Building a Fuzzy Model with Transparent Membership Functions through Constrained Evolutionary Optimization

Min-Soeng Kim, Chang-Hyun Kim, and Ju-Jang Lee

Abstract: In this paper, a new evolutionary scheme to design a TSK fuzzy model from relevant data is proposed. The identification of the antecedent rule parameters is performed via the evolutionary algorithm with the unique fitness function and the various evolutionary operators, while the identification of the consequent parameters is done using the least square method. The occurrence of the multiple overlapping membership functions, which is a typical feature of unconstrained optimization, is resolved with the help of the proposed fitness function. The proposed algorithm can generate a fuzzy model with transparent membership functions. Through simulations on various problems, the proposed algorithm found a TSK fuzzy model with better accuracy than those found in previous works with transparent partition of input space.

Keywords: Evolutionary algorithm, model interpretability, Takagi-Sugeno-Kang fuzzy model, time series prediction.

1. INTRODUCTION

Fuzzy systems are successfully applied to many different application areas. A fuzzy system consists of several fuzzy IF-THEN rules, which map inputs to outputs. Fuzzy systems are very suitable for complex systems when it is difficult or impossible to describe the system mathematically. One of the most important considerations when designing a fuzzy system is that there is rarely a systematic design procedure. The generation of membership functions for each fuzzy set is important, as well as the generation of the fuzzy rules.

There have been many efforts to solve this problem as well-addressed in [1] including the advantages and the disadvantages of each algorithm. Due to the capability of searching irregular and high-dimensional solution space, the evolutionary algorithms such as GAs (genetic algorithms) and ESs (evolutionary strategies) have received much attention recently as in [1-4] (find additional references and discussions in those papers). In particular, the Takagi-Sugeno-Kang (TSK)-type fuzzy model [5] has a great advantage due to its representative power; it is capable of describing a highly nonlinear system. In [4] and [6], GAs are used successfully to design a TSK fuzzy model. They used a rule-based approach with the triangular membership functions and applied GAs, not only to find the antecedent parameters of the fuzzy sets, but also to locate the consequent parameters of the fuzzy model, all of which enlarge the dimension of the search space as will be addressed in the remainder of this paper.

In the following, we propose a novel evolutionary scheme to design a TSK fuzzy model from data. The identification of the antecedent rule parameters is carried out via the evolutionary algorithm with the newly developed fitness function and the various evolutionary operators, while the identification of the consequent parameters is made with the data pattern manipulations and the pseudo inverse method. The problem of multiple overlapping membership functions is perceived as a critical issue named 'model interpretability' in fuzzy modeling as noticed in [14, 15]. The occurrence of the multiple overlapping membership functions, which is a typical feature of unconstrained optimization, is resolved with the help of the newly defined fitness function. This is accomplished while preserving the possibility of adapting the multiple overlapping membership functions if necessary to solve the given problem.

Moreover, by taking a partition-based approach to building a fuzzy model, it will be shown that the proposed algorithm provides a simple and intuitive design procedure and even results in superior performance capability. The remainder of the paper is as follows. Section 2 explains the TSK fuzzy model used and develops the procedure to obtain the consequent parameters in the TSK fuzzy model. Section 3 describes the evolutionary design process.

Manuscript received February 17, 2004; accepted July 1, 2004. Recommended by Editorial Board member Jietae Lee under the direction of Editor Jin Bae Park.

Min-Soeng Kim, Chang-Hyun Kim, and Ju-Jang Lee are with the Department of Electrical Engineering and Computer Science, KAIST, 373-1, Gusung-dong, Yusung-gu, Daejon 305-701, Korea (e-mails: {mibella, sunnine}@odyssey.kaist.ac.kr, jjlee@ee.kaist.ac.kr).

The performance of the proposed algorithm compared with previous works will be provided in Section 4. Finally, Section 5 concludes the paper.

2. TSK FUZZY MODEL

2.1. General descriptions

The TSK fuzzy model [5] consists of IF-THEN rules where the rule consequents are usually constant values (singletons) or linear functions of the inputs.

$$R_i : \text{IF } x_1 \text{ is } A_{i1} \text{ and } \cdots x_n \text{ is } A_{in},$$
$$\text{Then } y_i = c_{i0} + c_{i1}x_1 + \cdots + c_{in}x_n \quad (1)$$
$$\text{for } i = 1, 2, \cdots, N_R$$

where N_R is the number of rules, $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]$ is the input vector, y_i is the output of the i-th rule, A_{ij} are the antecedent fuzzy sets that are characterized by membership functions (MFs) $\mu_{A_{ij}}(x_j)$, and c_{ij} are real-valued weight parameters. The model output is computed by

$$y = \frac{\sum_{i=1}^{N_R} \tau_i y_i}{\sum_{i=1}^{N_R} \tau_i} = \frac{\sum_{i=1}^{N_R} \tau_i (c_{i0} + c_{i1}x_1 + \cdots + c_{in}x_n)}{\sum_{i=1}^{N_R} \tau_i}, \quad (2)$$

where τ_i is the firing strength of the rule R_i, which is defined as

$$\tau_i = A_{i1}(x_1) \times A_{i2}(x_2) \times \cdots \times A_{in}(x_n) \quad (3)$$

$\times$ operator in (3) represents *'fuzzy and'* operation and the algebraic product is used for this operation. In this paper, the membership function of an antecedent part has a Gaussian shape and is represented by the following equation.

$$\mu_{A_{ij}}(x_i) = \exp\left(\frac{-(x_i - w_{ij})^2}{\sigma_{ij}^2}\right), \quad (4)$$

where w_{ij} and σ_{ij} represent the center value and the width value of the Gaussian function, respectively.

2.2. Structure

To apply any evolutionary computation method for designing a fuzzy model, the structure of the fuzzy model should be determined. The structure is closely related to the representation scheme for evolutionary computations or other various evolutionary operators. We used the most general method to build a TSK fuzzy model. Once the number of membership functions for each input variable is determined, antecedents of fuzzy rules are obtained by combinations of each membership function of every input variable. If there are N_I input variables and M_i MFs for the i-th input, then the resulting fuzzy model has $\prod_{i=1}^{N_I} M_i$ fuzzy rules. Fig. 1 shows an example when there are 2 input variables, 2 MFs for the first input variable and 3 MFs for the second variable. The fuzzy model shown in Fig. 1 has 6 fuzzy rules. We call this method a 'partition-based' fuzzy model construction for convenience. Despite the fact that the number of fuzzy rules increases exponentially, this approach provides a simple and more intuitive design procedure because it is only necessary to determine the MFs for each input. Alternatively, there is also another popular fuzzy structure construction method when applying evolutionary computation techniques. In [2] and [6], different MFs have been used for each fuzzy rule, which we refer to as a 'rule-based' fuzzy model construction method for convenience.

