Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 735310, 19 pages
http://dx.doi.org/10.1155/2014/735310

Research Article

An Interval Type-2 Fuzzy System with a Species-Based Hybrid Algorithm for Nonlinear System Control Design

Chung-Ta Li,[1] Ching-Hung Lee,[2] Feng-Yu Chang,[1] and Chih-Min Lin[1]

[1] *Department of Electrical Engineering, Yuan Ze University, Taiwan*
[2] *Department of Mechanical Engineering, National Chung Hsing University, Taiwan*

Correspondence should be addressed to Ching-Hung Lee; chleenchu@dragon.nchu.edu.tw

Received 9 December 2013; Accepted 18 March 2014; Published 24 April 2014

Academic Editor: Jianming Zhan

We propose a species-based hybrid of the electromagnetism-like mechanism (EM) and back-propagation algorithms (SEMBP) for an interval type-2 fuzzy neural system with asymmetric membership functions (AIT2FNS) design. The interval type-2 asymmetric fuzzy membership functions (IT2 AFMFs) and the TSK-type consequent part are adopted to implement the network structure in AIT2FNS. In addition, the type reduction procedure is integrated into an adaptive network structure to reduce computational complexity. Hence, the AIT2FNS can enhance the approximation accuracy effectively by using less fuzzy rules. The AIT2FNS is trained by the SEMBP algorithm, which contains the steps of uniform initialization, species determination, local search, total force calculation, movement, and evaluation. It combines the advantages of EM and back-propagation (BP) algorithms to attain a faster convergence and a lower computational complexity. The proposed SEMBP algorithm adopts the uniform method (which evenly scatters solution agents over the feasible solution region) and the species technique to improve the algorithm's ability to find the global optimum. Finally, two illustrative examples of nonlinear systems control are presented to demonstrate the performance and the effectiveness of the proposed AIT2FNS with the SEMBP algorithm.

1. Introduction

In recent years, fuzzy neural networks (FNNs) have been used successfully in many applications [1–20]. For FNNs, the asymmetric fuzzy membership functions (AFMFs) have been discussed and analyzed to improve the approximation accuracies and to effectively reduce the number of fuzzy rules [3, 12, 14, 16, 19, 21]. These AFMFs also provide better performance than traditional fuzzy systems do in the applications of function approximation, modeling, and control. Thus, the learning capability and flexibility can be upgraded. In addition, there has been a growing interest in type-2 fuzzy sets (T2FSs), also known as interval-valued fuzzy sets [1, 2, 10, 11, 20, 22–36]. T2FSs, which are extended from type-1 fuzzy sets, were first proposed by Zadeh [37]. Recently, Mendel and Karnik developed the complete theory of interval type-2 fuzzy logic systems (IT2 FLSs) [28–33, 38]. Many studies have shown that interval-valued fuzzy sets and interval type-2 fuzzy sets (IT2FSs) are the same [12, 28, 29, 32, 37, 39]. The resulting type-2 fuzzy neural network (T2FNN), which adopts IT2FSs as membership functions, provides better performance than those of type-1 do [10–12].

In this paper, we propose an interval type-2 fuzzy neural system with AFMFs, referred to as an AIT2FNS, for nonlinear system controller design. The interval type-2 AFMFs (IT2 AFMFs) and the TSK-type consequent part are used to implement the network structure in AIT2FNS. It is well known that the major computational effort of an IT2FLS system is in type reduction, and the most commonly adopted method to accomplish this is the Karnik-Mendel (KM) procedure [32, 33]. The KM procedure computes the left and right endpoints needed to characterize type-2 fuzzy sets. As described in the literature [2], the type reduction is integrated into the network layers; that is, the KM procedure can be removed. This removal effectively reduces the required computational effort. In this paper, we use this integration technique to construct an AIT2FNS with reduced computational complexity.

For training of the fuzzy neural systems, the back-propagation (BP) algorithm is a powerful and widely used training technique [11, 14, 17, 18, 31]. This method typically cannot find the global solution even if a local minimum is obtained rapidly. The optimization algorithms are less likely to become stuck in a local minimum than are gradient-based learning algorithms (e.g., genetic algorithms, GAs; particle swarm optimizations, PSOs; and electromagnetism-like mechanisms, EMs) [4–6, 8, 9, 15, 17, 38, 40, 41]. A novel hybrid algorithm, the improved electromagnetism-like mechanism using a BP technique (IEMBP), has been proposed to improve the EM algorithm performance [8, 9]. In IEMBP, the random neighborhood local search of the EM algorithm is replaced by a competitive selection and BP technique. However, a statistical analysis should be performed to obtain average performance results, and results are dependent on the population size.

Here, we propose a species-based hybrid algorithm of the EM and BP algorithms (SEMBP) for the proposed AIT2FNS design. Several modifications are introduced to improve performance. One of these modifications is in the initialization step. The initial solution agents are generated using the uniform method [3, 35, 42], and these solution agents are evenly scattered over the feasible solution region. The uniform method does not require statistical analysis, which utilizes repetitive training to obtain average performance data. Additionally, the uniform method which has a lower probability of producing outliers can reduce the computational effort (only one trial should be performed) and can attain better performance. Another modification is the use of a species technique, wherein the population is dynamically divided into subpopulations (or subspecies) based on a similarity measurement, and multiple optima are located. By locating multiple optima, the possibility of finding the global optimum is increased. In these ways, the SEMBP algorithm combines the advantages of the uniform method, the species technique, and the BP local search by species seed. This algorithm has a faster convergence and a lower computational complexity; therefore, it is globally optimized. Finally, we use simulation results of nonlinear system control to illustrate the effectiveness of the AIT2FNS design with the SEMBP algorithm.

The remainder of this paper is organized as follows. Section 2 introduces the proposed AIT2FNS system. Section 3 introduces the hybrid SEMBP algorithm for AIT2FNS. Section 4 describes the simulation results and compares these results with other methods of nonlinear system control. Finally, Section 5 summarizes our conclusions.

2. An Interval Type-2 Fuzzy Neural System with Asymmetric Fuzzy Membership Functions (AIT2FNS)

2.1. Interval Type-2 Asymmetric Fuzzy Membership Functions. The literature indicates that using AFMFs can improve approximation accuracy and can effectively reduce the number of fuzzy rules [3, 12, 14, 16, 19, 21]. It also provides better performance than traditional fuzzy systems do in the applications of function approximation, modeling, and control. The learning capability and flexibility can be upgraded. Here,

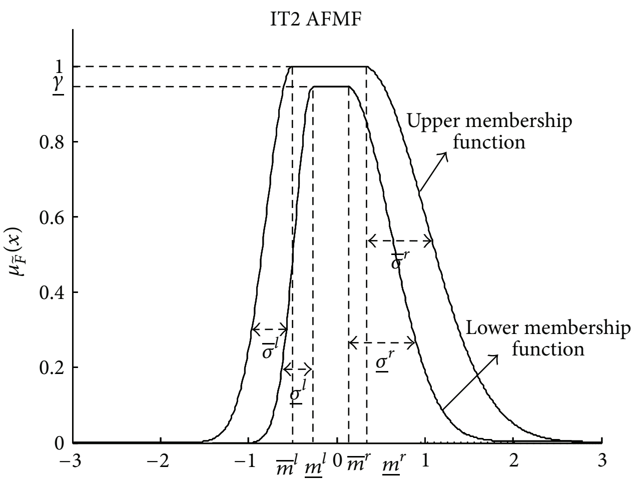


Figure 1: The interval type-2 AFMF [8, 12].

we introduce the IT2AFMFs. According to previous results [12], Gaussian functions are used to construct IT2AFMFs, as shown in Figure 1. Each IT2AFMF is constructed by parts of four Gaussian functions. The upper and lower membership functions (MFs) are constructed using two Gaussian MFs and one segment. The superscripts "l" and "r" denote the left and right curves of MF, respectively. The parameters of the lower and upper MFs are denoted by "_" and "¯," respectively. Accordingly, the upper AFMF is constructed as

$$\overline{\mu}_{\tilde{F}}(x) = \begin{cases} e^{-(1/2)((x-\overline{m}^l)/\overline{\sigma}^l)^2}, & \text{for } x \le \overline{m}^l \\ 1, & \text{for } \overline{m}^l < x < \overline{m}^r \\ e^{-(1/2)((x-\overline{m}^r)/\overline{\sigma}^r)^2}, & \text{for } \overline{m}^r \le x, \end{cases} \quad (1)$$

where $\overline{m}^l$ and $\overline{m}^r$ denote the means of two Gaussian MFs satisfying $\overline{m}^l \le \overline{m}^r$ and $\overline{\sigma}^l$ and $\overline{\sigma}^r$ denote the deviations (width) of the two Gaussian MFs. Similarly, the lower AFMF is given as

$$\underline{\mu}_{\tilde{F}}(x) = \begin{cases} \underline{\gamma} \cdot e^{-(1/2)((x-\underline{m}^l)/\underline{\sigma}^l)^2}, & \text{for } x \le \underline{m}^l \\ \underline{\gamma}, & \text{for } \underline{m}^l < x < \underline{m}^r \\ \underline{\gamma} \cdot e^{-(1/2)((x-\underline{m}^r)/\underline{\sigma}^r)^2}, & \text{for } \underline{m}^r \le x, \end{cases} \quad (2)$$

where $\underline{m}^l \le \underline{m}^r$. To avoid a small activation value, $\underline{\gamma}$ is chosen so that $0.5 \le \underline{\gamma} \le 1$. Thus, the restrictions $\overline{m}^l \le \underline{m}^l \le \underline{m}^r \le \overline{m}^r$, $\underline{\sigma}^l \le \overline{\sigma}^l$, $\underline{\sigma}^r \le \overline{\sigma}^r$, $0.5 \le \underline{\gamma} \le 1$ should be added to avoid unreasonable MFs.

2.2. Fuzzy Reasoning of the Proposed AIT2FNS. Given the system input data x_i $(i = 1, 2, \ldots, n)$ and the fact that the AIT2FNS has M fuzzy rules, the jth rule of the proposed AIT2FNS can be expressed as follows.

Rule j. If x_1 is $\tilde{F}_{1j}$ and $\cdots$ and x_n is $\tilde{F}_{nj}$ then

$$Y_j = C_{j0} + C_{j1}x_1 + C_{j2}x_2 + \cdots + C_{jn}x_n, \quad (3)$$

where $j = 1, 2, \ldots, M$; $\widetilde{F}_{ij}$ is the antecedent fuzzy set that is formed by the IT2AFMFs shown in Figure 1; C_{ij} denotes the consequent fuzzy set; Y_j is the output of jth rule. As described above, the antecedent part $\mu_{\widetilde{F}_{ij}}$ is an interval set; that is,

$$\mu_{\widetilde{F}_{ij}}\left(x_{ij}\right) = \left[\underline{\mu}_{\widetilde{F}_{ij}}\left(x_{ij}\right), \overline{\mu}_{\widetilde{F}_{ij}}\left(x_{ij}\right)\right]. \tag{4}$$

Then, the firing set of jth rule computed using the product t-norm is

$$F_j(\mathbf{x}) = \left[\underline{f}_j(\mathbf{x}), \overline{f}_j(\mathbf{x})\right], \tag{5}$$

where $\underline{f}_j(\mathbf{x}) = \underline{\mu}_{\widetilde{F}_{1j}}(x_{1j}) * \cdots * \underline{\mu}_{\widetilde{F}_{nj}}(x_{nj})$ and $\overline{f}_j(\mathbf{x}) = \overline{\mu}_{\widetilde{F}_{1j}}(x_{1j}) * \cdots * \overline{\mu}_{\widetilde{F}_{nj}}(x_{nj})$. The consequent fuzzy set is also an interval set; that is,

$$C_{ji} = \left[c_{ji} - s_{ji}, c_{ji} + s_{ji}\right], \tag{6}$$

where c_{ji} denotes the center of C_{ji}, and s_{ji} denotes the spread of C_{ji}. Hence, the consequent part of Rule j is also an interval set; that is, $Y^j = [y_j^l, y_j^r]$, where

$$\begin{aligned} y_j^l &= \left(c_{j0} + \sum_{i=1}^{n} c_{ji} x_{ij}\right) - \left(s_{j0} + \sum_{i=1}^{n} s_{ji}\left|x_{ij}\right|\right), \\ y_j^r &= \left(c_{j0} + \sum_{i=1}^{n} c_{ji} x_{ij}\right) + \left(s_{j0} + \sum_{i=1}^{n} s_{ji}\left|x_{ij}\right|\right). \end{aligned} \tag{7}$$

Hence, the output of the fuzzy logic system can be obtained using the extension principle and is given as

$$\begin{aligned} &Y_{\mathrm{TSK}}(u) \\ &\quad = \left[y^l, y^r\right] \\ &\quad = \frac{\int_{y_1 \in [y_1^l\ y_1^r]} \cdots \int_{y_M \in [y_M^l\ y_M^r]} \int_{f_1 \in [\underline{f}_1\ \overline{f}_1]} \cdots \int_{f_M \in [\underline{f}_M\ \overline{f}_M]} 1}{\sum_{j=1}^{M} f_j y_j / \sum_{j=1}^{M} f_j}. \end{aligned} \tag{8}$$

