Fuzzy Modeling of High-Dimensional Systems: Complexity Reduction and Interpretability Improvement

Yaochu Jin, *Member, IEEE*

***Abstract*—Fuzzy modeling of high-dimensional systems is a challenging topic. This paper proposes an effective approach to data-based fuzzy modeling of high-dimensional systems. An initial fuzzy rule system is generated based on the conclusion that optimal fuzzy rules cover extrema [8]. Redundant rules are removed based on a fuzzy similarity measure. Then, the structure and parameters of the fuzzy system are optimized using a genetic algorithm and the gradient method. During optimization, rules that have a very low firing strength are deleted. Finally, interpretability of the fuzzy system is improved by fine training the fuzzy rules with regularization. The resulting fuzzy system generated by this method has the following distinct features: 1) the fuzzy system is quite simplified; 2) the fuzzy system is interpretable; and 3) the dependencies between the inputs and the output are clearly shown. This method has successfully been applied to a system that has 11 inputs and one output with 20 000 training data and 80 000 test data.**

***Index Terms*—Complexity reduction, fuzzy modeling, interpretability, optimization, regularization.**

I. Introduction

EXPERT knowledge-based and data-driven rule generations are two main approaches to fuzzy modeling. Recently, data-based rule generation has been widely investigated and shown to be very successful. Despite that, conventional fuzzy models suffer from combinatorial rule explosion; in other words, the model complexity grows exponentially with the input dimension. To deal with this problem, several methods have been developed. One popular method is to replace the conventional fixed or adaptive fuzzy grid with a more flexible k-d tree [1], which was originally suggested to storage data. Alternatively, quad trees [2] have also been adopted for input space partition. Although the former is more flexible than the later, it has often a lot of nonuniform overlapping membership functions to which it is hard to assign an understandable linguistic term. Since all the input variables are not necessarily interactive, an iterative construction algorithm involving building and pruning has been developed in [3]. A hierarchical rule structure, where the number of fuzzy rules grows linearly with the input number, is proposed in [4]. Evolutionary algorithms have also been used as a new tool for building compact fuzzy systems. Both genetic algorithms [5] and evolution strategies [6] have been introduced to reduce the complexity of fuzzy rules. Apart from the above efforts on an efficient rule structure, use of interpolation [7] and optimal rules that cover the extrema [8] are another two interesting ideas for rule reduction. Recently, it is suggested that fuzzy systems with a union-rule configuration can reduce the rule complexity effectively [9]. Techniques of deleting redundant, inconsistent and inactive rules after rule generation are also very useful.

One of the most important issues in data-driven fuzzy rule generation is interpretability (also called transparency) of the fuzzy system. It is well known that one of the most important motivations to use a fuzzy system for system modeling is that a fuzzy system is easy to understand for human beings. However, interpretability of a fuzzy system might be lost for adaptive fuzzy systems using data-based learning. In [10] and [11], interpretability of fuzzy systems from the view point of fuzzy membership functions are discussed. In [10], interpretability is maintained by loosely limiting the location of the membership functions during learning. Alternatively, similar fuzzy membership functions are merged in [11] so that the resulting fuzzy partitions are interpretable. Interpretability of the consequent part of the Takagi–Sugeno fuzzy rules is considered in [12] through local learning.

The purpose of this paper is twofold. On the one hand, we try to develop a practical method for modeling high-dimensional systems with a simplified fuzzy system. On the other hand, we hope that the resulting fuzzy system is easy to understand. To this end, an initial fuzzy rule base is first generated by taking advantage of the conclusion that optimal fuzzy rules cover the extrema [8]. In rule generation, a fuzzy similarity measure is adopted to check the similarity of each rule before putting it into the rule base to avoid rule redundancy. Then both the structure and parameters of the fuzzy rules are optimized using a genetic algorithm and a gradient learning algorithm. The structure of the rule premise (i.e., which inputs should appear in the premise part of a fuzzy rule) is optimized using a genetic algorithm based on a local cost function, which is consistent with the motivation to establish local optimal fuzzy rules. It is shown that optimization of the rule structure cannot only reduce the rule complexity and improve the system performance, but also reveal the dependencies between the system inputs and the system output. After structure optimization, the parameters of the fuzzy rules are optimized by a gradient learning algorithm. During learning, fuzzy rules with a very low firing strength are deleted, which plays an important role in reducing the number of fuzzy rules. Finally, interpretability of the fuzzy system is improved by fine tuning the fuzzy rules with regularization.