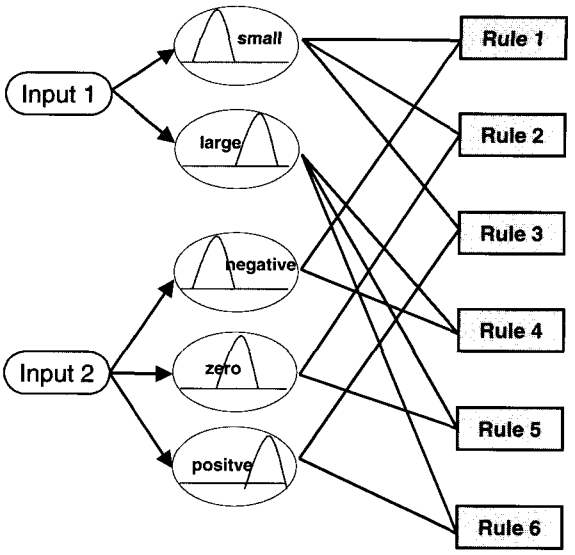


Fig. 1. The antecedent rule structure of the fuzzy model with 2 inputs and 2 MFs for the 1st input and 3 MFs for the 2nd input.

As stated in [1], the design of a fuzzy system can be formulated as a search problem in high-dimensional space. It is well-known that the parameters for the consequent parts can be obtained by various gradient methods or linear algebraic methods once the antecedent parts are determined. Also it is generally believed that it is more difficult to find parameters for the antecedent parts than to find parameters for the consequent parts. For these reasons, this paper and other works mentioned above are focusing on the design of the MFs of the antecedent parts and,

therefore, the dimension of the search space is determined only by the necessary number of tunable parameters in the antecedent parts. Also, one can see that the dimension of the search space has close relation to the complexity of the problem. We will briefly examine how the dimension of the search space (i.e., complexity of the problem) varies according to both of the two approaches.

When using a partition-based method, for example, if there are 4 inputs, 2 MFs for each input variable, and 2 tunable parameters for each MF, then this method builds a fuzzy model with 16 fuzzy rules and the necessary number of tunable parameters is 16 ($4\times2\times2$). If the number of MFs is increased from 2 to 3, then the number of fuzzy rules and the necessary number of tunable parameters becomes 81 and 24, respectively. It can be seen that the necessary number of tunable parameters is not increasing exponentially. That means that the dimension of the search space is not increasing exponentially.

In rule-based methods, however, enlargement of the search space may result even if the resulting fuzzy model has a less number of fuzzy rules. When one adopts the methods that are similar to the ones proposed in [2] and [6], if the number of fuzzy rules of the resulting fuzzy model is generously allowed to be 5, then this number of fuzzy rules does not change even if the number of MFs is changed. The necessary number of tunable parameters is 40 (2 parameters/MF × 4 MFs/rule × 5 rules), which is greater than what is necessary for the 'partition-based' model construction method. If the number of rules is increased, then the dimension of the search space becomes larger. The dimension of the search space is directly related to the complexity of the problem. For the same dilemma, if the complexity increases, then it may become more difficult to determine a solution. For example, one can easily see that it is easier to find 16 optimal parameters than to find 40 optimal parameters using evolutionary computations or other search methods. Moreover, when using a rule-based approach, there can be some cases where there is no corresponding fuzzy rule or fuzzy membership function for the particular input state. This may reduce the generalization ability of the resulting fuzzy model. As discussed above, both approaches have their advantages and disadvantages. From the viewpoint of the dimension of the search problem, however, we think the membership function-based fuzzy model construction method is preferable and we have adopted this method in our paper.

2.3. Calculating weight parameters

The aim of this subsection is to develop a procedure to obtain the consequent parameters c_{ij} in (2). Let the number of input variables be N_I, the number of fuzzy rules be N_R and then we can rewrite (2) in the following form:

$$y=\sum_{i=1}^{N_R} f_i(c_{i0}+c_{i1}x_1+\cdots+c_{iN_I}x_{N_I}), \tag{5}$$

where

$$f_i \equiv \tau_i / \sum_{j=1}^{N_R}\tau_j \tag{6}$$

is the normalized firing strength for the i-th rule. As stated in [7], (5) is nonlinear in the parameters. However, if the parameters in the antecedent MFs are fixed at the beginning of model construction (which is the case in this paper as will be addressed in Section III) so that the only free parameters are those in the linear regression equation, then (5) is linear in the parameters. Thus, once the antecedent MFs have been fixed, the consequent parameters c_{ij} in (2) can be calculated using some algebraic manipulation such as least square optimization.

A data pattern can be represented as follows:

$$z=[\boldsymbol{x}, y], \tag{7}$$

where $\boldsymbol{x}=[x_1, x_2, \cdots, x_{N_I}]$ is an input pattern and y is an output pattern. If we rewrite the right part of (5) in a matrix form, it becomes

$$C_f \bar{\boldsymbol{x}}, \tag{8}$$

where $\bar{\boldsymbol{x}}=[1, x_1, x_2, \cdots, x_{N_I}]^T \in R^{(N_I+1)\times 1}$ and

$$C_f=\begin{bmatrix} f_1c_{10} & f_1c_{11} & \cdots & f_1c_{1N_I} \\ f_2c_{20} & f_2c_{21} & \cdots & f_1c_{2N_I} \\ \vdots & \vdots & \vdots & \vdots \\ f_{N_R}c_{N_R0} & f_{N_R}c_{N_R1} & \cdots & f_{N_R}c_{N_RN_I} \end{bmatrix} \in R^{N_R\times(N_I+1)}. \tag{9}$$

With some matrix manipulations, (8) can be rewritten into the following form:

$$y=\bar{\boldsymbol{f}}C, \tag{10}$$

where

$$\bar{\boldsymbol{f}}=[f_1\ f_1x_1\ \cdots f_1x_{N_I}\ \cdots f_{N_R}x_1\ \cdots f_{N_R}x_{N_I}] \in R^{1\times[N_R\times(N_I+1)]} \tag{11}$$

and

$$C=[c_{10}\ c_{11}\ \cdots\ c_{1N_I}\ \cdots\ c_{N_R0}\ c_{N_R1}\ \cdots\ c_{N_RN_I}] \in R^{[N_R\times(N_I+1)]\times 1} \quad (12)$$

is the weight parameter matrix. Notice that the above equations correspond to the case when only a single data pattern is engaged. Assuming there are N_T data patterns available, that is $Z=[z_1\ z_2\ \cdots\ z_{N_T}]$, then we can construct the matrix

$$F=[\bar{f}(1)\bar{f}(2)\ \cdots\ \bar{f}(N_T)]^T \in R^{N^T\times[N_R\times(N_I+1)]}. \quad (13)$$

The number in the parentheses represents the index of the data pattern. If we consider the single-output for simplicity, then the desired output matrix is $Y=[y(1)\ y(2)\ \cdots\ y(N_T)] \in R^{N^T\times 1}$. Then (10) becomes

$$FC=Y. \quad (14)$$

Thus, the consequent parameter matrix C can be calculated using the pseudo-inverse method as follows:

$$C=(F^TF)^{-1}F^TY. \quad (15)$$

Once we determine the parameters for the antecedent parts of the fuzzy model, we can easily obtain the parameters for the consequent parts. In the next section, it will be explained how to obtain the parameters for the antecedent parts using an evolutionary algorithm.