Herein, we need to compute the left-end point y^l and right-end point y^r by utilizing the KM algorithm [32, 33]. In the KM algorithm, the left-end and right-end points are represented as

$$\begin{aligned} y^l &= \frac{\sum_{i=1}^{L} \overline{f}_i y_i + \sum_{L+1}^{M} \underline{f}_i y_i}{\sum_{i=1}^{L} \overline{f}_i + \sum_{L+1}^{M} \underline{f}_i}, \\ y^r &= \frac{\sum_{i=1}^{R} \underline{f}_i y_i + \sum_{R+1}^{M} \overline{f}_i y_i}{\sum_{i=1}^{R} \underline{f}_i + \sum_{R+1}^{M} \overline{f}_i}. \end{aligned} \tag{9}$$

Then, we have to find the proper switch point values L and R by iterative procedure, where more details can be referred to [32, 33]. However, this iterative procedure for finding switch point values is time-wasted. In the proposed AIT2FNS, a simple weight-average method is used to replace the iterative procedure for finding L and R. Hence, we define the left-most and right-most points of firing strength by

$$f_j^l = \frac{\overline{\omega}_j^l \overline{f}_j + \underline{\omega}_j^l \underline{f}_j}{\overline{\omega}_j^l + \underline{\omega}_j^l}, \qquad f_j^r = \frac{\overline{\omega}_j^r \overline{f}_j + \underline{\omega}_j^r \underline{f}_j}{\overline{\omega}_j^r + \underline{\omega}_j^r}, \tag{10}$$

where $\overline{\omega}_j^l$, $\underline{\omega}_j^l$, $\overline{\omega}_j^r$, and $\underline{\omega}_j^r$ are adjustable weights. In (10), the uncertainty of the antecedent is reduced such that the requirement of the type-reduction is satisfied. In the sequel, we obtain the left-end point and right-end point of the output of interval type-2 fuzzy inference system as follows:

$$y^l = \frac{\sum_{j=1}^{M} f_j^l y_j^l}{\sum_{j=1}^{M} f_j^l}, \qquad y^r = \frac{\sum_{j=1}^{M} f_j^r y_j^r}{\sum_{j=1}^{M} f_j^r}. \tag{11}$$

Finally, the defuzzified output of AIT2FNS is

$$y(x) = \frac{y^l + y^r}{2}. \tag{12}$$

As above description, the iterative KM algorithm is replaced by several direct node operations such that the computational complexity of the proposed method is intuitively lower. Therefore, we analyze the computational cost between KM algorithm and left-most and right-most layers. For a two-input-single-output interval type-2 fuzzy system, we accumulate the used times of the addition and the multiplication operations in each method. The comparison results are shown in Table 1. These results are performed using Matlab running on a computer with Intel i5-661 3.33 GHz and 3.24 GB of main memory. We can observe that the proposed simplified network structure does reduce the computational effort on computation time and operations.

2.3. Network Structure of the AIT2FNS.

Herein, we introduce the network structure of AIT2FNS. The multi-input-single-output case is considered here for convenience. An AIT2FNS with M fuzzy rules is implemented as the six-layer network shown in Figure 2. It can be noted that the operation of layers 4 and 5 is normalization, which will be introduced as follows. The signal propagation and operation functions of the nodes are indicated in each layer. In the following description, $O_i^{(k)}$ denotes the ith output of a node in the kth layer.

Layer 1: Input Layer. For the ith node of layer 1, the net input and output are represented as

$$O_i^{(1)} = x_i, \quad i = 1, \ldots, n, \tag{13}$$

Table 1: Computation results between the KM algorithm and left-most and right-most layer.

Number of rules	Operation	Computational effort	
		KM algorithm	Left-most and right-most layer
10	Addition	162.4	60
	Multiplication	81.9	50
	Computational time	0.000208 sec.	0.000120 sec.
20	Addition	325.6	120
	Multiplication	163.3	100
	Computational time	0.000278 sec.	0.000136 sec.
30	Addition	534.2	180
	Multiplication	267.6	150
	Computational time	0.000331 sec.	0.000171 sec.
40	Addition	715.8	240
	Multiplication	358.4	200
	Computational time	0.000493 sec.	0.000228 sec.

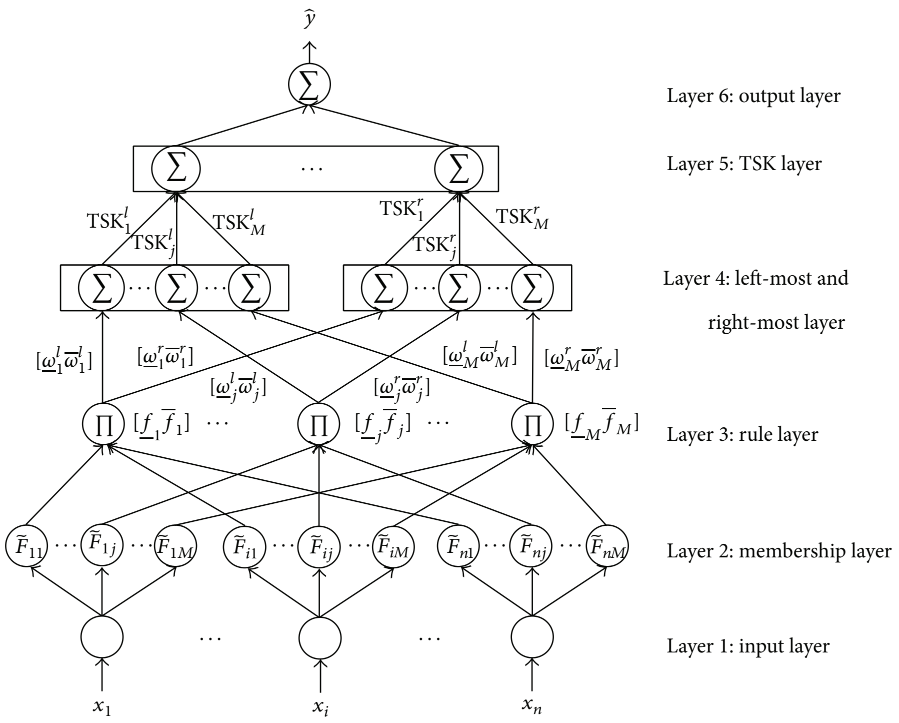

$\hat{y}$
Layer 6: output layer
Layer 5: TSK layer
TSK_1^l
TSK_j^l
TSK_M^l
TSK_1^r
TSK_j^r
TSK_M^r
Layer 4: left-most and right-most layer
$[\underline{\omega}_1^l \overline{\omega}_1^l]$
$[\underline{\omega}_1^r \overline{\omega}_1^r]$
$[\underline{\omega}_j^l \overline{\omega}_j^l]$
$[\underline{\omega}_j^r \overline{\omega}_j^r]$
$[\underline{\omega}_M^l \overline{\omega}_M^l]$
$[\underline{\omega}_M^r \overline{\omega}_M^r]$
$[\underline{f}_1 \overline{f}_1]$
$[\underline{f}_j \overline{f}_j]$
$[\underline{f}_M \overline{f}_M]$
Layer 3: rule layer
$\tilde{F}_{11}$ $\tilde{F}_{1j}$ $\tilde{F}_{1M}$ $\tilde{F}_{i1}$ $\tilde{F}_{ij}$ $\tilde{F}_{iM}$ $\tilde{F}_{n1}$ $\tilde{F}_{nj}$ $\tilde{F}_{nM}$
Layer 2: membership layer
Layer 1: input layer
x_1 x_i x_n

Figure 2: Network structure of the proposed AIT2FNS with M rules.

where x_i represents the ith input to the ith node of layer 1. The nodes in this layer only transmit input values to the next layer directly.

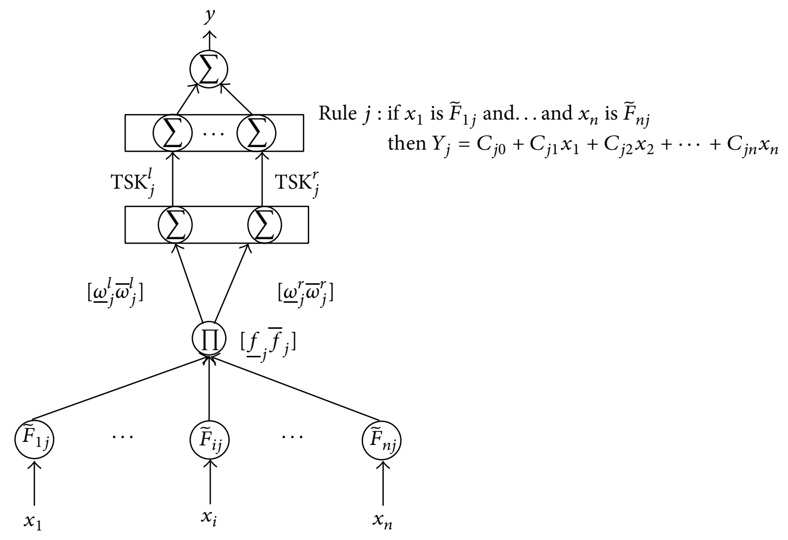
Layer 2: Membership Layer. In this layer, each node performs an IT2 AFMF:

$$\begin{aligned} O_{ij}^{(2)} &= \mu_{\tilde{F}_{ij}}\left(O_i^{(1)}\right) = \left[\underline{O}_{ij}^{(2)}, \overline{O}_{ij}^{(2)}\right] \\ &= \left[\underline{\mu}_{\tilde{F}_{ij}}\left(O_i^{(1)}\right), \overline{\mu}_{\tilde{F}_{ij}}\left(O_i^{(1)}\right)\right], \qquad (14) \\ & i = 1, \dots, n, \quad j = 1, \dots, M, \end{aligned}$$

where the subscript "ij" indicates the jth term of the ith input.

Layer 3: Rule Layer. The links in this layer are used to implement the antecedent matching. Using the product t-norm, the firing strength associated with the jth rule is

$$\begin{aligned} \underline{f}_j &= \underline{\mu}_{\tilde{F}_{1j}}\left(O_1^{(1)}\right) * \cdots * \underline{\mu}_{\tilde{F}_{nj}}\left(O_n^{(1)}\right), \\ \overline{f}_j &= \overline{\mu}_{\tilde{F}_{1j}}\left(O_1^{(1)}\right) * \cdots * \overline{\mu}_{\tilde{F}_{nj}}\left(O_n^{(1)}\right), \end{aligned} \qquad (15)$$



y
$\sum$
Rule j : if x_1 is $\tilde{F}_{1j}$ and. . . and x_n is $\tilde{F}_{nj}$
then $Y_j = C_{j0} + C_{j1}x_1 + C_{j2}x_2 + \cdots + C_{jn}x_n$
TSK_j^l
TSK_j^r
$[\underline{\omega}_j^l \overline{\omega}_j^l]$
$[\underline{\omega}_j^r \overline{\omega}_j^r]$
$\prod$
$[\underline{f}_j \overline{f}_j]$
$\tilde{F}_{1j}$
$\tilde{F}_{ij}$
$\tilde{F}_{nj}$
x_1
x_i
x_n

FIGURE 3: The connection based on the jth rule construction of the proposed AIT2FNS.

where $\underline{\mu}_{\tilde{F}_{ij}}(\cdot)$ and $\overline{\mu}_{\tilde{F}_{ij}}(\cdot)$ are the lower and upper membership grades of $\mu_{\tilde{F}}(\cdot)$, respectively. Therefore, a simple product operation is used. Accordingly,

$$O_j^{(3)} = \left[\underline{O}_j^{(3)}, \overline{O}_j^{(3)}\right] = \left[\prod_{i=1}^{n} \underline{O}_{ij}^{(2)}, \prod_{i=1}^{n} \overline{O}_{ij}^{(2)}\right]. \quad (16)$$

Layer 4: Left-Most and Right-Most Layer. This layer evaluates the left-most and right-most firing points. Therefore, the output of layer 4 is

$$O_j^{(4)} = \left[O_{jl}^{(4)}, O_{jr}^{(4)}\right] = \left[\frac{\overline{\omega}_j^l \overline{O}_j^{(3)} + \underline{\omega}_j^l \underline{O}_j^{(3)}}{\overline{\omega}_j^l + \underline{\omega}_j^l}, \frac{\overline{\omega}_j^r \overline{O}_j^{(3)} + \underline{\omega}_j^r \underline{O}_j^{(3)}}{\overline{\omega}_j^r + \underline{\omega}_j^r}\right], \quad (17)$$

where $O_{jl}^{(4)}$ and $O_{jr}^{(4)}$ denote the left-most and right-most points of firing strength, respectively; $\underline{\omega}_j^l$, $\overline{\omega}_j^l$, $\underline{\omega}_j^r$, and $\overline{\omega}_j^r$ are adjustable weights. According to the results presented in [2], the type reduction is integrated into the corresponding network layers. A simple weight-averaging method is used to allow the adaptive algorithm to evaluate the left-most and right-most firing points. Therefore, the traditional KM procedure, which is an iterative procedure for finding the right-most and left-most points of the centroid, is removed. Comparing with the use of a traditional KM algorithm for type reduction, there is a significant reduction in computational cost.