This method has been applied to an example with 11 input variables and one output variable. There are 100 000 data sets in total, among which 20 000 sets are for training and the rest are for test purpose only. The resulting fuzzy system is very inspiring. The final system contains only 27 fuzzy rules and its performance on both training and test data is satisfactory.

Manuscript received April 20, 1999; revised November 19, 1999.

The author is with the Future Technology Research Division, Honda R&D Europe GmbH, Offenbach/Main 63073 Germany.

Publisher Item Identifier S 1063-6706(00)03203-3.

In Section II, we describe how to generate a compact, locally optimal fuzzy rule base from the given data. Section III introduces the structure and parameter optimization based on genetic algorithms and the gradient method. The algorithm to improve interpretability of the fuzzy system is presented in Section IV. An application example is given in Section V, which shows that the proposed method works very effectively. Section VI concludes this paper with a summary of the work.

II. Generation of Local Optimal Rules

A. Rule Generation on Extrema

In [13], an approach to generating fuzzy rules from data has been proposed. In the method, each input variable is partitioned into a number of fuzzy subsets beforehand and fuzzy rules are generated based on these fuzzy subsets. The method works effectively when the number of data sets is limited. However, if the number of available data sets is huge, the number of generated fuzzy rules will increase tremendously. This gives rise to a serious problem because a fuzzy rule base with a huge number of fuzzy rules is undesirable. Besides, the heuristic fuzzy partitions for the input variables may not be optimal. In fact, it is not necessary to generate a fuzzy rule for each data set. As is shown in [8], optimal fuzzy rules cover extrema. This is a very interesting conclusion that enables us to reduce the number of fuzzy rules greatly when we generate fuzzy rules from data.

Suppose the total number of given data sets is N. If we divide the whole data sets into M patches, the number of data sets in each patch will be $P = N/M$, assuming that P is an integer. The patch size can be adjusted properly based on the smoothness of the output data. Normally, when the output varies rapidly, the patch size should be smaller; on the other hand, the patch size can be a little larger if the output surface is smooth.

For each patch of the data, we find the two data sets that have the minimal and maximal outputs respectively. Suppose for patch j, the two data sets are

$$(x_{j1}^{\max}, x_{j2}^{\max}, \cdots, x_{jn}^{\max}, y_j^{\max})$$
$$(x_{j1}^{\min}, x_{j2}^{\min}, \cdots, x_{jn}^{\min}, y_j^{\min})$$

where n is the dimension of the input. For these two data sets, two Sugeno–Takagi fuzzy rules can be generated

R_{j1}: If x_1 is $A_{j1}^{\max}(x_1), \cdots, x_n$ is $A_{jn}^{\max}(x_n)$, Then y is $y_j^{\max}$;

R_{j2}: If x_1 is $A_{j1}^{\min}(x_1), \cdots, x_n$ is $A_{jn}^{\min}(x_n)$, Then y is $y_j^{\min}$

where A_{ji} are fuzzy sets whose membership function can be described by

$$A_{ji}^{\max}(x_i) = \exp\Big(-\big(x_i - x_{ji}^{\max}\big)^2 \Big/ \delta_{x_i}^{\max}\Big) \tag{1}$$

$$A_{ji}^{\min}(x_i) = \exp\Big(-\big(x_i - x_{ji}^{\min}\big)^2 \Big/ \delta_{x_i}^{\min}\Big) \tag{2}$$

where $\delta_{x_i}^{\max}$ and $\delta_{x_i}^{\min}$ is the width of the Gaussian function, which can be selected on the basis of the amplitude of the variable. In this way, $2P$ fuzzy rules will be generated in total, which will be much smaller than N, because usually, the patch size could be ten or larger. Fig. 1 illustrates the two points A and B within patch j, on each of which one fuzzy rule will be generated.

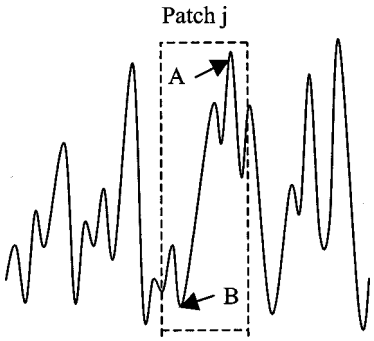


Fig. 1. Illustration of a data patch and its extrema.