3. EVOLUTIONARY DESIGN PROCESS

3.1. Representation

When designing a fuzzy model using an evolutionary algorithm, one of the most important considerations is the representation scheme, that is, how to encode the fuzzy system into the chromosome. Since the objective of the evolutionary optimization is to find an optimal value of centers and widths of the membership functions, it is only necessary to represent a MF by 2 real values. If there are M_i ($i=1,2,\cdots,N_I$) MFs for each input variable, then the total length of an individual (chromosome) becomes $L=\sum_{i=1}^{N_I}M_i$, which is the necessary number of tunable parameters. An individual is denoted as s_k where $k=1,2,\cdots,N_P$. N_P is the population size. An individual is represented in vector form as in the following:

$$s_k(t)=[v_{11}\ v_{12}\ \cdots\ v_{1M_1}\ \cdots\ v_{N_I1}\cdots v_{N_IM_{N_I}}], \quad (16)$$

where $v_{ij}=[w_{ij},\sigma_{ij}]$ represents the parameters of the j-th membership function for the i-th input variable and the number in the parenthesis represents the generation number.

3.2. Evaluation

Defining a proper fitness function is one of the most important issues when using an evolutionary approach since the fitness function guides the direction of the solution. To evaluate each individual when using the evolutionary approach, a fitness function that is appropriate for the given problem should be devised. The most common way to define a fitness function is to measure the performance of an individual in terms of the mean-squared-error (MSE)

$$MSE(s_k)=\frac{1}{N_T}\sum_{h}^{N_T}(y_h-\hat{y}_h)^2, \quad (17)$$

where y_h is the h-th desired output and $\hat{y}_h$ is the h-th model output. Usually, the inverse of the MSE value is used to evaluate individuals in evolutionary algorithms. However, if only the MSE is adopted in evaluating individuals, then there can be multiple overlapping MFs, which is a typical feature of unconstrained optimization as noticed in [6] and [8]. If there is an individual that contains multiple overlapping MFs, but shows relatively good MSE value at the initial stage of the evolution, then the individual receives a high fitness value and will have a high probability of survival, which may degrade the performance of the entire evolution process in the end. As well, the resulting overlapping MFs lose their interpretability [14]. Furthermore, it may be likely that the performance of the resulting fuzzy model become worse, since the fact that the overlapping membership functions cannot distinguish itself from other MFs means that it lost its freedom of representation ability.

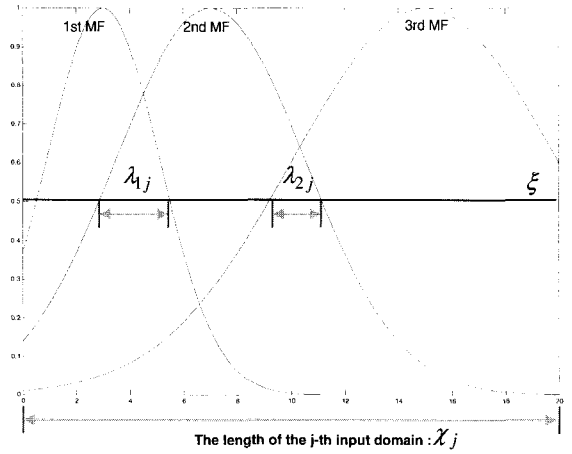


Fig. 2. Definition of the overlapping length in the penalty function.

To resolve this problem, constraints are defined in [6] so that the parameters of fuzzy MFs can vary only within a predefined range of their initial values. This approach, however, is somewhat strict in the sense that the multiple overlapping MFs can still be generated according to their initial positions. Also in [14], ε -constraints was used to restrict the overlapping. The constraints were strict and their method does not permit compromise between accuracy and interpretability of the fuzzy model. Thus, in this paper, a penalty function for an individual is defined. The proposed penalty function actively calculates the degree of overlapping between two MFs and is defined as the following:

$$PF(s_k) = \sum_{j=1}^{N_I} \frac{\sum_{i=1} \lambda_{ij}}{|\chi_j|}, \tag{18}$$

where λ_{ij} is the length of the i-th overlapping occurrence between two MFs in the j-th input domain. χ_j is the length of the j-th input domain. The specific level that constrains the overlapping between two MFs is denoted by ξ as shown in Fig. 2. Using this penalty value with the MSE value, the fitness function is defined as follows:

$$F(s_k) = \frac{1}{MSE(s_k) + \beta \cdot PF(s_k)}, \tag{19}$$

where β is a design parameter that is used to make a compromise between the MSE and the penalty function. With the proposed fitness function, one can avoid multiple overlapping MFs and the MF itself can be located anywhere on the input domain while not being restricted within the range that is determined by their initial positions. Since the proposed penalty function is not a strict constraint on individuals but rather evaluation criteria for individuals, the proposed algorithm can even construct a fuzzy model that has multiple overlapping membership functions if it is necessary to solve the given problem, which is not the case when restrictive constraints are used.

3.3. Evolutionary operators

3.3.1 Reproduction

According to the fitness value of each individual, we first apply a ranking method [9]. After ranking all the individuals in the population according to their fitness value, the upper 30% of the population is used to generate 50% of the new population. The remaining 70% of the population is used to generate 50% of the new population. Elitism was used to preserve the best individual. For better convergence performance, 30% of the new population is replaced by copying an elite individual into random locations. By hastening the convergence speed, we need less generations in the evolutionary process. However, since rapid convergence may result in finding local minima, we maintained the diversity of the population by other evolutionary operators as shown in the ensuing several paragraphs.