Layer 5: TSK Layer. Because the IT2 FSs are used for the antecedents and the interval type-1 fuzzy sets are used for the consequent sets of the type-2 TSK rules, it is possible to state that the C_{ji} terms are interval sets. In other words, $C_{ji} = [c_{ji} - s_{ji}, c_{ji} + s_{ji}]$, where $i = 1, \ldots, n$ and $j = 1, \ldots, M$. In this expression, c_{ji} denotes the center of C_{ji} and s_{ji} denotes the spread of C_{ji}, where $s_{ji} \geq 0$. Therefore, the consequent of rule j is

$$\begin{aligned} \mathrm{TSK}_j &= \left[\mathrm{TSK}_j^l, \mathrm{TSK}_j^r\right] \\ &= \left[\left(c_{j0} + \sum_{i=1}^{n} c_{ji}x_i\right) - \left(s_{j0} + \sum_{i=1}^{n} s_{ji}\left|x_i\right|\right),\right. \\ &\qquad \left.\left(c_{j0} + \sum_{i=1}^{n} c_{ji}x_i\right) + \left(s_{j0} + \sum_{i=1}^{n} s_{ji}\left|x_i\right|\right)\right]. \end{aligned} \quad (18)$$

Accordingly, the output of layer 5 is

$$O^{(5)} = \left[O_l^{(5)}, O_r^{(5)}\right] = \left[\frac{\sum_{j=1}^{M} O_{jl}^{(4)} \mathrm{TSK}_j^l}{\sum_{j=1}^{M} O_{jl}^{(4)}}, \frac{\sum_{j=1}^{M} O_{jr}^{(4)} \mathrm{TSK}_j^r}{\sum_{j=1}^{M} O_{jr}^{(4)}}\right]. \quad (19)$$

Layer 6: Output Layer. Layer 6 is used to implement the defuzzification operation. The output is

$$O^{(6)} = \frac{O_l^{(5)} + O_r^{(5)}}{2}. \quad (20)$$

Based on the above interval type-2 fuzzy inference system, a connection structure based on the jth fuzzy rule can be illustrated as in Figure 3, where $\sum$ denotes a normalization-like operation used in Layers 4 and 5 (see (17) and (19)), and TSK_j is introduced in (18).

3. Species-Based Hybrid Algorithm SEMBP for Training AIT2FNS

This section introduces the proposed species-based hybrid algorithm, SEMBP, for training of the AIT2FNS on nonlinear control applications. The SEMBP algorithm combines

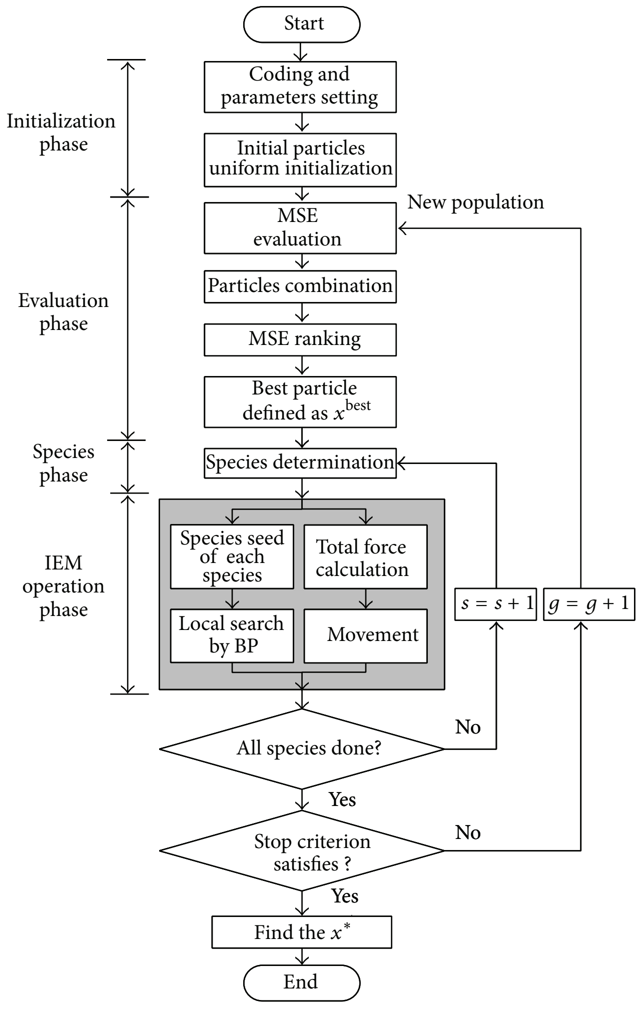


FIGURE 4: Description of the proposed SEMBP algorithm.

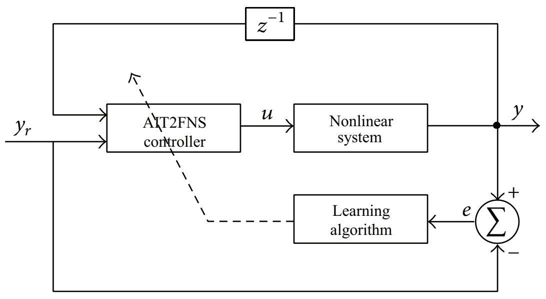


FIGURE 5: The SEMBP-based AIT2FNS control scheme for a nonlinear system.

the advantages of the EM and BP algorithms with uniform initialization, the species technique, and the BP local search by species seed. This combination results in high-speed convergence, lower computational complexity, and global optimization. Figure 4 shows the flow chart of the SEMBP algorithm. There are four phases in the SEMBP algorithm: "initialization," "evaluation," "species," and "IEM operation." Additionally, the modifications of the EM algorithm are the use of the uniform initialization method, the operation of the charge movement of each species, the local search using the best particle of each species, and the removal of the redundant particles.

The SEMBP for optimization problem is in the form of

$$\begin{aligned} &\text{Minimize } f(x) \\ &\text{subject to } x \in S, \quad S = \{x \in \mathfrak{R}^n \mid l_k \le x_k \le u_k, l_k, u_k \in \mathfrak{R}, \\ &\qquad\qquad k = 1, \dots, n\}, \end{aligned} \tag{21}$$

where u_k and l_k are the corresponding upper and lower bounds and $f(x)$ is the function that is being minimized. Each particle x represents a solution where the charge depends on the fitness function $f(x)$.

When considering the nonlinear control problem, our goal is to generate a proper control signal $u(k)$ such that the system output $y(k)$ follows the desired trajectory $y_r(k)$, where k is the discrete time index. The AIT2FNS with the SEMBP algorithm plays the role of controller for a nonlinear plant, and the SEMBP-based AIT2FNS control scheme is shown in Figure 5. The tracking error is $e(k) = y_r(k) - y(k)$. Next, we define the objective function as

$$E(\cdot) = \frac{1}{2} \sum_k e(k)^2. \tag{22}$$

Our goal is to generate the control signal such that the tracking error approaches zero, that is, to minimize the objective function $E(\cdot)$. Thus, the fitness function can be chosen as the mean-square-error of the tracking error; that is, MSE: $\sum_{k=1}^{T} e^2(k)/T$, where T is the data number.

3.1. Initialization Phase. By using the SEMBP algorithm for training the AIT2FNS, each particle denotes a weighting vector with dimension D (Figure 6). Typically, the initial particles are chosen randomly from the feasible solution region. When using random initialization, the particles may be crowded in a region or the particle diversity may be lost. Therefore, a statistical analysis with repetitive training should be performed. To overcome this requirement, we adopted the uniform initialization method [3, 42, 43]. The initial particles are "uniformly distributed"; that is, all initial particles can be evenly distributed within the high-dimensional region. Furthermore, limited data can be used to achieve credible results, and the required simulation time is greatly reduced. Here, we used the good lattice point method, which is one of the uniform methods for constructing the uniform arrays that are used in the initialization phase. Using the uniform method, a statistical analysis is not necessary, and only one trial should be performed. This reduction in trials reduces the computational effort required for performance analysis. In addition, the uniform method has a lower probability of producing outliers that may deeply affect the results. Here, the good lattice point method of uniform initialization provides

$\overline{m}^l_{ij}$	$\cdots$	$\underline{m}^l_{ij}$	$\cdots$	$\overline{m}^r_{ij}$	$\cdots$	$\underline{m}^r_{ij}$	$\cdots$	$\overline{\sigma}^l_{ij}$	$\cdots$	$\underline{\sigma}^l_{ij}$	$\cdots$	$\overline{\sigma}^r_{ij}$	$\cdots$	$\underline{\sigma}^r_{ij}$	$\cdots$	$\underline{\gamma}_{ij}$	$\cdots$	$\overline{\omega}_{ij}$	$\cdots$	$\underline{\omega}_{ij}$	$\cdots$	c_{ij}	$\cdots$	s_{ij}	$\cdots$

FIGURE 6: Particle representation of the AIT2FNS.

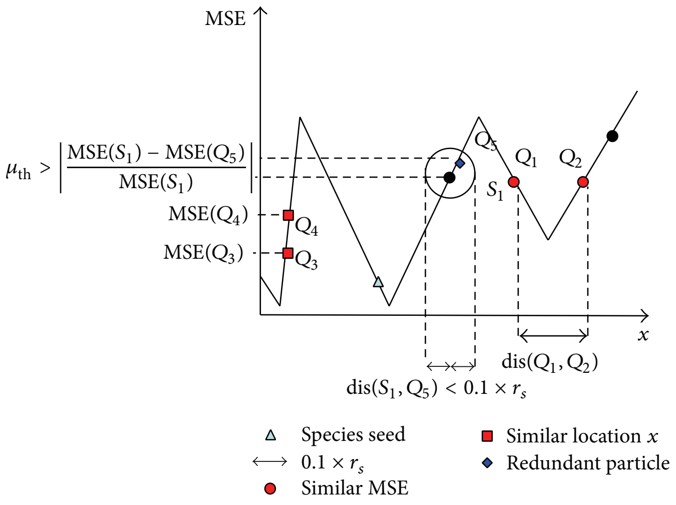


FIGURE 7: Illustrated example of removing redundant particles by similarity measurement in one dimension problem.

a series of uniform arrays, $U_N(q^m)$, for different values of m and q [42, 44]. Based on $U_N(q^m)$, N particles can be generated such that

$$x_i = \left[x_{l1} + \frac{2u_{i1} - 1}{2q}(x_{u1} - x_{l1}) \quad \cdots \quad x_{lm} + \frac{2u_{im} - 1}{2q}(x_{um} - x_{lm}) \right], \tag{23}$$

where x_i represents the particles, u_{ik} is the element of $U_N(q^m)$, $i = 1, \ldots, N$, and $k = 1, \ldots, m$. One can choose $u_k \in U$ such that the lattice points are uniformly scattered in the feasible solution region [42, 44].

As suggested by the literature [9], the population size (denoted as PS) can be chosen as the half of problem dimension; that is, PS $= D/2$. However, because of the restrictions imposed by a uniform array, the population size is chosen to be approximately $D+1$. By using a prime number, P, the lattice points are uniformly scattered in feasible solution region [42]. If the selected prime number, P, is smaller than the dimension, D, of the problem, we use the peak value of the upper MF. After training, this parameter was not changed and was used to supply the difference of parameters, $(D - P + 1)$, in the initial array.

3.2. Evaluation Phase. This phase is used to calculate fitness values for the entire set of particles and to take the MSE ranking for each species. Particles that have improved MSEs between the particles of generation g and $g + 1$ are retained, and this process can be used to attain better performance from the algorithm. Another task in this phase is to remove the redundant particles as determined by a similarity measurement. The conditions of the "particles combination" process are

$$\left\|x_i - x_j\right\|_2 < \frac{r_s}{10}, \qquad \left|\frac{\text{MSE}(x_i) - \text{MSE}(x_j)}{\text{MSE}(x_i)}\right| < \mu_{\text{th}}, \quad i \neq j, \tag{24}$$

where $\|\cdot\|_2$ is the Euclidean norm, r_s is the species radius, and μ_{th} is the threshold. In general, similar (or identical) particles may converge to the same (or similar) optima; our method retains the best of these similar particles. The other redundant particles do not contribute further to the improvement of convergence. In addition, all particles are retained when these particles are similar to the species seed (the best particle of species). Hence, the particles will be led in the best direction while the efficiency of the SEMBP algorithm is improved. Figure 7 describes the removal of redundant particles when particles are identified by similarity measurement. In the case of particles S_1 and Q_5, the particles satisfy the particles combination conditions, and particle Q_5 should be removed. The fitness values of particles Q_1 and Q_2 are almost equal, but the distance, dis(Q_1, Q_2), is not smaller than $0.1 \times r_s$. Although particles Q_3 and Q_4 have a similar location, they do not satisfy the other condition for particle removal.

