The performance comparison is shown in Table 3.

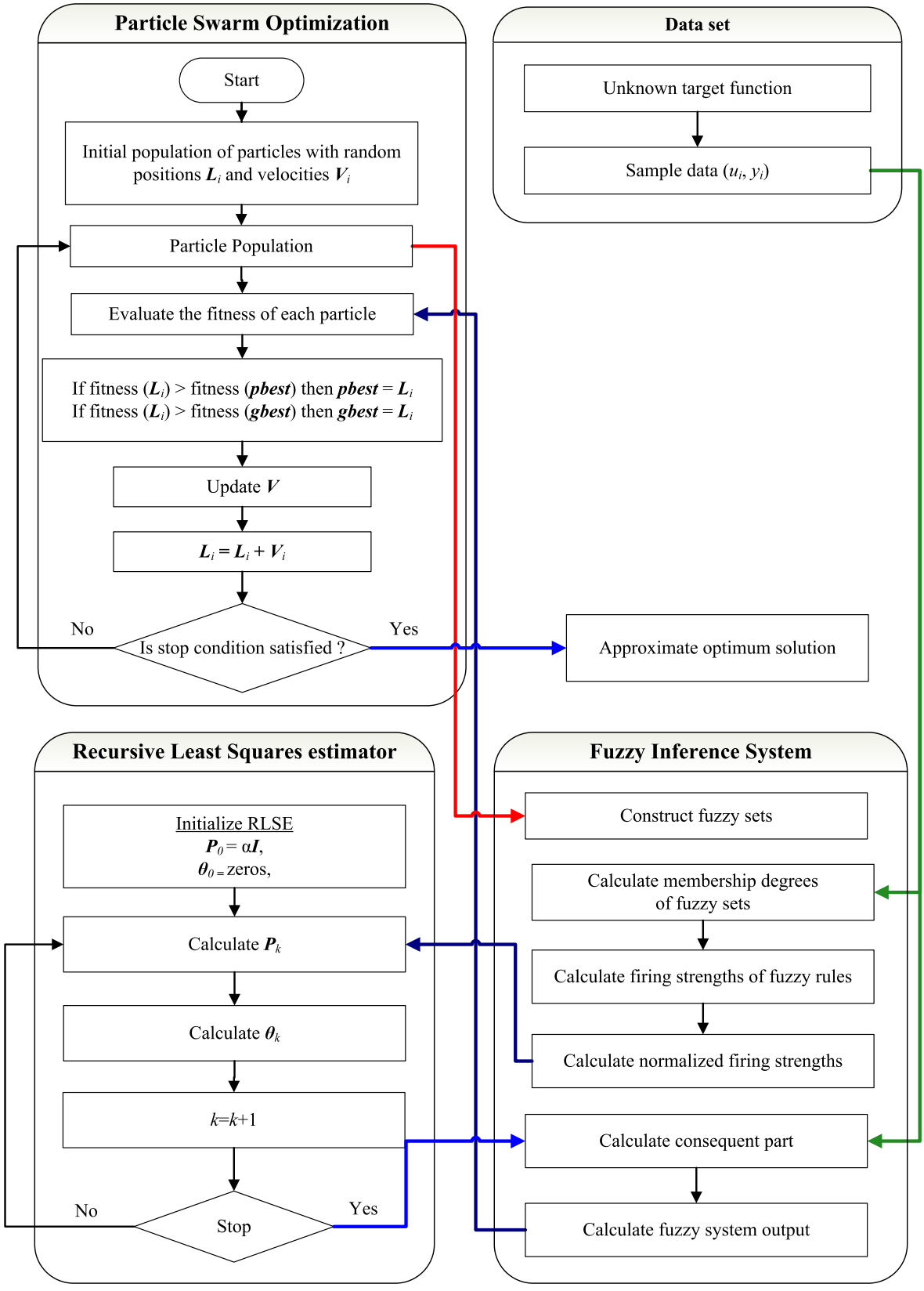

Fig. 2. PSO–RLSE function approximation process.