3.3.2 Crossover

Crossover is the process of exchanging portions of two 'parent' individuals. An overall probability is assigned to the crossover process, which is the probability that given two parents, the crossover operation will occur. For convenience, we rewrite (16) as $s_k(t) = [p_1\ p_2\ \cdots\ p_L]$ where p_i corresponds to v_{ij} in (16). We have used two types of crossover operations in this paper. The first is a bitwise crossover. When two parents $s_v(t)$ and $s_w(t)$ are selected for the crossover operation, changing point t is selected randomly within the range of an individual and swapping occurs as follows:

$$\begin{aligned} s_v(t+1) &= [v_1\ v_2\ \cdots\ v_k\ w_{k+1}\ \cdots\ w_L], \\ s_w(t+1) &= [w_1\ w_2\ \cdots\ w_k\ v_{k+1}\ \cdots\ v_L]. \end{aligned} \tag{20}$$

The next crossover is an arithmetic crossover operator, which produces children using a linear combination of two parents as follows:

$$\begin{aligned} s_v(t+1) &= \alpha \cdot s_v(t) + (1-\alpha) \cdot s_w(t), \\ s_w(t+1) &= \alpha \cdot s_w(t) + (1-\alpha) \cdot s_v(t), \end{aligned} \tag{21}$$

where the parameter α is generated randomly each time the arithmetic crossover is applied. The arithmetic crossover is applied to the center value and the width value. Generally, the value of α is in the range of [0, 1], which can only produce a value between two parameter values of the selected parents. This is, however, not desirable because the value of offspring cannot be outside the value of parents. This can reduce the diversity of the entire population. Therefore we changed the range of α to [-0.5, 1.5] so that the center of a membership function can be located not only inside of the values of two parents, but also outside. Since the center value of a membership function can be very distant from the effective input domain after several arithmetic crossover operations are applied (which is also not a desirable result), some boundary values are defined so that the center values can remain within the effective input domain. These values are determined according to the range of each input domain. In summary, the first crossover helps in changing the structure of fuzzy model, while the second crossover tries to tune the parameters of the membership functions.

3.3.3 Mutation

The next operation is mutation. Mutation consists of changing an element's individual value at random, often with a constant probability. Mutation is performed column-wise for every center value and every width value of all individuals as follows:

$$\theta(t+1) = \theta(t) + N(0, \rho), \tag{22}$$

where θ is a parameter value: the center or the width of a membership function and $N(\cdot,\cdot)$ represents normal distribution function, which generates a random value around the zero-mean with the variance parameter ρ. For both the center and the width, the different values of ρ are defined as ρ_{center} and ρ_{width}, respectively. As a result of the crossover operations and the mutation operations, the width can be less or equal to zero. To prevent this, a lower boundary value $\rho_{\min}$ for the width is set. Since we use the elite-based ranking method, which can speed up the convergence and can result in finding local minima, a rather high value was used for the mutation probability to maintain diversity.

3.4. Summary

Given the data pattern matrix Z, the maximum number of generation $G_{\max}$, the population size N_P, and the number of input variables N_I, we set the probabilities for the crossover operations and the mutation operations, ρ_{center}, ρ_{width}, $\rho_{\min}$ and the design parameters β and ξ. According to the range of each input variable, the boundary values are determined. The framework of evolutionary procedure used in this paper is as follows:

1) Generate an initial population $P(0) = [s_1(0)\ s_2(0) \cdots s_{N_P}(0)]$ at random and set $i = 0$.

2) Using the parameter values of each individual, construct a fuzzy model and calculate consequent parameters as addressed in Section II-3 for all individuals.

3) Evaluate every individual.

4) Apply evolutionary operators to obtain the next population $P(i+1)$.

5) $i = i+1$, return to step 2) if the $G_{\max}$ is not reached or the procedure is terminated

Because elitism is used in the proposed algorithm, the best fuzzy model is easily extracted when the entire procedure is over.

4. NUMERICAL EXAMPLES

In the following subsections, we applied the proposed algorithm to two kinds of problems: nonlinear dynamic plant modeling and a chaotic time-series prediction problem.

4.1. A nonlinear plant modeling problem

The aim of this subsection is to find a fuzzy model for a nonlinear dynamic plant using the proposed algorithm. We will consider the second-order nonlinear plant studied in [2,10,11].

$$y(t) = g(y(t-1), y(t-2)) + u(t), \tag{23}$$

where

$$g(y(t-1), y(t-2)) = \frac{y(t-1)y(t-2)(y(t-1)-0.5)}{1+y^2(t-1)+y^2(t-2)}. \tag{24}$$

The goal is to approximate the nonlinear component $g(y(t-1), y(t-2))$, which is usually called the 'unforced system' in control literature, using a TSK fuzzy model. The problem involves 2 inputs and 1 output example. As in [6], 400 simulated data points were generated from the plant model (23). With the starting equilibrium state (0, 0), 200 samples of training data were obtained using a random input signal $u(t)$ that is uniformly distributed in $[-1.5, 1.5]$. The remaining 200 samples of validation data were obtained using a sinusoidal input signal $u(t) = \sin(2\pi t/25)$. The resulting signals are shown in Fig. 3.

Parameter settings for the given problem are as follows: The population size N_P was 100. The maximum generation $G_{\max}$ was 100. Crossover probability was 0.6. Mutation probability was 0.3 for the center value and 0.7 for the width value. ρ_{center} and ρ_{width} were 0.05 and 0.05, respectively. β

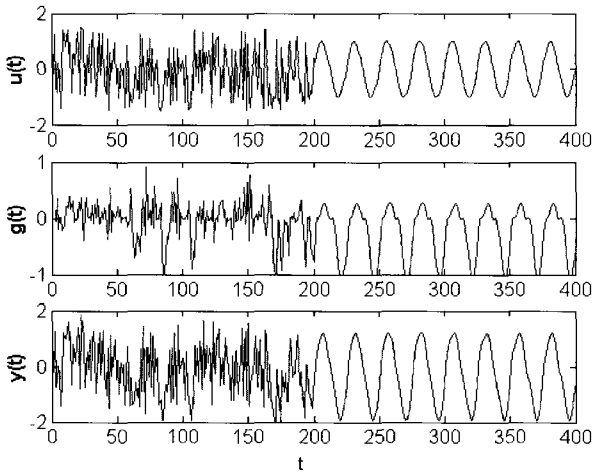


Fig. 3. Input u(t), unforced system g(t), and output y(t) of the plant in (23).

was 0.6 and ξ was 0.7. We applied the proposed algorithm with 3 different settings of the number of MFs. We used 2, 3 and 4 MFs for each input in each setting, respectively. A linear consequent TSK model was used. Because the number of inputs was 2, the necessary number of tunable parameters was 8 (2 inputs × 2MFs/input × 2 parameters/MF) when 2 MFs were used. Similarly, the necessary number of tunable parameters was 12 with 3 MFs and 16 with 4 MFs, respectively. We ran the simulation 10 times for each case and the obtained results are shown in Table I.

When 4 MFs are used, the minimum MSE (mean squared error) for both the training and the validation data were $1.5e^{-6}$ and $3.4e^{-6}$, respectively. The mean MSE for both the training and the validation date were $4.8e^{-6}$ and $1.13e^{-5}$, respectively. When 3 MFs are used, the minimum MSE for both the training and the validation data were $1.26e^{-5}$ and $1.2e^{-5}$, respectively. The mean MSE for both the training and the validation date were $2.4e^{-5}$ and $5.27e^{-5}$, respectively. When 2 MFs are used, the minimum MSE for both the training and the validation data were $1.41e^{-4}$ and $1.53e^{-4}$, respectively. The mean MSE for both the training and the validation data were $1.42e^{-4}$ and $1.79e^{-4}$, respectively.