3.3. Species Determination Phase. The species technique aims to identify multiple species i population followed by identification of the best particle in each species. The dominant particle in each species is regarded as the "neighborhood best" and is deemed the "species seed." All particles that fall within a given distance from the species seed are classified

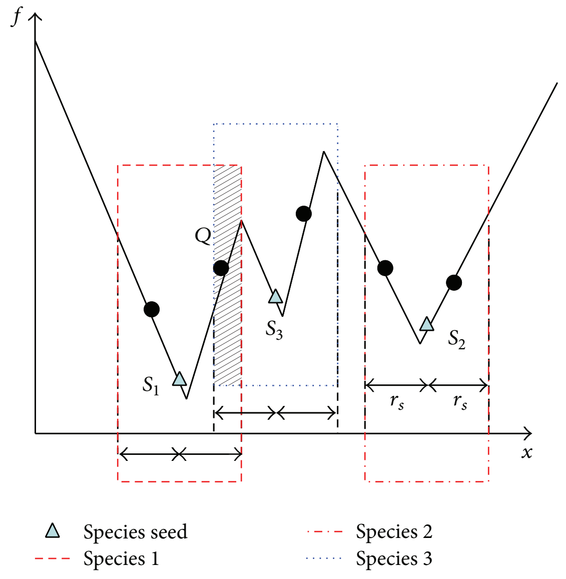


Figure 8: Illustration of species determination for a one-dimensional problem.

as being of the same species. This distance depends on the radius, r_s, which is the particle's distance from the seed. Therefore, if r_s is small, many isolated species are created in each generation. These isolated particles tend to prematurely identify a local minimum. If there are not sufficient numbers of particles in each species, the species will stop evolving. However, if r_s is large, it is possible to cover the entire variable range with one species, and the species technique would have no effect.

Next, the particles in the population are sorted by their fitness values in the MSE ranking step. Then, during species determination, the best performing particle is denoted as the species seed. The particles whose distance from the first species seed is smaller than r_s are categorized into the first species. The remaining particles do not categorize to the first species seed, and the particle with the minimum MSE is selected as a new species seed. The remaining particles are then checked to determine if they belong to the new species. These steps are repeated until all of the particles have been categorized. Figure 8 illustrates this phase. In this example, the algorithm locates three species, and particles S_1, S_2, and S_3 are identified as species seeds. It can be noted that there is an overlap between the first species and the third species. Hence, the previously identified species (centered at S_1) dominates the overlap that belongs to the third species. In other words, particle Q should belong to the species led by S_1.

3.4. IEM Operation Phase. There are three steps in the IEM operation phase: "local search for best particle by BP," "total force calculation," and "movement." To improve the random process, the step length (λ) in movement is chosen to be one that is expected to accelerate the convergence speed. After species determination, each subspecies proceeds to the total force calculation and movement steps, which are the same as in IEMBPs [9]. Nevertheless, in contrast to that of the complete population, determining the electromagnetic charge of each particle in the subpopulations has lower computational complexity. In other words, the complete population should require $\mathrm{PS}\times(\mathrm{PS}-1)\times\cdots\times 1$ charge computations during total force calculation, but the subpopulations should only require $\sum_s p_s \times (p_s - 1) \times \cdots \times 1$ computations. In these expressions, $s$is the number of species and p_s is the number of particles in each of the subpopulation, respectively. The best particle of the subspecies then proceeds to the local search by BP step.

3.4.1. Local Search of Species Seed by BP. The BP technique is adopted to derive a local search procedure in the SEMBP algorithm for AIT2FNS optimization. For clarification, we consider the single-output system and define the error cost function as (22); that is, $E(g) = (1/2)\sum_k e(k)^2$, where g is the generation index. Using the BP technique, the updated parameters law is

$$\mathbf{W}(g+1) = \mathbf{W}(g) + \Delta\mathbf{W}(g) = \mathbf{W}(g) + \eta\left(-\frac{\partial E(g)}{\partial \mathbf{W}}\right), \tag{25}$$

where η is the learning rate. $\mathbf{W} = [\underline{\mathbf{W}}, \overline{\mathbf{W}}, \gamma, \mathbf{W}_\omega, \mathbf{C}]^T$ denote the adjustable parameters, where $\mathbf{C}$ represents the parameters of the TSK layer, $\mathbf{W}_\omega$ represents the consequent weights, $\underline{\mathbf{W}}$ represents the parameters of the lower MFs, $\overline{\mathbf{W}}$ represents the upper MFs parameters, and $\underline{\gamma}$ represents the column vectors; that is,

$$\begin{aligned} \mathbf{C} &= \begin{bmatrix} c & s \end{bmatrix}^T, \\ \mathbf{W}_\omega &= \begin{bmatrix} \underline{\omega}^l & \underline{\omega}^r & \overline{\omega}^l & \overline{\omega}^r \end{bmatrix}^T, \\ \underline{\mathbf{W}} &= \begin{bmatrix} \underline{m}^l & \underline{m}^r & \underline{\sigma}^l & \underline{\sigma}^r \end{bmatrix}^T, \\ \overline{\mathbf{W}} &= \begin{bmatrix} \overline{m}^l & \overline{m}^r & \overline{\sigma}^l & \overline{\sigma}^r \end{bmatrix}^T. \end{aligned} \tag{26}$$

For the nonlinear controller design problem, $\partial E(g)/\partial \mathbf{W}$ cannot be obtained directly [7, 9]. The update law should be multiplied by the system information term (i.e., the plant sensitivity, $\partial y/\partial u$). Based on the results described in the literature [8, 9, 18], the gradient of$E$ should be replaced by the following:

$$\begin{aligned} \frac{\partial E}{\partial \mathbf{W}} &= \frac{\partial E}{\partial e} \cdot \frac{\partial e}{\partial y} \cdot \frac{\partial y}{\partial u} \cdot \frac{\partial u}{\partial \mathbf{W}} \\ &= -\sum_k (e + \Delta e) \frac{\partial u}{\partial \mathbf{W}} \\ &= -\sum_k (e + \Delta e) \frac{\partial O^{(6)}}{\partial \mathbf{W}}, \end{aligned} \tag{27}$$

where $e(k) = y_r(k) - y(k)$ and $\Delta e(k) = e(k) - e(k-1)$. Thus, (25) can be rewritten as

$$\mathbf{W}(g+1) = \mathbf{W}(g) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial \mathbf{W}}. \tag{28}$$

The remaining steps require finding the corresponding partial derivative with respect to each parameter.

The Derivation of the Update Law for **C**. The update law for the AIT2FNS parameter **C** is

$$\begin{aligned} c_j(k+1) &= c_j(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial c_j}, \\ s_j(k+1) &= s_j(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial s_j}, \end{aligned} \tag{29}$$

where

$$\begin{aligned} \frac{\partial O^{(6)}(k)}{\partial c_{j0}} &= \frac{1}{2}\left[\frac{O_{jl}^{(4)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} + \frac{O_{jr}^{(4)}}{\sum_{j=1}^{M} O_{jr}^{(4)}}\right], \\ \frac{\partial O^{(6)}(k)}{\partial c_{ji}} &= \frac{1}{2}\cdot x_i \cdot \left[\frac{O_{jl}^{(4)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} + \frac{O_{jr}^{(4)}}{\sum_{j=1}^{M} O_{jr}^{(4)}}\right], \\ \frac{\partial O^{(6)}(k)}{\partial s_{j0}} &= \frac{1}{2}\left[-\frac{O_{jl}^{(4)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} + \frac{O_{jr}^{(4)}}{\sum_{j=1}^{M} O_{jr}^{(4)}}\right], \\ \frac{\partial O^{(6)}(k)}{\partial s_{ji}} &= \frac{1}{2}\cdot |x_i| \cdot \left[-\frac{O_{jl}^{(4)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} + \frac{O_{jr}^{(4)}}{\sum_{j=1}^{M} O_{jr}^{(4)}}\right]. \end{aligned} \tag{30}$$

The Derivation of the Update Law for $\mathbf{W}_\omega$. The update law for the AIT2FNS parameter $\mathbf{W}_\omega$ is

$$\begin{aligned} \underline{\omega}_j(k+1) &= \underline{\omega}_j(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial \underline{\omega}_j}, \\ \overline{\omega}_j(k+1) &= \overline{\omega}_j(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial \overline{\omega}_j}, \end{aligned} \tag{31}$$

where

$$\begin{aligned} \frac{\partial O^{(6)}(k)}{\partial \underline{\omega}_j^l} &= \frac{1}{2}\left[\frac{\mathrm{TSK}_j^l - O_l^{(5)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} \cdot \frac{\underline{O}_j^{(3)} - O_{jl}^{(4)}}{\underline{\omega}_j^l + \overline{\omega}_j^l}\right], \\ \frac{\partial O^{(6)}(k)}{\partial \underline{\omega}_j^r} &= \frac{1}{2}\left[\frac{\mathrm{TSK}_j^r - O_r^{(5)}}{\sum_{j=1}^{M} O_{jr}^{(4)}} \cdot \frac{\underline{O}_j^{(3)} - O_{jr}^{(4)}}{\underline{\omega}_j^r + \overline{\omega}_j^r}\right], \\ \frac{\partial O^{(6)}(k)}{\partial \overline{\omega}_j^l} &= \frac{1}{2}\left[\frac{\mathrm{TSK}_j^l - O_l^{(5)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} \cdot \frac{\overline{O}_j^{(3)} - O_{jl}^{(4)}}{\underline{\omega}_j^l + \overline{\omega}_j^l}\right], \\ \frac{\partial O^{(6)}(k)}{\partial \overline{\omega}_j^r} &= \frac{1}{2}\left[\frac{\mathrm{TSK}_j^r - O_r^{(5)}}{\sum_{j=1}^{M} O_{jr}^{(4)}} \cdot \frac{\overline{O}_j^{(3)} - O_{jr}^{(4)}}{\underline{\omega}_j^r + \overline{\omega}_j^r}\right]. \end{aligned} \tag{32}$$

The Derivation of the Update Law for $\underline{\mathbf{W}}$. The update law for the AIT2FNS parameter $\underline{\mathbf{W}}$ is

$$\begin{aligned} \underline{m}_{ij}(k+1) &= \underline{m}_{ij}(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial \underline{m}_{ij}} \\ \underline{\sigma}_{ij}(k+1) &= \underline{\sigma}_{ij}(k) + \eta \sum_k (e + \Delta e) \frac{\partial O^{(6)}(k)}{\partial \underline{\sigma}_{ij}}, \end{aligned} \tag{33}$$

where

$$\begin{aligned} \frac{\partial O^{(6)}(k)}{\partial \underline{\mathbf{W}}} &= -\frac{1}{4}\gamma \left[\frac{\mathrm{TSK}_j^l - O_l^{(5)}}{\sum_{j=1}^{M} O_{jl}^{(4)}} \cdot \frac{\underline{\omega}_j^l \cdot \underline{O}_j^{(3)}}{\underline{\omega}_j^l + \overline{\omega}_j^l}\right. \\ &\qquad \left. + \frac{\mathrm{TSK}_j^r - O_r^{(5)}}{\sum_{j=1}^{M} O_{jr}^{(4)}} \cdot \frac{\underline{\omega}_j^r \cdot \underline{O}_j^{(3)}}{\underline{\omega}_j^r + \overline{\omega}_j^r}\right] \\ &\qquad \times \frac{\partial\left((x - \underline{m})/\underline{\sigma}\right)^2}{\partial \underline{\mathbf{W}}}, \\ \frac{\partial}{\partial \underline{m}_{ij}^l}\left(\frac{x_i - \underline{m}_{ij}^l}{\underline{\sigma}_{ij}^l}\right)^2 &= -2\frac{\left(x_i - \underline{m}_{ij}^l\right)}{\left(\underline{\sigma}_{ij}^l\right)^2}, \\ \frac{\partial}{\partial \underline{m}_{ij}^r}\left(\frac{x_i - \underline{m}_{ij}^r}{\underline{\sigma}_{ij}^r}\right)^2 &= -2\frac{\left(x_i - \underline{m}_{ij}^r\right)}{\left(\underline{\sigma}_{ij}^r\right)^2}, \\ \frac{\partial}{\partial \underline{\sigma}_{ij}^l}\left(\frac{x_i - \underline{m}_{ij}^l}{\underline{\sigma}_{ij}^l}\right)^2 &= -2\frac{\left(x_i - \underline{m}_{ij}^l\right)^2}{\left(\underline{\sigma}_{ij}^l\right)^3}, \\ \frac{\partial}{\partial \underline{\sigma}_{ij}^r}\left(\frac{x_i - \underline{m}_{ij}^r}{\underline{\sigma}_{ij}^r}\right)^2 &= -2\frac{\left(x_i - \underline{m}_{ij}^r\right)^2}{\left(\underline{\sigma}_{ij}^r\right)^3}. \end{aligned} \tag{34}$$