Example 2. The s-type function (Sun, Tsai, Tsai, Huo, & Liu, 2008) is used in example 2, given as follows.

$$f(u) = \frac{(u-2)(2u-1)}{1+u^2}, \tag{22}$$

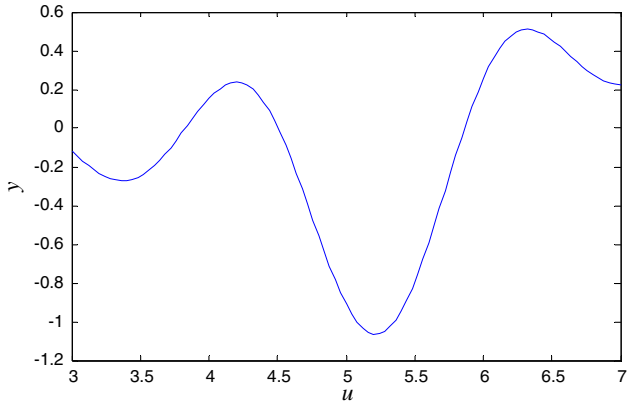

Fig. 3. Tooth function.

Table 1
Settings for the PSO.

PSO	Amount of premise part parameters	12
	Amount of consequent part parameters	27
	Swarm size	633
	Limit for particle velocity	$[-10,10]$

Table 2
Settings for the PSO–RLSE.

PSO	Amount of premise part parameters	12
	Amount of consequent part parameters	27
	Swarm Size	60
	Limit for particle velocity	$[-10,10]$
RLSE	α	10^9
	$\boldsymbol{\theta}_0$	27×1 zero vector
	$\boldsymbol{P}_0$	$\alpha \boldsymbol{I}$
	$\boldsymbol{I}$	27×27 identity matrix

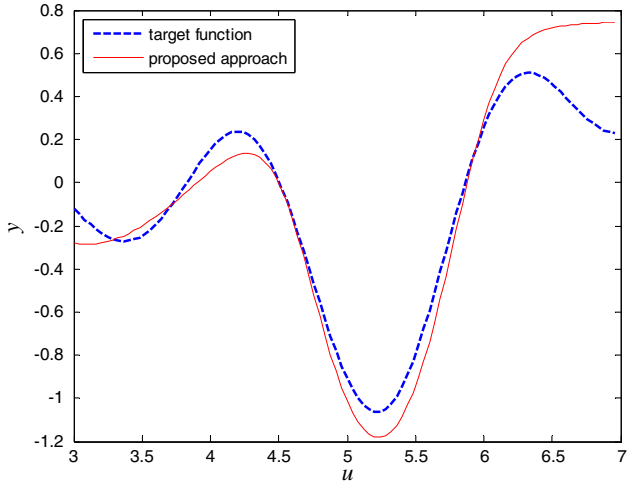

Fig. 4. Function approximation by the PSO learning method for the fuzzy approximator in (21).

where u is randomly selected from $[-8,12]$. The training data are made up of 50 randomly selected samples, and the root mean square error (RMSE) in (7) is used to represent the approximation performance by the proposed approach. The fuzzy approximator is designed the same as that used in Example 1. The hybrid PSO–RLSE learning method is used to evolve the approximator. The settings for the hybrid learning method are given in Table 2. The proposed approach is compared to (Sun et al., 2008). The performance comparison is listed in Table 4. The results by the PSO–RLSE are

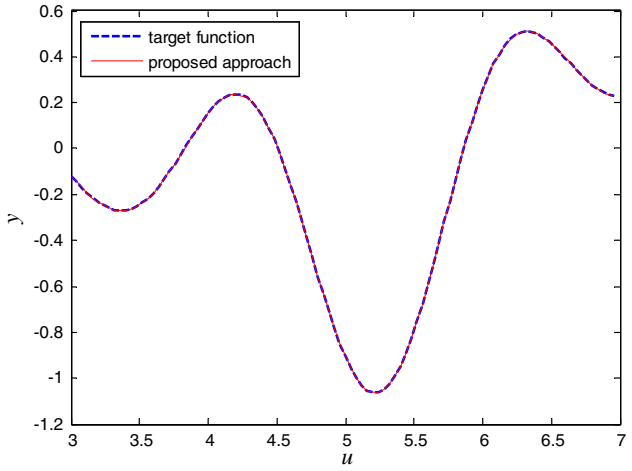

Fig. 5. Function approximation result by the PSO–RLSE learning method for the fuzzy approximator in (21).