Fig. 4 presents the desired and the obtained outputs $y(t)$ for both the learning and the validation data using the same $u(t)$. The difference between the two outputs is shown in Fig. 5. This result is adopted from the single simulation where the MSE for the training pattern is $1.5e^{-6}$ and the MSE for the validation pattern is $3.4e^{-6}$ with 4 MFs. The MFs for this single simulation are also shown in Fig. 6 and the parameter values are summarized in Table II.

To compare our results with those obtained by different approaches proposed in [6,10,11], the best results obtained in each case are summarized in Table 1. Also, we build a fuzzy model using AFNIS algorithm [16], which is one of the most well- known fuzzy modeling techniques, while changing the number of MFs for each input from 2 to 4, respectively. We obtained the best ANFIS model by searching the optimal number of iterations after iterating the algorithm 1000 cycles.

In the second row of Table I, the small value of MSE for the training data in contrast to the MSE for the evaluation data may indicate overtraining as pointed out in [6]. Since the necessary number of tuning parameters is directly related to the dimension of search space, a column for the necessary number of tunable parameters N_{TP} is added beside the MSE results.

From Table 1, one can easily see the effectiveness

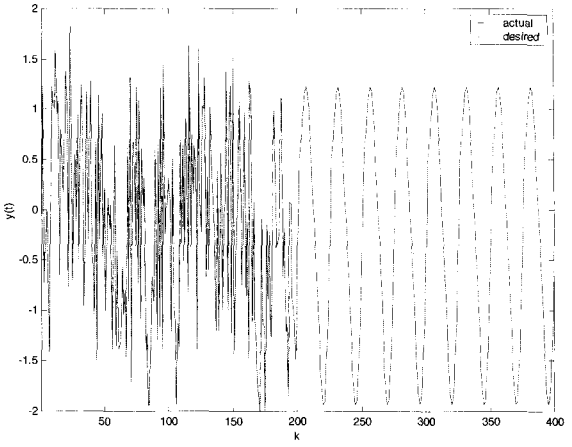


Fig. 4. Desired output and obtained output y(t) for both the learning and validation data using the same u(t) and the built fuzzy model.

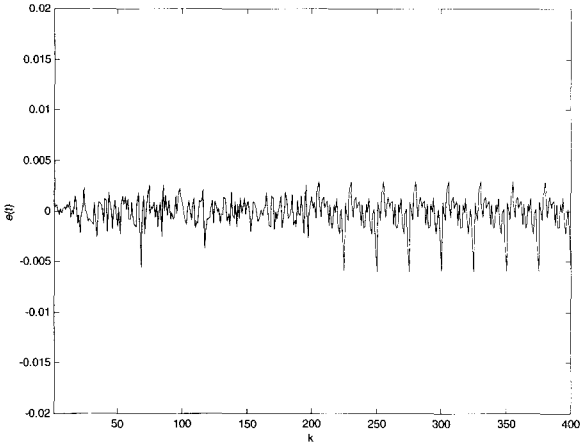


Fig. 5. Residual errors between the desired and the obtained output for both the learning and validation data.

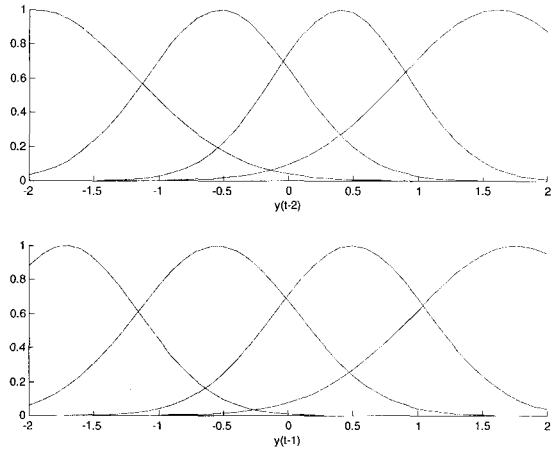


Fig. 6. An example of the obtained membership functions for both y(t-1) and y(t-2) by the proposed algorithm.

of the proposed algorithm. With the small number of necessary tunable parameters, the proposed algorithm

Table 1. Comparison results for the nonlinear system modeling problem.

Methods	No. rule	N_{TP}	MSE(train)	MSE(val)
[10]	24	-	2.0e(-6)	6.4e(-4)
[11]	25	50	2.3e(-4)	4.1e(-4)
	20	40	6.8e(-4)	2.4e(-4)
[6]	5	30	5.8e(-3)	2.5e(-3)
	5	24	7.5e(-4)	3.5e(-4)
	4	12	1.2e(-3)	4.7e(-4)
ANFIS[16]	4	4	1.54e(-4)	2.0e(-4)
	9	12	1.33e(-4)	1.65e(-4)
	16	16	5.59e(-5)	5.76e(-4)
Proposed With 2MFs	4	4	min. 1.40e(-4) mean. 1.42e(-4)	1.53e(-4) 1.79e(-4)
Proposed With 3MFs	9	12	min. 1.26e(-5) mean.2.40e(-5)	1.2e(-5) 5.27e(-5)
Proposed With 4MFs	16	16	min. 1.50e(-6) mean. 4.80e(-5)	3.4e(-6) 1.13e(-5)

Table 2. An example of the obtained parameter values for the MFs for the (23) when 4MFs are used.

y(t-1)	center	1.758	-1.717	-0.548	0.489
	width	1.098	0.797	0.873	0.837
y(t-2)	center	1.611	-0.526	0.402	-1.962
	width	1.056	0.805	0.738	1.100

can find a TSK fuzzy model that can effectively approximate the given plant with fewer errors. The more the number of MFs increases, the more the proposed algorithm becomes effective. Since the ANFIS model uses hybrid gradient methods, it can finally drive the center of the MF into the undesirable position as shown in Fig. 7, or it can make the width of the MF almost negligible as shown in Fig. 8. Fig. 7 was the case for which 3 MFs were used and Fig. 8 was the case for which 4 MFs were used for the ANFIS algorithm. In this case, the iteration was 1000. Meanwhile, the proposed algorithm can find a set of MFs that does not lose the interpretability as shown in Fig. 6. This is mainly due to the proposed fitness function. To see the effect of the penalty function more clearly, we separately run the proposed algorithm with the same settings. Only the β was set to 0, which means no penalty constraints were imposed. Figs. 9 and 10 show the obtained MFs without using the penalty function. As shown in the figures, the resulting MFs overlap each other, signifying that the resulting MFs lose their interpretability. Also the performance of the fuzzy model becomes inferior, since the fact that the overlapping MFs cannot distinguish themselves from other MFs, signifying a loss of freedom of representation ability. However, with the help of penalty constraints, the proposed algorithm can find non-overlapping MFs as shown in Fig. 6.