The Derivation of the Update Law for $\overline{\mathbf{W}}$. The update law for the AIT2FNS parameter $\overline{\mathbf{W}}$ is

$$\begin{aligned}\overline{m}_{ij}(k+1) &= \overline{m}_{ij}(k) + \eta \sum_k (e+\Delta e) \frac{\partial O^{(6)}(k)}{\partial \overline{m}_{ij}},\\ \overline{\sigma}_{ij}(k+1) &= \overline{\sigma}_{ij}(k) + \eta \sum_k (e+\Delta e) \frac{\partial O^{(6)}(k)}{\partial \overline{\sigma}_{ij}},\end{aligned} \tag{35}$$

where

$$\begin{aligned}\frac{\partial O^{(6)}(k)}{\partial \overline{\mathbf{W}}} &= -\frac{1}{4}\left[\frac{\mathrm{TSK}_j^l - O_l^{(5)}}{\sum_{j=1}^M O_{jl}^{(4)}} \cdot \frac{\overline{\omega}_j^l \cdot \overline{O}_j^{(3)}}{\underline{\omega}_j^l + \overline{\omega}_j^l} + \frac{\mathrm{TSK}_j^r - O_r^{(5)}}{\sum_{j=1}^M O_{jr}^{(4)}} \cdot \frac{\overline{\omega}_j^r \cdot \overline{O}_j^{(3)}}{\underline{\omega}_j^r + \overline{\omega}_j^r}\right] \times \frac{\partial((x-\overline{m})/\overline{\sigma})^2}{\partial \overline{\mathbf{W}}},\\ \frac{\partial}{\partial \overline{m}_{ij}^l}\left(\frac{x_i - \overline{m}_{ij}^l}{\overline{\sigma}_{ij}^l}\right)^2 &= -2\frac{\left(x_i - \overline{m}_{ij}^l\right)}{\left(\overline{\sigma}_{ij}^l\right)^2},\\ \frac{\partial}{\partial \overline{m}_{ij}^r}\left(\frac{x_i - \overline{m}_{ij}^r}{\overline{\sigma}_{ij}^r}\right)^2 &= -2\frac{\left(x_i - \overline{m}_{ij}^r\right)}{\left(\overline{\sigma}_{ij}^r\right)^2},\\ \frac{\partial}{\partial \overline{\sigma}_{ij}^l}\left(\frac{x_i - \overline{m}_{ij}^l}{\overline{\sigma}_{ij}^l}\right)^2 &= -2\frac{\left(x_i - \overline{m}_{ij}^l\right)^2}{\left(\overline{\sigma}_{ij}^l\right)^3},\\ \frac{\partial}{\partial \overline{\sigma}_{ij}^r}\left(\frac{x_i - \overline{m}_{ij}^r}{\overline{\sigma}_{ij}^r}\right)^2 &= -2\frac{\left(x_i - \overline{m}_{ij}^r\right)^2}{\left(\overline{\sigma}_{ij}^r\right)^3}.\end{aligned} \tag{36}$$

The Derivation of the Update Law for $\underline{\gamma}$. The update law for the AIT2FNS parameter $\underline{\gamma}$ is

$$\underline{\gamma}_{ij}(k+1) = \underline{\gamma}_{ij}(k) + \eta \sum_k (e+\Delta e) \frac{\partial O^{(6)}(k)}{\partial \underline{\gamma}_{ij}}, \tag{37}$$

where

$$\frac{\partial O^{(6)}(k)}{\partial \underline{\gamma}_{ij}} = \frac{1}{2}\left[\frac{\mathrm{TSK}_j^l - O_l^{(5)}}{\sum_{j=1}^M O_{jl}^{(4)}} \cdot \frac{\underline{\omega}_j^l}{\underline{\omega}_j^l + \overline{\omega}_j^l} + \frac{\mathrm{TSK}_j^r - O_r^{(5)}}{\sum_{j=1}^M O_{jr}^{(4)}} \cdot \frac{\underline{\omega}_j^r}{\underline{\omega}_j^r + \overline{\omega}_j^r}\right] \frac{\underline{O}_j^{(3)}}{\underline{\gamma}_{ij}}. \tag{38}$$

4. Simulation Results

This section presents two examples of nonlinear system control to demonstrate the performance of the AIT2FNS with the SEMBP algorithm. The first example is nonlinear plant tracking control. In the second example, we focus on a more practical control problem. Hence, the control of temperature in a water bath system is presented. All simulations were performed using Matlab running on an Intel Pentium 4 computer with a clock rate of 3 GHz and 1.25 GB of main memory. The SEMBP algorithm is adopted to adjust the parameters of an AIT2FNS controller such that the tracking error approaches zero. The block diagram of the SEMBP-based AIT2FNS control scheme for the nonlinear system is shown in Figure 5. We adopt MSE as the performance index, which is defined as

$$\mathrm{MSE} \equiv E(g) = \frac{1}{T}\sum_{k=1}^{T} e^2(k), \tag{39}$$

where T is the data number.

Example 1 (nonlinear plant tracking control). Consider the tracking control of the following nonlinear system [42]:

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k), \tag{40}$$

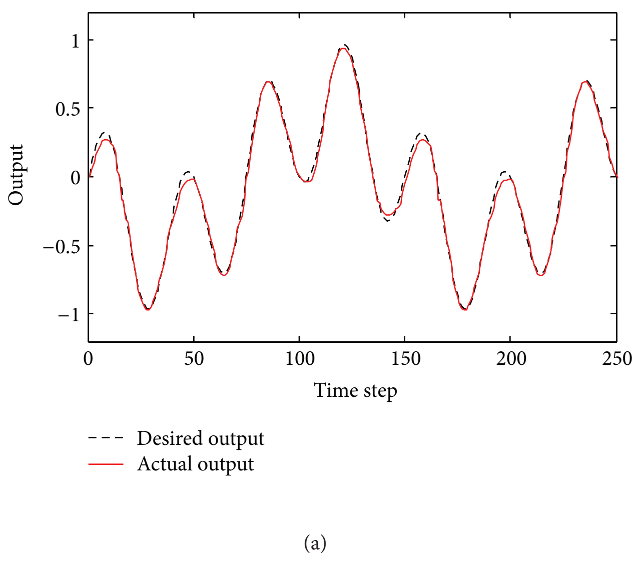
where $y(k)$ is the nonlinear system output and $u(k)$ is the control input of the plant. In designing the AIT2FNS controller, the reference trajectory $y_r(k)$ is

$$y_r(k) = \sin\left(\frac{\pi k}{50}\right)\cos\left(\frac{\pi k}{30}\right). \tag{41}$$

The inputs and output of AIT2FNS are $y_r(k)$, $y(k-1)$, and $u(k)$, respectively. The total number of time steps, N, is 250. The dimension, D, is 120, and the population size, PS, is 113. The initial parameters for AIT2FNS are generated uniformly between $[-1.5, 1.5]$. The threshold, μ_{th}, is selected to be 0.5, and the learning rate of BP is 0.01. The parameters of the SEMBP algorithm and the AIT2FNS are chosen in a manner described as follows:

(i) total number of rules (R): 4;
(ii) network structure (layer 1~layer 6): (2-8-4-8-2-1);
(iii) parameters number of AIT2FNS (D): 120;
(iv) population size (PS): 113;
(v) maximum generation (G): 20.

The simulation results are shown in Figure 9. Figure 9(a) shows the system trajectories after training. A comparison of MSEs between the SEMBP and other algorithms is shown in Figure 9(b). Obviously, the SEMBP algorithm exhibited a similar performance (in terms of MSEs) to the SEM and EM algorithms. However, the SEMBP algorithm spent less computational time and exhibited a better performance than the other algorithms. Table 2 summarizes the corresponding comparisons of simulation results and computational time when using Matlab. After training (20 generations), the MSE of the AIT2FNS was 0.0017631, which was less than the best results of other algorithms (Figure 9(b)). Thus, the AIT2FNS with the SEMBP algorithm spends less time than the other



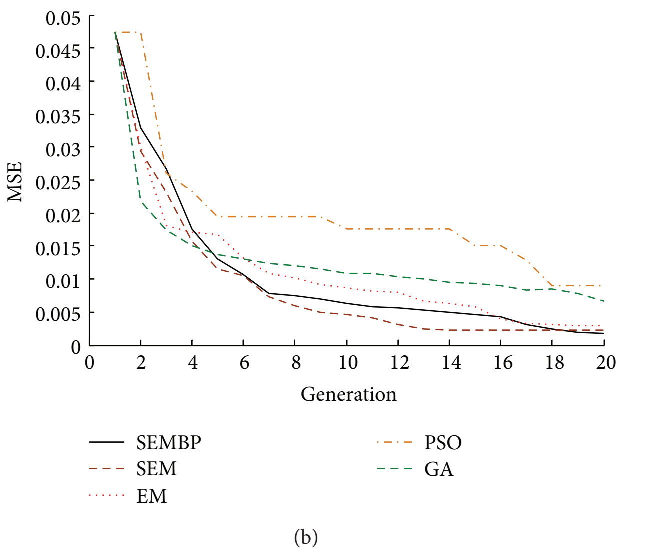


FIGURE 9: Simulation results for Example 1: (a) output trajectory (solid line: actual output; dotted line: desired output) and (b) the best MSE in 20 generations.

TABLE 2: Comparison of average performance in terms of MSEs and computational complexity for Example 2 when using different algorithms ($G = 20$; $D = 120$; $P_s = 113$).

Algorithm	SEMBP	SEM	EM	PSO	GA
Average time	528.4	14933.7	949.5	338.3	333.2
Average MSE	0.0017631	0.0031130	0.0086983	0.0139041	0.0108090
Best MSE	—	0.0022200	0.0029797	0.0089709	0.0066581
Worst MSE	—	0.0037332	0.0244482	0.0313254	0.0140582

algorithms do to achieve higher accuracy and the optimal AIT2NFS. The final IT2 AFMFs are shown in Figures 10(a) and 10(b). The constructed fuzzy rules are given as follows.

Rule 1.

IF x_1 is $\widetilde{F}_{11}$ and x_2 is $\widetilde{F}_{21}$
THEN

$$Y_1 = [-0.3638, 0.0744] + [-0.9677, -0.3175]\, x_1 + [-0.8312, 0.0316]\, x_2, \tag{42}$$

where $\overline{\omega} = [-1.0174, -1.0077]$ and $\underline{\omega} = [-1.0100, -1.0100]$.

Rule 2.

IF x_1 is $\widetilde{F}_{12}$ and x_2 is $\widetilde{F}_{22}$
THEN

$$Y_2 = [-0.2134, 0.8584] + [-0.9346, 0.3284]\, x_1 + [0.0845, 1.5459]\, x_2, \tag{43}$$

where $\overline{\omega} = [-0.8089, -0.8089]$ and $\underline{\omega} = [-0.8089, -0.8089]$.

Rule 3.

IF x_1 is $\widetilde{F}_{13}$ and x_2 is $\widetilde{F}_{23}$
THEN

$$Y_3 = [0.2733, 1.9825] + [-0.5960, 1.3016]\, x_1 + [0.4477, 2.5505]\, x_2, \tag{44}$$

where $\overline{\omega} = [-0.6747, -0.6746]$ and $\underline{\omega} = [-0.6747, -0.6747]$.

Rule 4.

IF x_1 is $\widetilde{F}_{14}$ and x_2 is $\widetilde{F}_{24}$
THEN

$$Y_4 = [-2.6773, -0.3213] + [-0.2803, 2.3337]\, x_1 + [-2.0843, 0.7631]\, x_2, \tag{45}$$

where $\overline{\omega} = [-0.4633, -0.3492]$ and $\underline{\omega} = [-0.4438, -0.4435]$.

4.1. Discussion of Algorithms. Table 2 shows the comparison of MSEs for 20 training generations, and the learning process is repeated for 20 independent runs. Several multiagent based algorithms, SEM, EM, PSO, and GA, are tested. The SEMBP algorithm has a smaller MSE (0.0017631) than the other algorithms do(SEM: 0.0022200, EM: 0.0029797, PSO: 0.0089709, and GA: 0.0066581). Additionally, the SEMBP algorithm only requires 528.4 seconds, which indicates a reduction in the

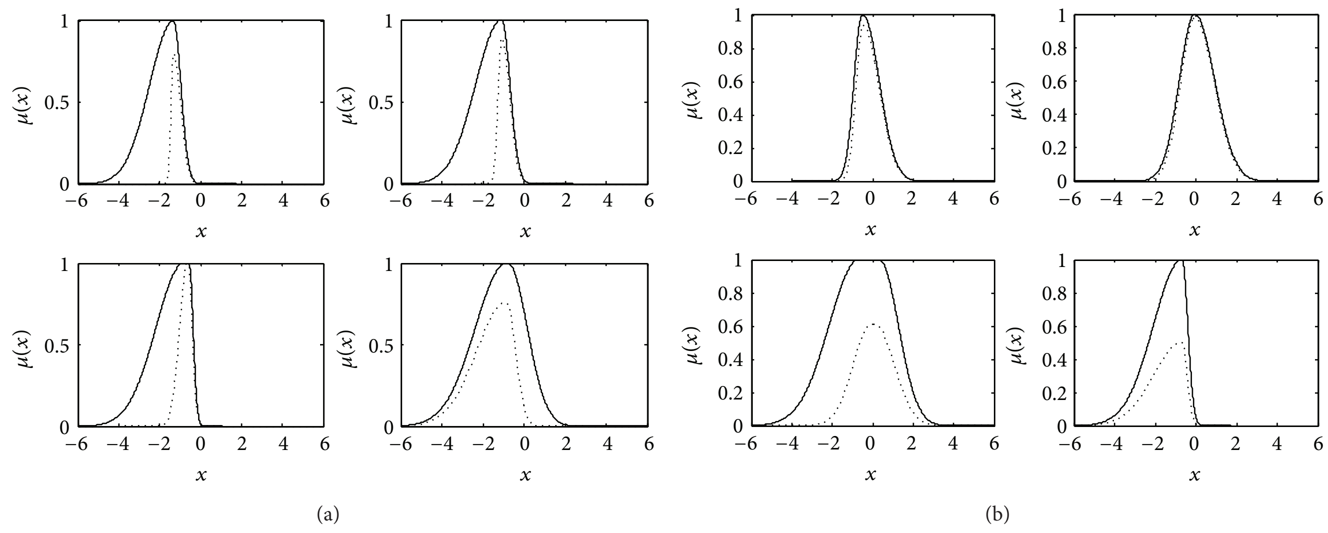


Figure 10: IT2 AFMFs after training for Example 1: (a) MFs for x_1; (b) MFs for x_2.