B. Similarity Checking

Usually there would be a lot of redundant or sometimes inconsistent rules if the generated fuzzy rules are not properly checked. Redundant or inconsistent rules are generated when the inputs of different rules are quite similar. If the consequent parts of these rules are also similar, the rules may be redundant. If, usually due to noises in the data, the consequent parts are significantly different, inconsistent rules will be generated. To prevent the algorithm from generating fuzzy rules that are redundant or inconsistent, a similarity measure for fuzzy rules needs to be developed. In this work we use a measure called similarity of rule premise (SRP) proposed in [6]. Consider the following two fuzzy rules:

R_i: If x_1 is $A_{i1}(x_1), x_2$ is $A_{i2}(x_2), \cdots, x_n$ is $A_{in}(x_n)$, Then y is $B_i(y)$;

R_k: If x_1 is $A_{k1}(x_1), x_2$ is $A_{k2}(x_2), \cdots, x_n$ is $A_{kn}(x_n)$, Then y is $B_k(y)$

the SRP of the two rules is defined by

$$\mathrm{SRP}(i,k) = \min_{j=1}^{n} S(A_{ij}, A_{kj}) \tag{3}$$

where $S(A,B)$ is a fuzzy similarity measure for fuzzy sets A and B, which is defined by

$$S(A,B) = \frac{M(A \cap B)}{M(A) + M(B) - M(A \cap B)} \tag{4}$$

where $M(A)$ is the size of fuzzy set A

$$M(A) = \int_{-\infty}^{+\infty} A(x)\, dx. \tag{5}$$

Please refer to [6] for the details for computing the fuzzy similarity measure.

By checking the SRP of the fuzzy rules, redundant and inconsistent rules can be removed. In this way, the rule base can be simplified greatly.

III. Structure and Parameter Optimization

A. Local Structure Optimization

Structure optimization for fuzzy rule systems is very important for complexity reduction and performance enhancement. In a conventional fuzzy system, each rule contains all the input variables in its premise. In practice, it is found that such a rule system is hard to simplify and the system performance is not satisfactory. In addition, as the number of variables increases in the rule premise, it becomes harder and harder for human beings to understand the rule. To deal with this problem, we introduce genetic algorithms (GA's) to optimize the structure of the rule base.

GA's are a class of stochastic algorithms for adaptive systems and optimization [14]. The GA usually consists of three main operations, namely, selection, reproduction, and mutation, which are inspired from natural process of evolution. In optimization, the objective parameters are encoded into a chromosome or an individual using direct or indirect genetic coding. A number of individuals representing the initial solution for the parameters are then generated randomly for the initial population $P(t)$. Each individual in population $P(t)$ is then evaluated with a performance index called fitness. Based largely on the principle of "survival of the fittest," a number of individuals in population $P(t)$ are selected to produce new individuals using crossover. The resulting new individuals are called offspring that form a new population $P(t+1)$. Sometimes, the individual(s) with the best fitness will be directly passed to the population $P(t+1)$ (*elitism*). To emulate genetic mutations in the natural evolution, mutation is implemented with a small probability on the individuals in population $P(t+1)$. This process continues until a termination condition is met and the individual with the best fitness in the final population is the solution to the problem.

The genetic coding for fuzzy rule structure optimization is very straightforward. Suppose we are to optimize a fuzzy system consisting of N Takagi–Sugeno fuzzy rules. The total number of inputs is n. Therefore, there will be $n \times N$ genes in the chromosome, each consisting of only one binary bit "0" or '1." A "0" denotes that the input variable does not appear in the corresponding fuzzy rule and "1" means does. Assume there are five input variables in total and the piece of code for the jth fuzzy rule is "10 101," then the fuzzy rule looks like

$$R_j: \quad \text{If } x_1 \text{ is } A_{1j}, x_3 \text{ is } A_{3j}, x_5 \text{ is } A_{5j}, \text{ then } y \text{ is } B_j$$

where A_{ij} $(i = 1, 3, 5)$ are fuzzy subsets for input variable x_i in rule j and B_j is a fuzzy subset for the output y. If no input appears in the premise, the rule is removed.

The purpose of rule structure optimization is to generate local optimal fuzzy rules, therefore, we use a local fitness function for GA. That is to say, the fuzzy rules will be optimized within each data patch. Suppose there are k fuzzy rules for patch j, then the following error function is used for evaluating rule R_{jk}:

$$E_j = \frac{1}{2} \sum_{m=1}^{P} (y_j - y_m)^2 \tag{6}$$

where P is the patch size defined Section II-A, y_m is the mth output data of the system in patch in j, and y_j is the crisp output of rules R_{jk}

$$y_j = \frac{\sum_k w_{jk} y_{jk}}{\sum_k w_{jk}} \tag{7}$$

$$w_{jk} = \prod_{i=1}^{n} g_{ik}(A_{ik}) \tag{8}$$

where g_{ik} is a logic function to be defined in (10). Recall that the maximal number of fuzzy rules generated for patch j is two. Due to similarity checking, one or both of them may be removed. Therefore, $0 \le k \le 2$. Obviously, if both rules in patch j are removed, the data sets in this patch will not play a role in rule structure optimization.