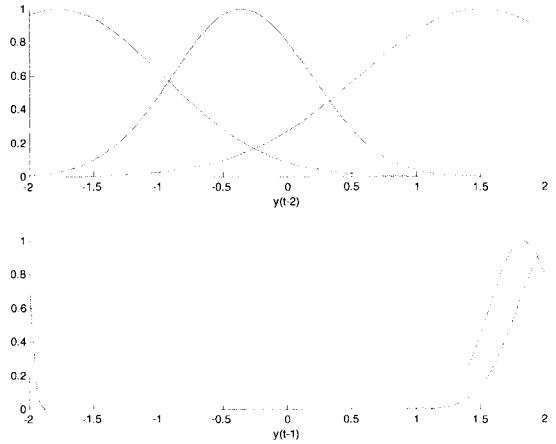


Fig. 7. An example of the obtained MFs for both y(t-1) and y(t-2) by the ANFIS algorithm with 3 MFs.

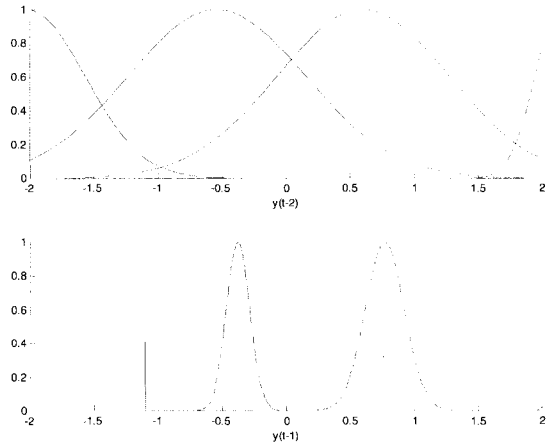


Fig. 8. An example of the obtained MFs for both y(t-1) and y(t-2) by the ANFIS algorithm with 4 MFs.

4.2. Time series prediction problem

A time-series prediction problem based on the chaotic Mackey-Glass differential equation [12] is one of the most famous benchmarks for comparing the various abilities of a fuzzy system and neural networks. A time series is generated using the following equation.

$$\frac{dx(t)}{dt} = -b \cdot x(t) + \frac{a \cdot x(t-\tau)}{1 + x^{10}(t-\tau)}, \tag{25}$$

where $b = 0.1, a = 0.2$, and $\tau = 17$ as in [2] and [13]. The task of a fuzzy model is to predict the value of the time series $\hat{x}(t+T)$ from the knowledge of n samples of the previous time series as in the following equation [2]:

$$\hat{x}(t+T) = f_{pr}(x(t), x(t-\Delta), \cdots, x(t-(k-1)\Delta), \tag{26}$$

where Δ is the lag time and k is an embedding

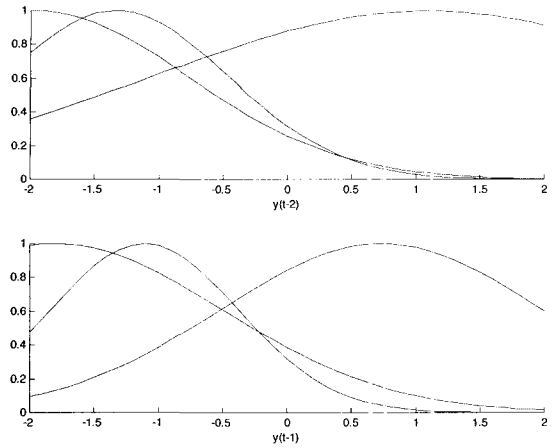


Fig. 9. An example of the obtained MFs for the plant (23) without PF(s) (i.e., $\beta = 0$), when 3 MFs are used for the proposed algorithm.

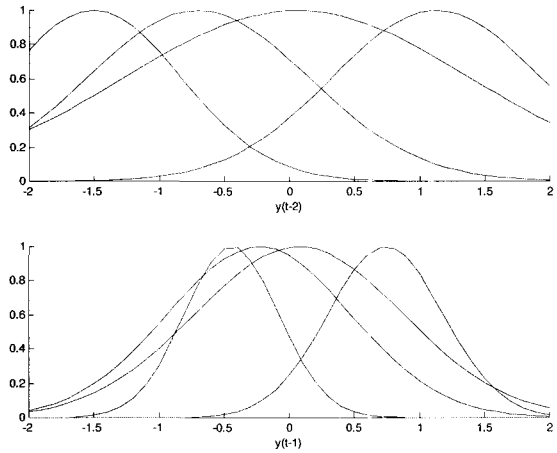


Fig. 10. An example of the obtained MFs for the plant (23) without PF(s) (i.e., $\beta = 0$), when 4 MFs are used for the proposed algorithm.

dimension. To make comparisons, $k = 4$ and $\Delta = T = 6$ were chosen. The data points were generated with the initial condition $x(0) = 1.2$. The 1000 patterns used in the training and in the validation phase have the following format $[x(t-24); x(t-18); x(t-12); x(t-6); x(t)]$ from the generated data with t in $[123, 1123]$. The first 500 patterns were used in the training phase, whereas the last 500 were used in the validation phase. The problem involves 4 inputs and 1 output example.

The population size N_P was 100. The maximum generation $G_{\max}$ was 300. Crossover probability was 0.7. Mutation probability was 0.6 for the center value and 0.6 for the width value. β was 0.6 and ξ was 0.6. A linear consequent TSK model is used. Because the number of inputs is 4, the necessary number of tunable parameters was 16 (4 inputs × 2 MFs/input × 2 parameters/MF). We ran the simulation 10 times and averaged the results.

The minimum RMSE (root mean squared error) for both training and validation data were 0.0014 and 0.0013, respectively. The mean RMSE for both training and validation date were 0.0015 and 0.0014, respectively. Figs. 11 and 12 indicate the desired and the obtained outputs for learning data and validation data, respectively. Because no visible difference is shown in Figs. 11 and 12, the difference between the two outputs is shown in Figs. 13 and 14. The obtained MFs are presented in Fig. 15 and the values of the obtained parameters are summarized in Table 3.

Table 3 presents comparison results of the prediction performance for the validation data among various predictors. Since the result by [13] outperforms other approaches such as cascade neural network, autoregressive model, etc, we only adopted the result by [13]. For comparison, we also trained a RBF model, which is not included in the work of [13], with 100, 200 and 500 hidden neurons. K-means clustering and gradient method was used to obtain optimal value for the center, width and output weights of RBF networks [17]. When we use 500 hidden neurons, the RBF model perfectly approximates the training data. However, the error for the validation data becomes worse because of overfitting. In fact, the best results reported in literature came from the GERFEX (genetic fuzzy rule extractor), which requires 20 rules. We can actually obtain an ANFIS model of which performance is better than that reported in [13] and [2]. Gaussian MFs and product operation for fuzzy reasoning was used for the ANFIS

Table 3. An example of the obtained parameter values for the MFs for the Mackey-Glass time series prediction problem.

x(t-6)	center	0.596	1.4
	width	0.348	0.3
x(t-12)	center	0.4	1.233
	width	0.544	0.303
x(t-18)	center	0.802	1.385
	width	0.247	0.554
x(t-24)	center	0.759	1.4
	width	0.527	0.244

Table 4. Comparison results for the nonlinear time series prediction problem.