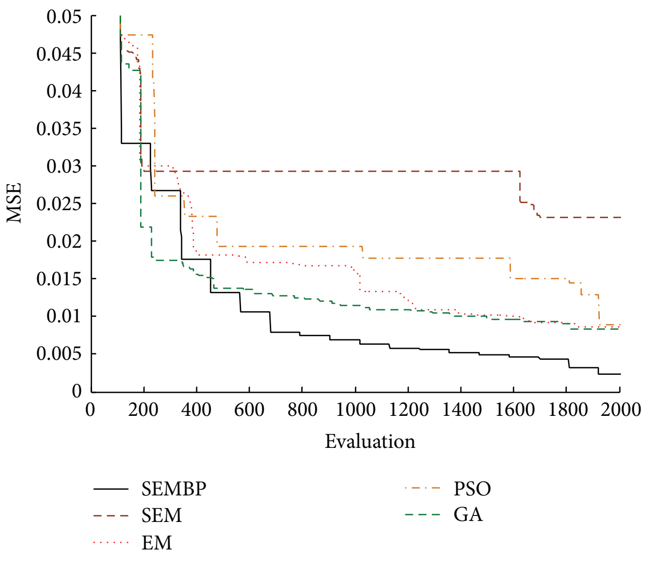


Figure 11: MSE versus number of evaluations for algorithms tested in Example 1.

computational complexity. GA and PSO require less computational time, but they obtain worse results. Figure 11 shows a comparison of various algorithms in terms of MSE versus the number of evaluations. The SEMBP algorithm exhibits better performance in terms of MSE and convergence than the other algorithms.

4.2. Discussion of Network Structure. The comparison results from Example 1 using different fuzzy neural networks with SEMBP are shown in Figure 12 and Table 3. Figure 12(a) shows the comparison of different fuzzy neural networks using the same number of rules ($R = 4$). Figure 12(b) shows the comparison of different fuzzy neural networks using a similar parameter number. Figure 12 and Table 3 show that the AIT2FNS has a better MSE and a higher convergence speed. Due to the TSK-type consequent part and the AFMFs, the AIT2FNS exhibits better approximation performance than the others networks.

4.3. Discussion of Type-Reduction Method. See Table 3; the AIT2FNS and the IT2TFNN-A have similar structure except the type-reduction part. The AIT2FNS adopts the left-most and right-most layers while the IT2TFNN-A adopts the traditional KM algorithm for type-reduction. The result using same rule of AIT2FNS and IT2TFNN-A are 0.0017631 and 0.0085178. The computational costs of them are 528.4 and 678.1 second. As above description, we easily find that the AIT2FNS performs better than IT2TFNN-A with fewer computational effort. We further increased the number of rules to make the parameter number of IT2TFNN-A close to 120. The result improved but is still worse than the result of AIT2FNS.

4.4. Discussion of Maximum Generation (G). The comparison of MSEs for AIT2FNS while using different algorithms for 100 generations is shown in Figure 13. Most curves tend to smooth at a maximum generation ($G = 20$). Therefore, twenty is suggested as a proper maximum generation for AIT2FNSs.

4.5. Discussion of Species Radius (r_s). Table 4 shows the comparison results from Example 1 using various species radii. We used $d_{\max}/3$ (8.439) as the species radius, r_s, which achieves the same MSE with lower computational complexity. In other words, for a species radius smaller than 8.439, the SEMBP algorithm may achieve a similar MSE result, but it requires more computational effort. Therefore, we suggest $d_{\max}/3$ (8.439) to be a proper species radius for an AIT2FNS optimization using uniform initialization via the SEMBP algorithm.

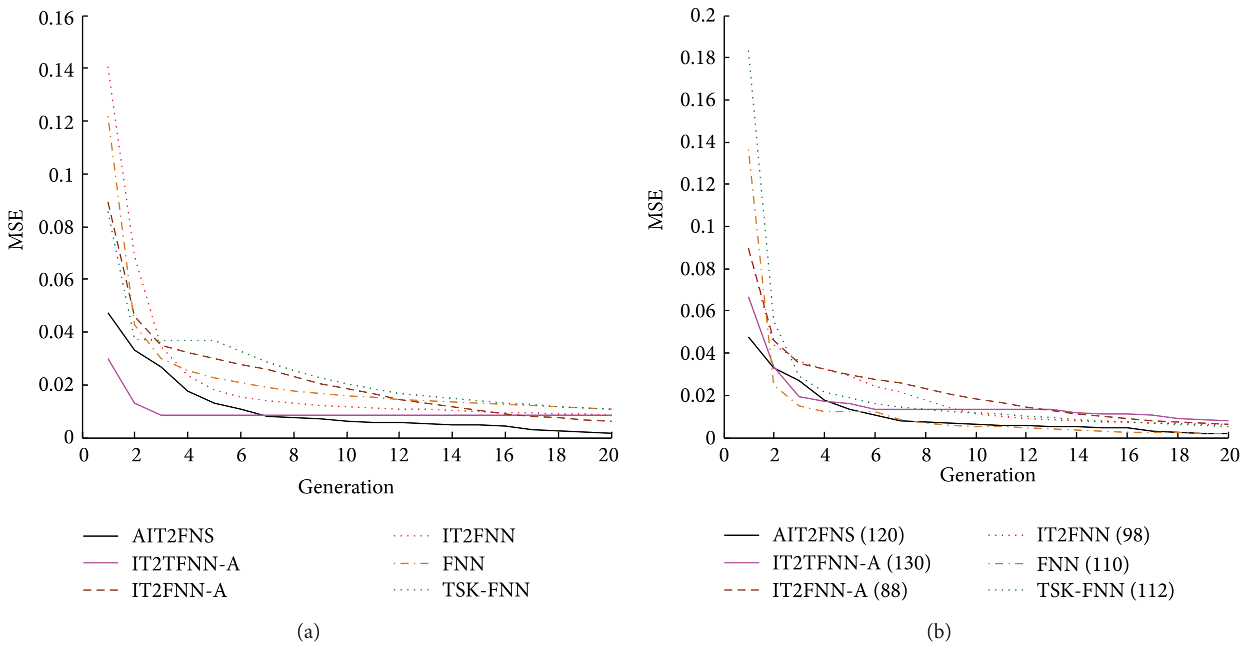


FIGURE 12: Simulation results using different networks for Example 1: (a) the same number of rules and (b) a similar parameter number.

TABLE 3: Comparison of parameter number, computational complexity, and MSE of the SEMBP algorithm for Example 1 using different networks and number of rules ($G = 20$; $P_s = 113$).

Network	Network structure	Number of rules	Parameter number	Time	MSE
AIT2FNS	2-8-4-8-2-1	4	120	528.4	0.0017631
IT2TFNN-A	2-8-4-1	4	104	678.1	0.0085178
	2-10-5-1	5	130	1044.6	0.0077511
IT2FNN-A	2-8-4-1	4	88	630.3	0.0063496
IT2FNN	2-8-4-1	4	56	580.6	0.0085290
	2-14-7-1	7	98	851.5	0.0062085
FNN	2-8-4-1	4	20	178.1	0.0107710
	2-20-10-1	10	50	408.2	0.0039030
	2-32-16-1	16	80	633.2	0.0036811
	2-44-22-1	22	110	768.8	0.0019379
TSK-FNN	2-8-4-1	4	28	185.3	0.0107014
	2-20-10-1	10	70	408.2	0.0036001
	2-32-16-1	16	112	684.6	0.0034152

4.6. Discussion of Threshold for Particle Recombination (μ_{th}). Table 5 illustrates the discussion of threshold in Example 1. The method of decreasing the threshold can effectively reduce the computational complexity in the SEMBP algorithm. Finally, we set the threshold value to 0.5.

Example 2 (control of a water bath temperature system). Herein, we present a more practical control problem. The control of the water bath temperature system is given by [5, 15, 45]

$$\frac{dy(t)}{dt} = \frac{u(t)}{C} + \frac{Y_0 - y(t)}{RC}, \qquad (46)$$

where $y(t)$ is the system output temperature in °C, $u(t)$ is heating flowing inward the system, Y_0 is the room temperature, C is the equivalent thermal capacity, and R is the equivalent thermal resistance between the system borders and surroundings.

Assuming that R and C are constants, we can rewrite (46) into a discrete-time form with some reasonable approximation. Hence, the system is described as

$$y(k+1) = e^{-\alpha T_s} y(k) + \frac{(\beta/\alpha)\left(1 - e^{-\alpha T_s}\right)}{1 + e^{0.5y(k)-40}} u(k) + \left[1 - e^{-\alpha T_s}\right] Y_0. \qquad (47)$$

Table 4: Comparison of MSEs and computational complexity for Example 1 when using different radii ($G = 20$; $D = 120$; $P_s = 113$).

Species radius	16.879 ($d_{\max}/1.5$)	12.659 ($d_{\max}/2$)	10.127 ($d_{\max}/2.5$)	8.439 ($d_{\max}/3$)	7.234 ($d_{\max}/3.5$)
Average time	364.3	374.4	419.0	528.4	1040.5
Average MSE	0.011802	0.008501	0.004335	0.0017631	0.0017631

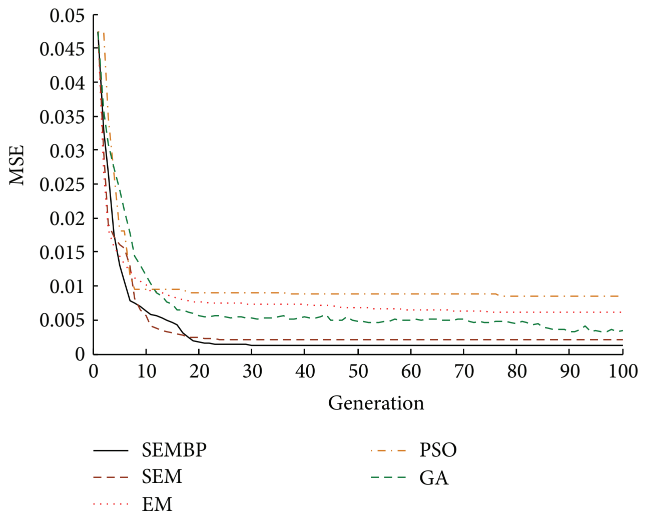


Figure 13: Comparison of MSEs for Example 1 in 100 generations.

Table 5: Comparison of MSEs and computational complexity for Example 1 when using different thresholds ($G = 20$; $D = 120$).

μ_{th}	1.5	1	0.5	0.1	0.01
Time	468.9	487.3	528.4	551.6	611.0
MSE	0.0023779	0.0019786	0.0017631	0.0017631	0.0017631
Redundant particle	14	12	9	6	0

The system parameters used here are $\alpha = 1.00151 \times 10^{-4}$, $\beta = 8.67973 \times 10^{-3}$, and $Y_0 = 25$ (°C), which were obtained from a real world water bath manufacturer [45]. The plant control input $u(k)$ should be positive and is limited between 0 V and 5 V. The reference water temperature $y_r(k)$ is

$$y_r(k) = \begin{cases} 34^\circ\text{C} & \text{for } k \le 30, \\ (34 + 0.5(k-30))^\circ\text{C} & \text{for } 30 < k \le 50, \\ (44 + 0.8(k-50))^\circ\text{C} & \text{for } 50 < k \le 70, \\ (60 + 0.5(k-70))^\circ\text{C} & \text{for } 70 < k \le 30, \\ 70^\circ\text{C} & \text{for } 90 < k \le 120. \end{cases} \tag{48}$$

It can be noted that the upper and lower bounds of the reference water temperature are 34 and 70. The actual water temperature is limited at 10°C~90°C. Thus, the inputs for AIT2FNS are the previous plant output, $(y(k-1)-50)/20$, and the reference water temperature, $(y_r(k-1)-50)/20$, which are scaled to the interval $[-2, 2]$ by experience. The output of the AIT2FNS algorithm is the plant control input, $u(k)$. Each simulation is performed over 120 time steps. The AIT2FNS controller is used to generate the proper control input such that the system output follows the reference trajectory.

In this simulation, the dimension size is 90 and the nearest prime number is 89. Therefore, a population size of 89 is chosen to construct the uniform array using the good lattice point method between $[-2, 2]$. The threshold, μ_{th}, is selected to be 0.5, and the learning rate of BP is set at 0.01. The parameters of the SEMBP algorithm and the AIT2FNS are chosen as follows:

(i) total number of rules (R): 3;

(ii) network structure (layer 1~layer 6): (2-6-3-6-2-1);

(iii) parameters number of AIT2FNS (D): 90;

(iv) population size (PS): 89;

(v) maximum generation (G): 20.