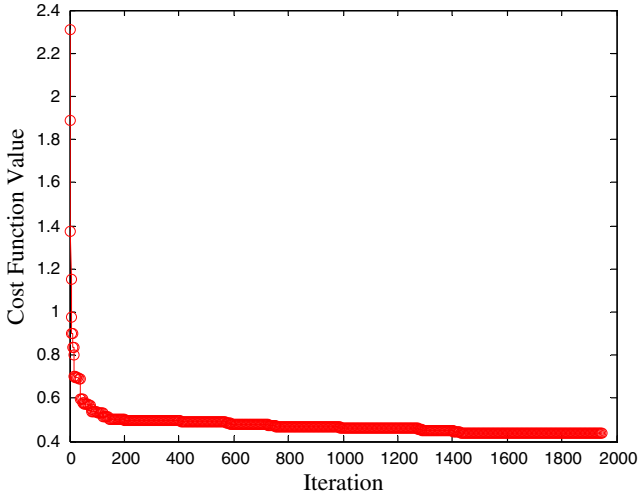

Fig. 6. Learning curve by the PSO method.

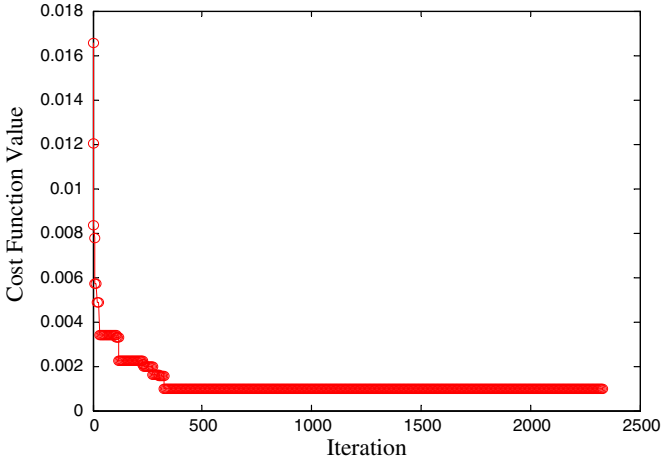

Fig. 7. Learning curve by the PSO–RLSE method.

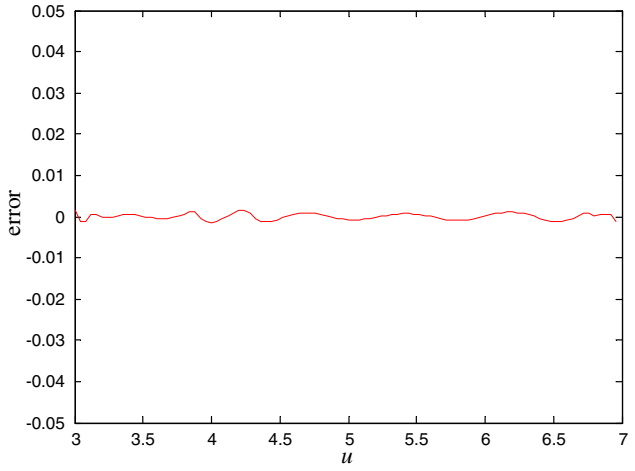

Fig. 8. Function approximation error by the PSO–RLSE approach for the tooth function.

Table 3
Performance comparison for Example 1.

Method	RMSE
PSO	3.0514×10^{-2}
PSO–RLSE	8.92×10^{-4}

Table 4
Performance comparison of the *s*-type function.

Method	RMSE
MLPLS-GBFNFIS (Sun et al., 2008)	8.43×10^{-2}
LW-GRBFNFIS (Sun et al., 2008)	8.7×10^{-3}
PSO–RLSE	5.29×10^{-4}

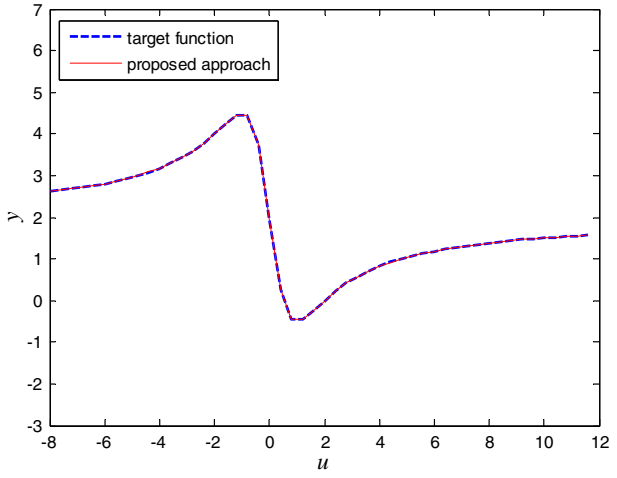

Fig. 9. Function approximation result by the proposed PSO–RLSE approach in Example 2.

shown in Fig. 9, and the approximation error by the proposed approach is shown in Fig. 10.