Methods	RMSE
Kim and Kim [13]	0.026
GERFEX [2]	0.0061
RBF network (100)	0.0054
RBF network (200)	0.0036
RBF network (500)	0.0336
ANFIS (16)	0.0045
Proposed algorithm	min. 0.0013 mean. 0.0014

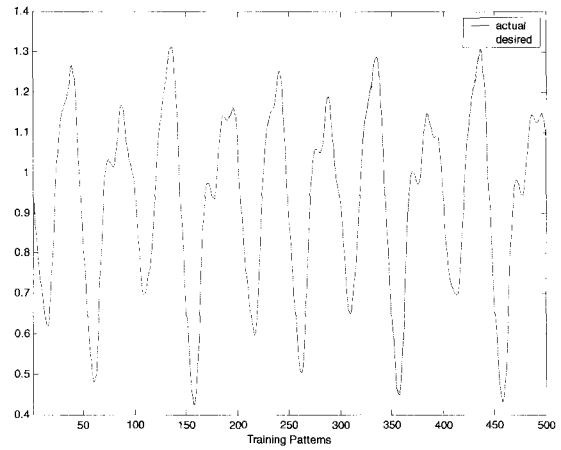


Fig. 11. Desired and obtained output for Mackey-Glass problem (learning).

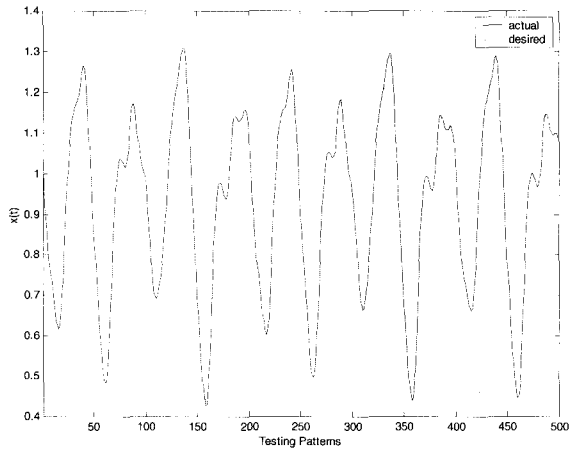


Fig. 12. Desired and obtained output for Mackey-Glass problem (validation).

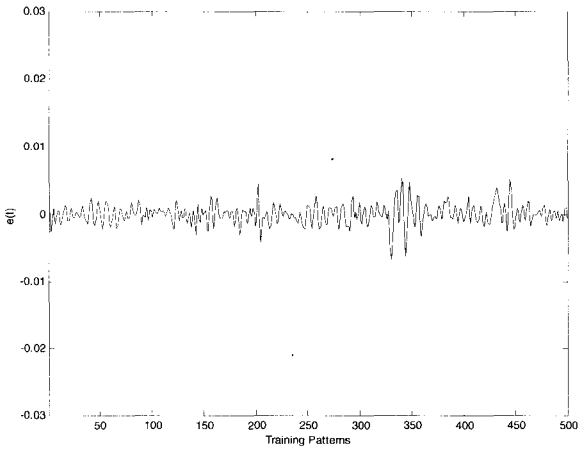


Fig. 13. Residual error for Mackey-Glass problem (learning) RMSE = 0.0014.

model. Although N_{TP} (the necessary number of tunable parameters) does not obviously appear in [2], we can infer the necessary number of tunable parameters from the fact that a) the GERFEX uses N_I MFs in each fuzzy rule to the maximum, b) each

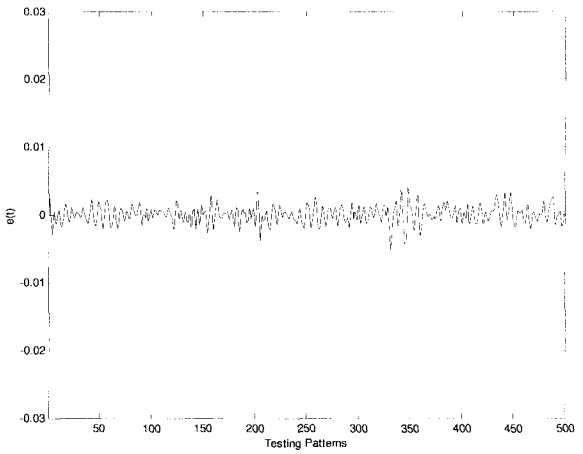


Fig. 14. Residual error for Mackey-Glass problem (validation) RMSE = 0.0013.

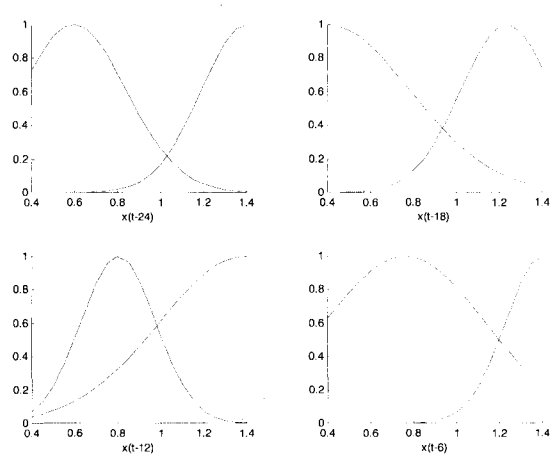


Fig. 15. An example of the obtained MFs for the Mackey-Glass time series prediction problem.

fuzzy rule requires different fuzzy MFs and c) each MF uses 2 parameters: center and width. The necessary number of tunable parameters may be about 160, which is much greater than that of the proposed algorithm in this paper, which is only 16. Also $G_{\max}$ in [2] was 50,000, whereas $G_{\max}$ in the proposed algorithm was 300, which proves the fact that the necessary update to find an optimal solution is greatly reduced due to the reduction of the search space (i.e., the reduction of the complexity of the problem). It is more difficult to find 160 optimal parameters than to find 16 optimal parameters. The performance of the proposed algorithm also surpasses all previous works. It can be said that the proposed algorithm achieved superior performance with a smaller number of tunable parameters. Also in GERFEX, since different membership functions are used in each fuzzy rule, the resulting fuzzy model loses its interpretability.

In summary, it can be said that the proposed algorithm improved both the prediction capability and the compactness of the fuzzy system with a smaller number of parameters while not losing its interpretability.