The simulation results are shown in Figure 14. Figure 14(a) shows the system trajectories after 20 training generations (solid line: plant output; dotted line: reference signal; blue dashed line: control signal). A comparison of the MSE results between the SEMBP and each algorithm's best result (SEM, EM, PSO, and GA) is shown in Figure 14(b). The final IT2 AFMFs for x_1 and x_2 are shown in Figures 15(a) and 15(b), respectively. The constructed fuzzy rules are as follows:

Rule 1.

IF x_1 is $\tilde{F}_{11}$ and x_2 is $\tilde{F}_{21}$
THEN

$$\begin{aligned} Y_1 = {} & [-1.1280, -0.0510] + [-1.5286, 2.2266]\, x_1 \\ & + [-1.8038, 0.7194]\, x_2, \end{aligned} \tag{49}$$

where $\overline{\omega} = [0.9664, 1.0040]$ and $\underline{\omega} = [1.0040, 1.0040]$.

Rule 2.

IF x_1 is $\tilde{F}_{12}$ and x_2 is $\tilde{F}_{22}$
THEN

$$\begin{aligned} Y_2 = {} & [0.4594, 1.6820] + [-2.8075, 0.9879]\, x_1 \\ & + [0.4894, 3.0392]\, x_2, \end{aligned} \tag{50}$$

where $\overline{\omega} = [-0.3894, -0.3894]$ and $\underline{\omega} = [-0.3894, -0.3894]$.

Rule 3.

IF x_1 is $\tilde{F}_{13}$ and x_2 is $\tilde{F}_{23}$

TABLE 6: Comparison results of algorithms, in terms of MSE and computational complexity, for Example 2 ($G = 20$; $D = 90$; $P_s = 89$).

Algorithm	SEMBP	SEM	EM	PSO	GA
Average time	276.1	4981.5	465.2	232.4	236.0
Average MSE	1.9181	1.9279	2.2301	2.1697	2.2299
Best MSE	—	1.9188	1.9689	1.9191	1.9443
Worst MSE	—	1.9584	2.8230	2.5505	2.5018

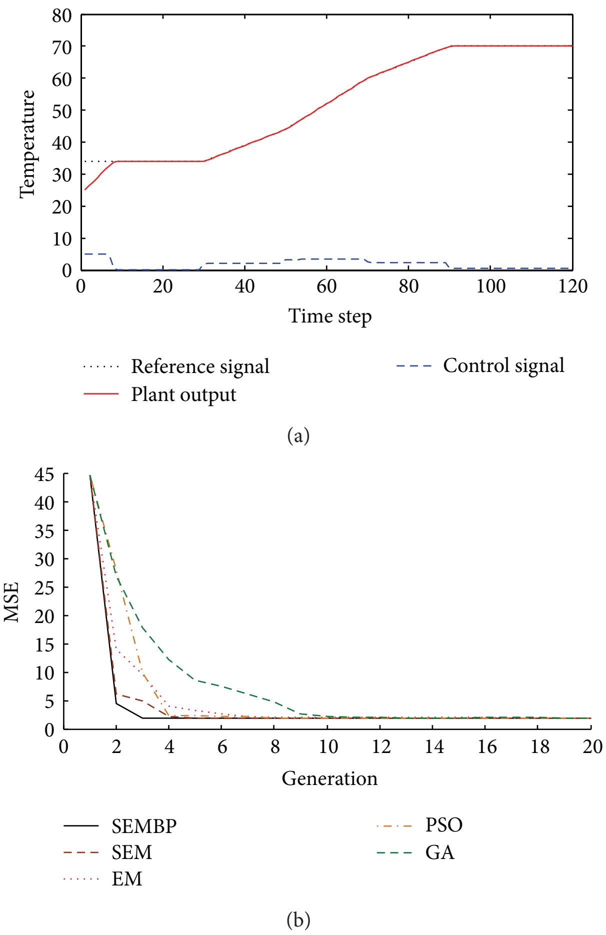


FIGURE 14: Simulation results of Example 2: (a) output trajectories (solid line: plant output; dotted line: reference signal; blue dashed line: control signal) and (b) comparison results in MSE after 20 generations.

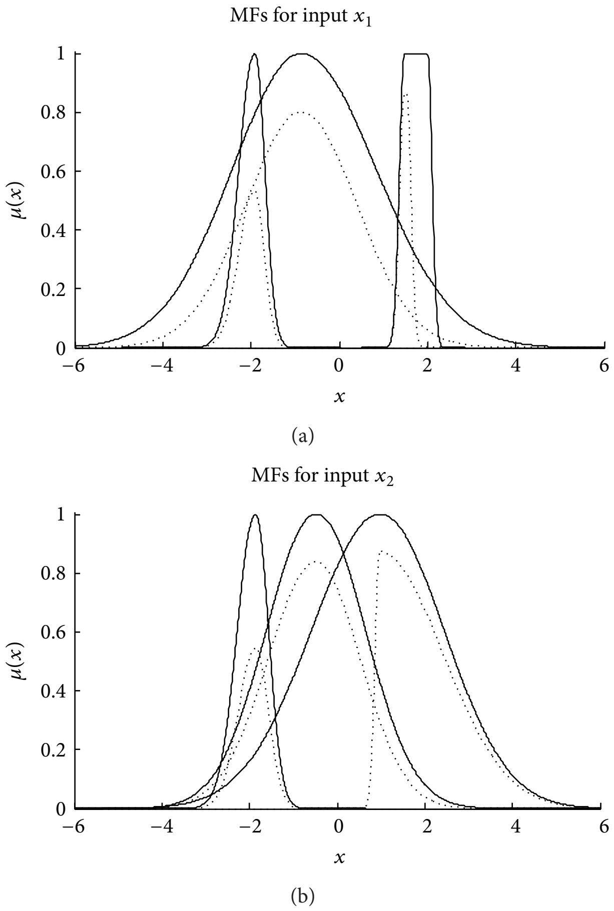


FIGURE 15: Membership functions for Example 2: (a) MFs for x_1 after training and (b) MFs for x_2 after training.

THEN

$$Y_3 = [-1.8313, -0.5693] + [-0.2372, 3.6584]\, x_1 + [-0.7165, 1.9071]\, x_2, \tag{51}$$

where $\overline{\omega} = [-1.5306, -1.5279]$ and $\underline{\omega} = [-1.5285, -1.5283]$.

4.7. Discussion of Algorithms. The curves in Figure 14(b) seem to settle at the similar value since they are depicted by their best results. However, as shown in Figure 14(b), we can observe that the SEMBP algorithm has the fastest convergence speed than other algorithms. Table 6 summarizes the comparison of MSE results and computational effort for 20 training generations repeated for 20 independent runs (algorithms: SEM, EM, PSO, and GA). The MSE of the SEMBP algorithm (MSE: 1.9181) is smaller than the other average MSEs of other algorithms (SEM: 1.9279, EM: 2.2301, PSO: 2.1697, and GA: 2.2299). The SEM algorithm also exhibits better simulation results than the EM algorithm does. This result indicates that the species technique can enhance the approximation accuracy. For comparison of computational complexity, the SEMBP algorithm requires 276.1 seconds, which indicates a reduction in the computational effort. Additionally, the SEMBP algorithm produced better MSEs than the GA and PSO algorithms did, which had similar (or smaller) computational times. From the best MSE, the SEM and PSO algorithms exhibit similar performance to

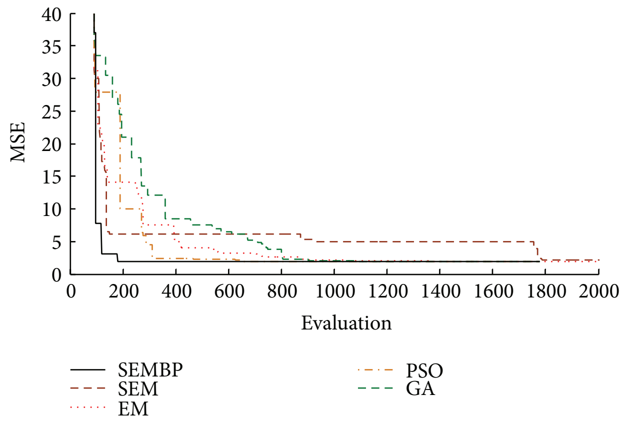


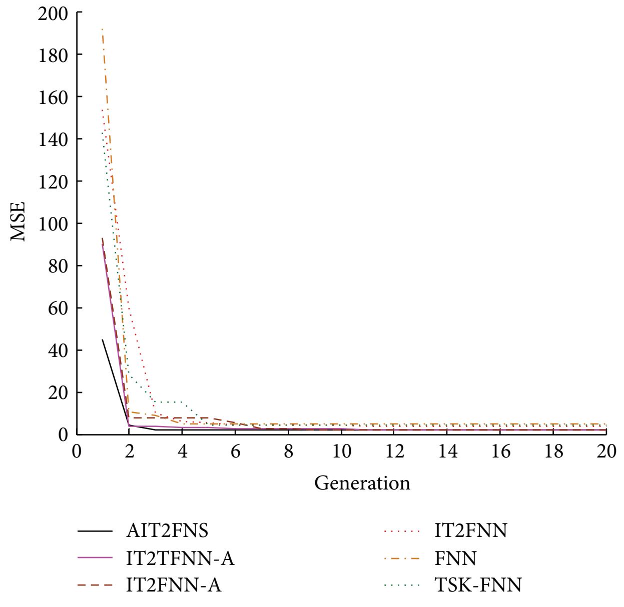
Figure 16: Comparison of MSE versus number of evaluations for algorithms examined in Example 2.

that of the SEMBP algorithm. However, they require more independent trials, where only one trial is needed for the SEMBP. Additionally, Figure 16 shows the comparison results in MSEs versus the number of evaluations for the best result of each algorithm. The SEMBP algorithm also achieves better MSEs for the same number of evaluations. As above description, we can conclude that the SEMBP demonstrates the better performance than other algorithms.

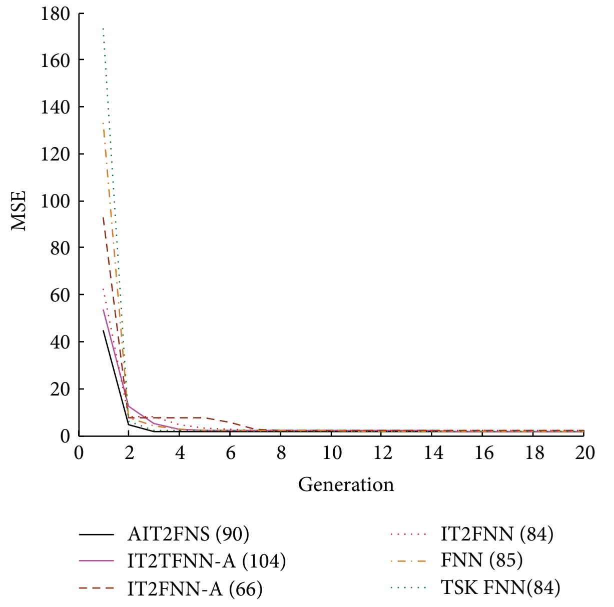
4.8. Discussion of Network Structures. The comparative results of the network structure for Example 2 are shown in Figure 17. Figure 17(a) shows the comparison of different fuzzy neural networks using the same number of rules ($R = 3$). Figure 17(b) shows the comparison of different fuzzy neural networks using a similar parameter number. Figure 17 demonstrates that the AIT2FNS algorithm has a better initial MSE and a higher convergence speed. As seen in Table 7, the AIT2FNS performs better than the other algorithms do while using the same number of rules ($R = 3$). Obviously, the AIT2FNS using TSK fuzzy rules and the AFMFs demonstrate better performance than the other networks do.

4.9. Discussion of Type-Reduction Method. See Table 7; the results using same rule of AIT2FNS and IT2TFNN-A are 01.9181 and 2.0504. The computational costs of them are 276.1 and 367.7 second. Obviously, the AIT2FNS performs better than IT2TFNN-A with fewer computational effort. We also increased the number of rule to make the parameter number of IT2TFNN-A similar to AIT2FNS case. The result in MSE is 1.9371 which is very close to the result of AIT2FNS. However, the computational cost is much huger.

4.10. Discussion of Maximum Generation (G). The comparison of AIT2FNS, in terms of MSE, with regard to the use of different algorithms (where $D = 90$, $P_s = 89$) for 100 generations is shown in Figure 18. These six algorithms achieve similar MSEs when the number of generations is sufficiently large ($G > 40$). Also, the curves tend to smooth at approximately generation ($G = 20$). Therefore, we suggest



(a)



(b)

Figure 17: Simulation results comparing the different networks for Example 2: (a) the same number of rules and (b) the similar parameter number.

that 20 is a proper maximum generation for AIT2FNS optimization. This parameter can be modified to be the MSE specification, the maximum number of evaluations, or other stop criterion.