Example 3. The sine wave function (Bouzerdoum, 2000) is given as follows.

$$f(u) = 0.1 + 1.2u + 2.8\sin(4\pi u^2), \tag{23}$$

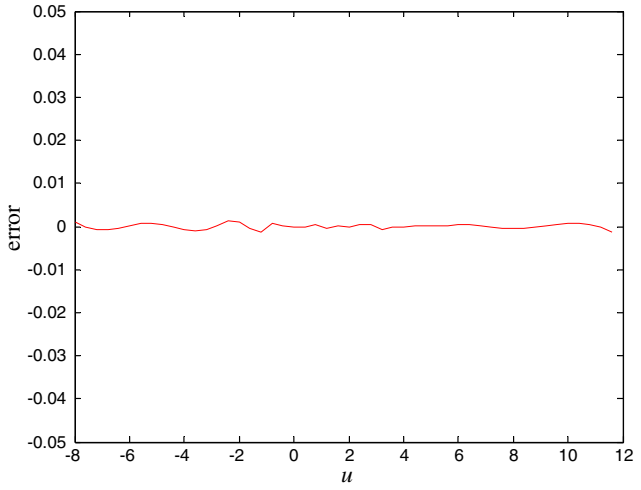

Fig. 10. Function approximation error by the proposed PSO–RLSE approach in Example 2.

Table 5
Performance comparison of the sin wave function.

Method	MSE
MLP (Bouzerdoum, 2000)	6.57228×10^{-3}
SIANN (Bouzerdoum, 2000)	1.76114×10^{-3}
PSO–RLSE (for learning phase)	1.6×10^{-5}
PSO–RLSE (for testing phase)	8.3651×10^{-5}

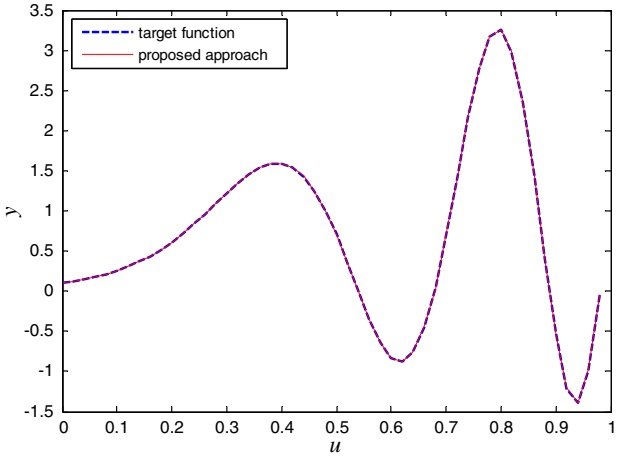

Fig. 11. Function approximation result (for training phase) by the proposed PSO–RLSE approach in Example 3.

where u is randomly selected from (Rovatti, 1998). The fuzzy approximator is designed the same as that used in Example 1. The hybrid PSO–RLSE learning method is used to evolve the approximator. The settings for the hybrid learning method are given in Table 2. The proposed approach is compared to (Bouzerdoum, 2000). The performance comparison is listed in Table 5. 50 random samples are used for training, and 101 random samples are used to test the model. The mean square error (MSE) is used to indicate the performance. The approximation resulted by the proposed PSO–RLSE is shown in Figs. 11 and 12. The approximation error by the proposed approach is shown in Figs. 13 and 14. The parameters after learning are shown in Table 6.

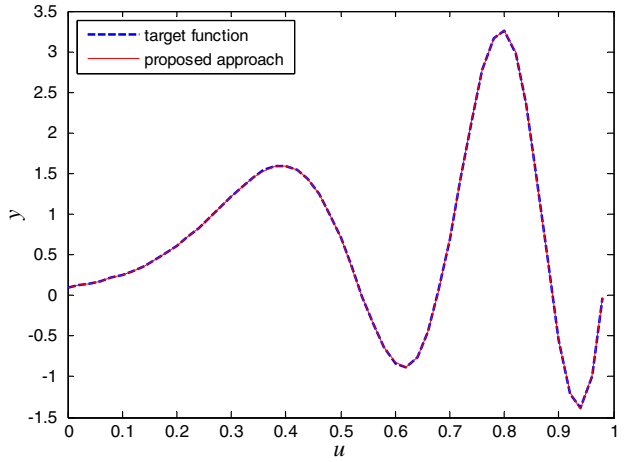

Fig. 12. Function approximation result (for testing phase) by the proposed PSO–RLSE approach in Example 3.