5. CONCLUSIONS

An evolutionary design process of the TSK type fuzzy model is studied in this paper. The occurrence of the multiple overlapping membership functions, which is a typical feature of unconstrained optimization, is resolved with the help of the newly defined fitness function. This was accomplished while preserving the possibility of adapting the multiple overlapping membership functions if deemed necessary to solve the given problem. It also improves the interpretability of the resulting fuzzy model by actively maintaining a certain degree of overlapping between membership functions. Through the evolutionary optimizations, the values of the antecedent parameters are identified and the identification of the consequent parameters is made compactly with the data pattern manipulations and the pseudo inverse method. By taking a partition-based approach to building a TSK fuzzy model, the proposed algorithm provides a simple and more intuitive design procedure and even results in better performance capability.

To demonstrate the effectiveness of the proposed algorithm, several simulations both for the nonlinear dynamic system modeling and for a chaotic time series prediction problem have been performed. It was shown that the proposed algorithm found a TSK fuzzy model that produces a smaller error than established in previous works with a smaller number of necessary tunable parameters.

REFERENCES

[1] Y. Shi, R. Eberhart, and Y. Chen, "Implementation of evolutionary fuzzy systems," *IEEE Trans. on Fuzzy Syst.*, vol. 7, no. 2, pp. 109-119, 1999.

[2] M. Russo, "Genetic fuzzy learning," *IEEE Trans. on Evolutionary Computation.*, vol. 4, no. 3, pp. 259-273, 2000.

[3] O. Cordó n and F. Herrera, "A two-stage evolutionary process for designing TSK fuzzy rule-based systems," *IEEE Trans. on Syst., Man, Cybern., pt, B*, vol. 29, pp. 703-715, December 1999.

[4] S. J. Kang, H. S. Hwang, and K. N. Woo, "Evolutionary design of fuzzy rule base for nonlinear system modeling and control," *IEEE Trans. on Fuzzy Syst.*, vol. 8, pp. 37-45, February 2000.

[5] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. on Syst., Man, Cybern.*, vol. 15, pp. 116-132, 1985.

[6] M. Setnes and H. Roubos, "GA-fuzzy modeling and classification: Complexity and performance," *IEEE Trans. on Fuzzy Syst.*, vol. 8, pp. 509-522, October 2000.

[7] J. Yen, L. Wang, and C. W. Gillespie, "Improving the interpretability of TSK fuzzy models by combining global learning and local learning," *IEEE Trans. on Fuzzy. Syst.*, vol. 6, pp. 530-537, November 1998.

[8] M.-S. Kim and J.-J. Lee, "Evolutionary design of takagi-sugeno type fuzzy model for nonlinear system identification and time series prediction," *Proc. on of the International Conference on Control, Automation and Systems,* Cheju Univ. Korea, pp. 667-670, October 2001.

[9] Z. Michalewicz, *Genetic algorithms + Data Stuructures = Evolution Programs*, Springer, New York, 1999.

[10] J. Yen and L. Wang, "Application of statistical information criteria for optimal fuzzy construction," *IEEE Trans. on Fuzzy Syst.*, vol. 6, pp. 362-371, August 1998.

[11] J. Yen and L. Wang, "Simplifying fuzzy rule-based models using orthogonal transformation methods," *IEEE Trans. on Syst., Man, Cybern., pt, B*, vol. 29, pp. 13-24, February 1999.

[12] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," *Science,* vol. 197, pp. 287-289, 1977.

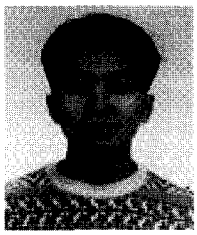
[13] D. Kim and C. Kim, "Forecasting time series with genetic fuzzy predictor ensemble," *IEEE Trans. on Fuzzy Syst.*, vol. 5, pp. 523-535, November 1997.

[14] M. R. Delgado, F. V. Zuben, and F. Gomide, "Multi-objective decision making towards improvement of accuracy interpretability and design autonomy in hierarchical genetic fuzzy systems," *FUZZ-IEEE'02. Proc. of the 2002 IEEE International Conference on Fuzzy Systems,* vol. 2, pp. 1222-1227, 2002.

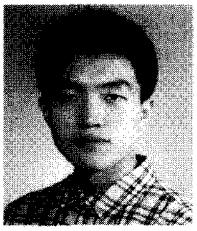
[15] Y. Jin, "Fuzzy modeling of high-dimensional systems: Complexity reduction and interpretability improvement," *IEEE Trans. on Fuzzy Syst.*, vol. 8, no. 2, pp. 212-221, 2000

[16] J.-S.R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. on Syst., Man, Cybern.*, vol. 23, no. 3, pp. 665-685, 1993.

[17] Simon S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1998.



Min-Soeng Kim received his B.S. degree in Electronic Engineering from HanYang University, Seoul, Korea, in 1997 and his M.S. degree in Electrical Engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejon, Korea, in 1999, where he is also currently pursuing his Ph.D. degree. His research interests include artificial intelligence, fuzzy modeling, and evolutionary optimization.



Chang-Hyun Kim received his B.S. and M.S. degrees in Electrical and Electronic Engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 2000 and 2002, respectively. As of 2002, he has been working toward his Ph.D. degree at KAIST. His research interests include artificial intelligence, intelligent control, and robotics. He is a member of ICASE.

Ju-Jang Lee (M'86–SM'98) was born in Seoul, Korea, on November 14, 1948. He received his B.S. and M.S. degrees, both in Electrical Engineering, from Seoul National University in 1973 and 1977, respectively, and his Ph.D. degree in Electrical Engineering from the University of Wisconsin in 1984. From 1977 to 1978, he was a Research Engineer at the Korean Electric Research and Testing Institute, Seoul. From 1978 to 1979, he was a Design and Processing Engineer at G. T. E. Automatic Electric Co.,Waukesha, WI. For a brief period in 1983, he was the Project Engineer for the Research and Development Department of the Wisconsin Electric Power Co., Milwaukee, WI. He joined the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejon, Korea, in 1984, where he is currently a Professor. In 1987, he was a Visiting Professor at the Robotics Laboratory of Imperial College Science and Technology, London, U.K. From 1991 to 1992, he was a Visiting Scientist at the Robotics of Carnegie Mellon University, Pittsburgh, PA. His research interests are in the areas of intelligent control of mobile robots, service robotics for the disabled, space robotics, evolutionary computation, variable structure control, chaotic control systems, electronic control units for automobiles, and power system stabilizers. Dr. Lee is a Senior Member of IEEE and a Member of the IEEE Robotics and Automation Society, the IEEE Evolutionary Computation Society, the IEEE Industrial Electronics Society, KIEE, KITE, and KISS. He is also a Vice President of ICASE and a Director of SICE in Japan.