4.11. Discussion of Species Radius (r_s). Table 8 shows the comparison of results from Example 2 when using different species radii. The good lattice point method is used to construct the uniform initialization; therefore, the population size in Example 2 is chosen as PS = 89, where $D = 90$

Table 7: Comparison of parameter number, computational complexity, and MSEs with the SEMBP algorithm for Example 2 while using different networks and number of rules ($G = 20$).

Network	Network structure	Number of rules	Parameter number	Time	MSE
AIT2FNS	2-6-3-6-2-1	3	90	276.1	1.9181
IT2TFNN-A	2-6-3-1	3	78	367.7	2.0504
	2-8-4-1	4	104	587.8	1.9371
IT2FNN-A	2-6-3-1	3	66	302.9	1.9844
IT2FNN	2-6-3-1	3	42	231.8	3.6466
	2-12-6-1	6	84	418.5	1.9694
FNN	2-6-3-1	3	15	94.4	11.4574
	2-12-6-1	6	30	146.4	4.9611
	2-24-12-1	12	60	272.2	2.0214
	2-34-17-1	17	85	344.3	1.9210
TSK-FNN	2-6-3-1	3	21	141.8	8.5124
	2-12-6-1	6	42	187.8	3.8451
	2-24-12-1	12	84	308.56	1.9332

Table 8: Comparison of MSEs and computational complexity for Example 2 while using different radii ($G = 20$; $D = 90$; $P_s = 89$).

Species radius	19.4765 ($d_{\max}/2$)	14.6074 ($d_{\max}/2.5$)	11.6859 ($d_{\max}/3$)	9.7382 ($d_{\max}/3.5$)	8.3471 ($d_{\max}/4$)
Average time	230.0	240.2	276.1	473.6	558.9
Average MSE	1.9211	1.9211	1.9181	1.9181	1.9181

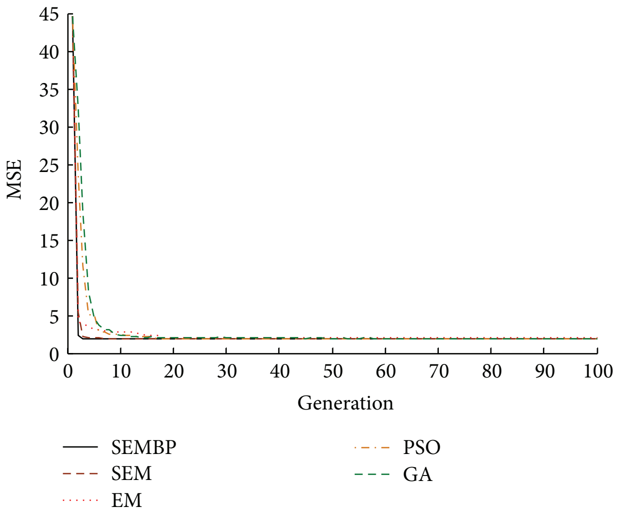


Figure 18: Comparison of MSEs for 100 generations.

is the dimension of the problem. We use $d_{\max}/3$ (11.6859) as the species radius r_s, which achieves a similar MSE and requires less computational complexity. In other words, if the species radius is smaller than 11.6859, the SEMBP algorithm achieves a similar result in terms of MSE, but it requires more computational effort. Therefore, we propose $d_{\max}/3$ (11.6859) to be a proper species radius for AIT2FNS optimization via the SEMBP algorithm.

4.12. Discussion of Threshold for Particle Recombination (μ_{th}). Table 9 shows the comparative results from Example 2 while

Table 9: Comparison of MSEs and computational complexity for Example 2 when using different thresholds ($G = 20$; $D = 90$).

μ_{th}	0.8	0.5	0.1	0.01	0.001
Time	268.9	276.1	283.2	293.6	305.8
MSE	2.012	1.9181	1.9181	1.9181	1.9186
Redundant particles	9	7	4	1	0

using different thresholds μ_{th}. The smaller threshold may have more redundant particles, which cause redundant computation. Alternately, a larger threshold, which requires less time, removes more redundant particles. Nevertheless, higher MSE values are obtained because too many particles are removed. As shown in the above discussion, a smaller threshold not only attains better results but also reduces computational complexity. We conclude that the method for decreasing the threshold can effectively reduce computational complexity in the SEMBP algorithm. Finally, we set the threshold, μ_{th}, to be 0.5.

5. Conclusion

In this paper, we propose a novel interval type-2 fuzzy neural system with the hybrid-learning algorithm SEMBP for nonlinear controller design. The proposed AIT2FNS method uses a type-2 AFMFS and a TSK-type consequent part to enhance the performance of the IT2FNN. In addition, the type reduction was integrated into the adaptive network layers, and this integration reduced the computational complexity. The update laws of the AIT2FNS for nonlinear system control are given by SEMBP. The SEMBP algorithm combines EM and BP with the species technique and uniform initialization to obtain faster convergence, reduced computational complexity, and global optimization. Simulation results including nonlinear system tracking control and water bath temperature control were presented to demonstrate the effectiveness and the performance of the proposed SEMBP and AIT2FNS methods.

Conflict of Interests

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

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments and valuable suggestions. This work was partially supported by the National Science Council, Taiwan, under contract no. NSC-100-2221-E-005-093-MY2.

References

[1] M. Biglarbegian, W. W. Melek, and J. M. Mendel, "On the stability of interval type-2 tsk fuzzy logic control systems," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 40, no. 3, pp. 798–818, 2010.

[2] J. R. Castro, O. Castillo, P. Melin, and A. Rodríguez-Díaz, "A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks," *Information Sciences*, vol. 179, no. 13, pp. 2175–2193, 2009.

[3] K.-H. Cheng, C.-F. Hsu, C.-M. Lin, T.-T. Lee, and C. Li, "Fuzzy-neural sliding-mode control for DC-DC converters using asymmetric gaussian membership functions," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp. 1528–1536, 2007.

[4] C.-F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 34, no. 2, pp. 997–1006, 2004.

[5] C.-F. Juang and P.-H. Chang, "Designing fuzzy-rule-based systems using continuous ant-colony optimization," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 138–149, 2010.

[6] C.-F. Juang, C.-M. Hsiao, and C.-H. Hsu, "Hierarchical cluster-based multispecies particle-swarm optimization for fuzzy-system optimization," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 14–26, 2010.

[7] C.-H. Lee, "Stabilization of nonlinear nonminimum phase systems: adaptive parallel approach using recurrent fuzzy neural network," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 34, no. 2, pp. 1075–1088, 2004.

[8] C.-H. Lee, C.-T. Li, and F.-Y. Chang, "A species-based improved electromagnetism-like mechanism algorithm for TSK-type interval-valued neural fuzzy system optimization," *Fuzzy Sets and Systems*, vol. 171, pp. 22–43, 2011.

[9] C.-H. Lee, F.-K. Chang, C.-T. Kuo, and H.-H. Chang, "A hybrid of electromagnetism-like mechanism and back-propagation algorithms for recurrent neural fuzzy systems design," *International Journal of Systems Science*, vol. 43, no. 2, pp. 231–247, 2012.

[10] C. H. Lee, J. L. Hong, Y. C. Lin, and W. Y. Lai, "Type-2 fuzzy neural network systems and learning," *International Journal of Computational Cognition*, vol. 1, no. 4, pp. 79–90, 2003.

[11] C.-H. Lee and Y.-C. Lin, "An adaptive type-2 fuzzy neural controller for nonlinear uncertain systems," *Control and Intelligent Systems*, vol. 33, no. 1, pp. 13–25, 2005.

[12] C.-H. Lee and H.-Y. Pan, "Performance enhancement for neural fuzzy systems using asymmetric membership functions," *Fuzzy Sets and Systems*, vol. 160, no. 7, pp. 949–971, 2009.

[13] C.-H. Lee and C.-C. Teng, "Identification and control of dynamic systems using recurrent fuzzy neural networks," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 4, pp. 349–366, 2000.

[14] C.-H. Lee and C.-C. Teng, "Fine tuning of membership functions for fuzzy neural systems," *Asian Journal of Control*, vol. 3, no. 3, pp. 216–225, 2001.

[15] C.-J. Lin, "A GA-based neural fuzzy system for temperature control," *Fuzzy Sets and Systems*, vol. 143, no. 2, pp. 311–333, 2004.

[16] C.-J. Lin and W.-H. Ho, "An asymmetry-similarity-measure-based neural fuzzy inference system," *Fuzzy Sets and Systems*, vol. 152, no. 3, pp. 535–551, 2005.

[17] C. T. Lin and C. S. G. Lee, *Neural Fuzzy Systems*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.

[18] F.-J. Lin, R.-J. Wai, and C.-C. Lee, "Fuzzy neural network position controller for ultrasonic motor drive using push-pull DC-DC converter," *IEE Proceedings: Control Theory and Applications*, vol. 146, no. 1, pp. 99–107, 1999.

[19] P.-Z. Lin and T.-T. Lee, "Robust self-organizing fuzzy-neural control using asymmetric Gaussian membership functions," *International Journal of Fuzzy Systems*, vol. 9, no. 2, pp. 77–85, 2007.

[20] P. Melin and O. Castillo, "A new method for adaptive control of non-linear plants using Type-2 fuzzy logic and neural networks," *International Journal of General Systems*, vol. 33, no. 2-3, pp. 289–304, 2004.

[21] T. Ozen and J. M. Garibaldi, "Effect of type-2 fuzzy membership function shape on modelling variation in human decision making," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 971–976, July 2004.

[22] O. Castillo, L. Aguilar, N. Cázarez, and S. Cárdenas, "Systematic design of a stable type-2 fuzzy logic controller," *Applied Soft Computing Journal*, vol. 8, no. 3, pp. 1274–1279, 2008.

[23] O. Castillo and P. Melin, "A new approach for plant monitoring using Type-2 fuzzy logic and fractal theory," *International Journal of General Systems*, vol. 33, no. 2-3, pp. 305–319, 2004.

[24] O. Castillo and P. Melin, *Type-2 Fuzzy Logic: Theory and Applications*, Springer, 2008.

[25] H. A. Hagras, "A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 4, pp. 524–539, 2004.

[26] H. Hagras, F. Doctor, V. Callaghan, and A. Lopez, "An incremental adaptive life long learning approach for type-2 fuzzy embedded agents in ambient intelligent environments," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 1, pp. 41–55, 2007.

[27] H. Hagras, "Type-2 FLCs: a new generation of fuzzy controllers," *IEEE Computational Intelligence Magazine*, vol. 2, no. 1, pp. 30–43, 2007.

[28] B. Q. Hu and C. K. Kwong, "On type-2 fuzzy sets and their t-norm operations," *Information Sciences*, vol. 255, pp. 58–81, 2014.

[29] B. Q. Hu and C. Y. Wang, "On type-2 fuzzy relations and interval-valued type-2 fuzzy sets," *Fuzzy Sets and Systems*, vol. 236, pp. 1–32, 2014.

[30] R. John and S. Coupland, "Type-2 fuzzy logic: a historical view," *IEEE Computational Intelligence Magazine*, vol. 2, no. 1, pp. 57–62, 2007.

[31] N. N. Karnik, J. M. Mendel, and Q. Liang, "Type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 6, pp. 643–658, 1999.

[32] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice-Hall, Upper Saddle River, NJ, USA, 2001.

[33] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, 2002.

[34] R. Martínez, O. Castillo, and L. T. Aguilar, "Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms," *Information Sciences*, vol. 179, no. 13, pp. 2158–2174, 2009.

[35] R. Sepúlveda, O. Castillo, P. Melin, A. Rodríguez-Díaz, and O. Montiel, "Experimental study of intelligent controllers under uncertainty using type-1 and type-2 fuzzy logic," *Information Sciences*, vol. 177, no. 10, pp. 2023–2048, 2007.

[36] R. Sepúlveda, O. Castillo, P. Melin, and O. Montiel, "An efficient computational method to implement type-2 fuzzy logic in control applications," in *Analysis and Design of Intelligent Systems Using Soft Computing Techniques*, pp. 45–52, 2007.

[37] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," *Information Sciences*, vol. 8, pp. 199–249, 1975.

[38] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[39] H. Bustince, "Indicator of inclusion grade for interval-valued fuzzy sets: application to approximate reasoning based on interval-valued fuzzy sets," *International Journal of Approximate Reasoning*, vol. 23, no. 3, pp. 137–209, 2000.

[40] S. I. Birbil and S.-C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.

[41] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 440–458, 2006.

[42] K.-T. Fang and Y. Wang, *Number-Theoretic Methods in Statistics*, vol. 51, Chapman & Hall, 1994.

[43] Q. Zhang, J. Sun, E. Tsang, and J. Ford, "Hybrid estimation of distribution algorithm for global optimization," *Engineering Computations*, vol. 21, no. 1, pp. 91–107, 2004.

[44] K. T. Fang, Y. Wang, and P. M. Bentler, "Some applications of number-theoretic methods in statistics," *Statistical Science*, vol. 9, no. 3, pp. 416–428, 1994.

[45] J. Tanomaru and S. Omatu, "Process control by on-line trained neural controllers," *IEEE Transactions on Industrial Electronics*, vol. 39, no. 6, pp. 511–521, 1992.