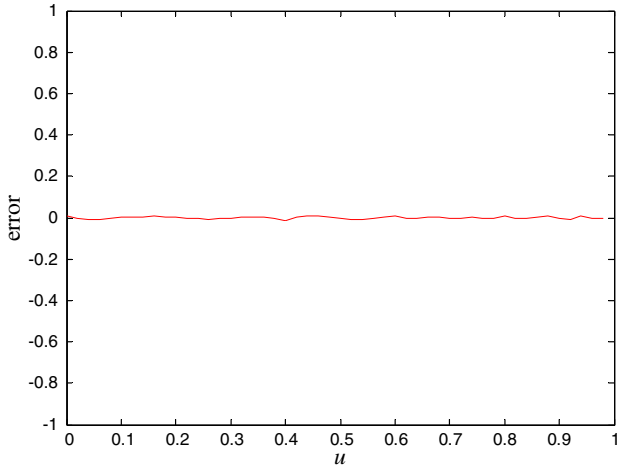

Fig. 13. Function approximation error (for training phase) by the proposed PSO–RLSE approach for the sine wave function in Example 3.

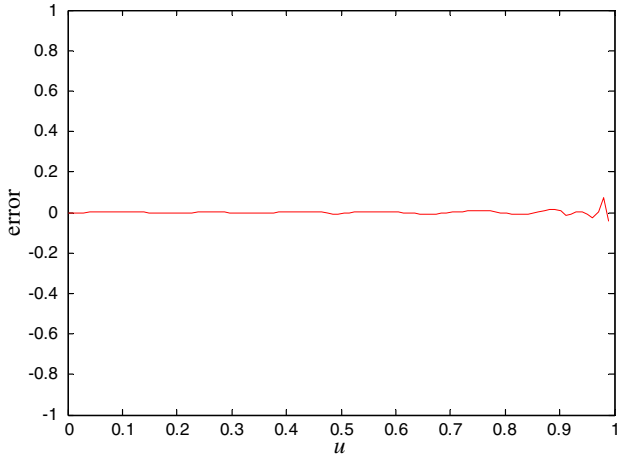

Fig. 14. Function approximation error (for testing phase) by the proposed PSO–RLSE approach for the sine wave function in Example 3.

Table 6
Parameters after learning by the PSO–RLSE.

Fuzzy set	δ	c	δ	c	δ	c
Premise part						
A_1	−0.1783	0.9287	0.6709	−0.0376	0.3995	1.0213
A_2	−0.1179	1.0852	0.5966	0.9759	1.2072	0.8100
Rule	$a_0 + a_1h_1 + a_2h_2$					
	a_0		a_1		a_2	
Consequent part						
R_1	−1000.8827		436.6754		416.6578	
R_2	−311.6719		269.0347		262.8013	
R_3	−337.3004		133.4058		126.6598	
R_4	−62.9539		35.1292		33.8701	
R_5	11.2374		−10.4707		−10.2459	
R_6	−0.8544		4.9819		4.9649	
R_7	−1000.8827		436.6754		416.6578	
R_8	−150.1913		−71.5157		−74.5196	
R_9	87.1036		161.0332		162.7753	

4. Discussion and conclusion

The adaptive fuzzy approach to the problem of function approximation has been presented in the paper. The paper has reported the study of exploring the universal capability of the proposed fuzzy approximator using the PSO–RLSE method for the problem of function approximation. The proposed fuzzy system is designed as the function approximator. Fuzzy systems have the potential of being universal approximator. Theoretically, they can approximate any function to any accuracy. The proposed approach uses the hybrid learning method including both the PSO and the RLSE to update the system parameters of the proposed fuzzy approximator. The system parameters are divided into two subsets, which are called the premise subset and the consequent subset. The premise subset includes the parameters of the fuzzy sets of the premise part of the proposed fuzzy approximator, and the consequent subset collects the parameters of the consequent part of the approximator. In the hybrid learning method, the PSO is responsible to update the premise subset, and the RLSE is used to evolve the consequent subset, as described in Section 2. In the hybrid way, the learning process can converge to the optimal or near-optimal solution swiftly, because the division of the system parameters can efficiently reduce the problem space of the system parameters into two smaller search spaces. And it turns out that the proposed hybrid learning method can be easier to approach to the optimal or near-optimal solution.

To explore the performance of function approximation, we used three examples to scrutinize the proposed adaptive fuzzy approximator. The proposed approach is compared to other approaches for performance comparison. In Example 1 using the tooth function, we compared the proposed PSO–RLSE method to the PSO method alone to train the fuzzy approximator. The performance comparison is shown in Table 3, in which the proposed learning method has the approximation accuracy of RMSE = 8.92e-4, much better than the accuracy of RMSE-3.05e-2 by the compared PSO method. This verifies the thought of the proposed hybrid learning approach. In Examples 2 and 3, the proposed approach is compare to other approaches (Sun et al., 2008; Bouzerdoum, 2000). The performance comparisons are listed in Tables 4 and 5, in which the proposed approach shows very excellent performance in function approximation.

The proposed PSO–RLSE fuzzy approach has been successfully applied to the problem of function approximation, with three examples. From the experimental results and performance comparison, the proposed approach has shown excellent performance in function approximation in terms of learning convergence and approximation accuracy.

References

Angeline, P. J. (1998). Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. *Lecture Notes in Computer Science, 1447*, 601–610.

Astrom, K. J., & Wittenmark, B. (1994). *Adaptive control*. Boston, MA, USA: Addison-Wesley Longman Publishing Co. Inc..

Bouzerdoum, A. (2000). Classification and function approximation using feed-forward shunting inhibitory artificial neural networks. *IEEE International Conference on Neural Networks, 6*, 613–618.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in amultidimensional complex space. *IEEE Transactions on Evolutionary Computation, 6*, 58–73.

Dickerson, J. A., & Kosko, B. (1996). Fuzzy function approximation with ellipsoidal rules. *IEEE Transactions on Systems, Man, and Cybernetics, Part B, 26*, 542–560.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory.

Feng, H. M. (2005). Particle swarm optimization learning fuzzy systems design. In *Proceedings of the third international conference on information technology and applications (ICITA'05)* (Vol. 2, pp. 363–366).

Hao, W.-J, Qiao, Y,-H., Liu, G.-L., & Qiang, W.-Y. (2006). Fuzzy controller design and parameter optimization. In *2006 IEEE international conference on machine learning and cybernetics* (pp. 489–493).

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. New York: Macmillan College Publishing Company Inc..

Hopfield, J. J., & Tank, D. W. (1986). Computing with neural circuits: A model. *Science, 233*, 625–633.

Jang, J. S. R., Sun, C. T., & Mizutani, E. (1997). *Neuro-fuzzy and soft computing*. Prentice Hall Upper Saddle River.

Kosko, B. (1992). Fuzzy systems as universal approximators. *IEEE International Conference on Fuzzy Systems*, 1153–1162.

Kosko, B. (1994). Fuzzy systems as universal approximators. *IEEE Transactions on Computers, 43*, 1329–1333.

Kuncheva, L. I. (2000). *Fuzzy classifier design*. Heidelberg, New York: Physica Verlag.

Mamdani, E. H., & Assilian, S. (1999). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Human–Computer Studies, 51*, 135–147.

Rovatti, R. (1998). Fuzzy piecewise multilinear and piecewise linear systems as universal approximators in Sobolev norms. *IEEE Transactions on Fuzzy Systems, 6*, 235–249.

Sousa, J. M. C., & Kaymak, U. (2002). *Fuzzy decision making in modeling and control*. World Scientific Pub Co. Inc..

Sun, T. Y., Tsai, S. J., Tsai, C. H., Huo, C. L., & Liu, C. C. (2008). Nonlinear function approximation based on least Wilcoxon Takagi–Sugeno fuzzy model. *IEEE International Conference on Intelligent Systems Design and Applications, 1*, 312–317.

Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics, 15*, 116–132.

Wang, L. X. (1992). Fuzzy systems are universal approximators. In *Proceedings of IEEE international conference on fuzzy systems, San Diego, CA* (pp. 1163–1170).

Wang, L., Kang, Q., Qiao, F., & Wu, Q. (2005). Fuzzy logic based multi-optimum programming in particle swarm optimization. In *Proceedings of 2005 IEEE networking, sensing and control* (pp. 473–477).

Ying, H. (1994). Sufficient conditions on general fuzzy systems as function approximators. *Automatica, 30*, pp. 521-521.

Zeng, X. J., & Singh, M. G. (1994). Approximation theory of fuzzy systems-SISO case. *IEEE Transactions on Fuzzy Systems, 2*, 162–176.