Since a conventional genetic algorithm aims to maximize its fitness function and our goal is to minimize the local modeling error, we use the following fitness function for each subsystem:

$$F_j = \frac{1}{1.0 + E_j} \tag{9}$$

where the constant 1 is added to avoid computational troubles.

The motivation to locally optimize the structure of the rules is clear, i.e., to let the fuzzy rules cover the data sets in the corresponding patch as efficiently as possible. In the application example provided in Section V, we will show that the structure optimization is vital for complexity reduction of the generated fuzzy system.

B. Global Parameter Learning

Fuzzy systems that can learn are becoming more and more popular since the beginning of the last decade. Several learning algorithms developed in the field of artificial neural networks have been applied to fuzzy system learning. Supervised learning based on the gradient method [15], unsupervised learning [16], and reinforcement learning [17] have proved to be very effective to improve the performance of the fuzzy systems. Furthermore, fuzzy systems that can learn from data will be much more objective and the knowledge they acquired is believed to be more profound.

In this subsection, we use the learning algorithms developed on the basis of the gradient method to optimize the parameters of the fuzzy system, including the parameters for the membership functions in the rule premise and the constants in the rule consequent.

Suppose the fuzzy rules generated in the last subsection can be expressed in the following general form:

$$\begin{aligned} &\text{If } g_{1j}\{x_1 \text{ is } A_{1j}\}, \text{ and} \cdots \text{and } g_{nj}\{x_n \text{ is } A_{nj}\}, \\ &\text{Then } y \text{ is } y_j \end{aligned}$$

where $g_{ij}\{\cdot\}$ $(i = 1, 2, \cdots, n)$ is a logic function indicating whether input variable x_i appears in the premise of rule j

$$g_{ij}\{z\} = \begin{cases} z, & \text{if } x_i \text{ appears in the premise} \\ 1, & \text{if } x_i \text{ does not appear in the premise.} \end{cases} \tag{10}$$

Notice that if variable x_i does not appear in the rule, it is equivalent to the condition "x_i is any. "Therefore, the firing strength

is always one for this part of the condition. The cost function used for fuzzy system learning is

$$E = \frac{1}{2}(y(t) - y_d(t))^2 \tag{11}$$

where $y_d(t)$ is the real output of the tth data set and $y(t)$ is the output of the fuzzy model for the tth inputs

$$y(t) = \frac{\sum_{j=1}^{N} w_j y_j}{\sum_{j=1}^{N} w_j} \tag{12}$$

$$w_j = \prod_{i=1}^{n} g_{ij}(A_{ij}). \tag{13}$$

Based on the gradient method, it is straightforward to derive the learning algorithm for the parameters of the fuzzy membership functions. The learning algorithm will be given in Section IV when we discuss the regularization algorithm for interpretability improvement.

In the process of fuzzy system learning, the average firing strength of each fuzzy rule is checked. Those rules with a firing strength that is lower than a prescribed threshold are believed to be inactive rules and will be deleted. In our simulation example, it is found that if each input variable appears in the premise part of all the fuzzy rules, rule reduction is difficult. On the contrary, if the structure of the fuzzy system is optimized, only a small number of the rules are frequently fired. In this case, a large number of inactive rules can be removed without the deterioration of the system performance.

IV. Interpretability Considerations

A. Interpretability of Fuzzy Systems

One important motivation of using fuzzy systems is their explanation capacity. Besides good system performance, one also hopes to gain insights into an unknown system by building a fuzzy model. However, a fuzzy system generated from data is not necessarily understandable. Interpretability or transparency can thus be used to indicate how easily a fuzzy can be understood by human beings. Currently, there exists no well-established definitions for interpretability of a fuzzy system. In [18], [19], some important aspects of fuzzy systems pertaining to interpretability have been discussed. Generally, interpretability of fuzzy systems depends on the following aspects. The first and most important factor is the distribution of the fuzzy membership functions (can be called a fuzzy partition loosely). For fuzzy rules that are easy to understand, a fuzzy partition should be both complete and distinguishable, so that a sound physical meaning (a linguistic term) could be associated with each fuzzy subset in the partition. In most cases, the distinguishability of the fuzzy subsets is the first thing to concern to improve the interpretability of a fuzzy system. In Fig. 2, examples of: 1) indistinguishable and 2) distinguishable fuzzy partitions are illustrated. It should be pointed out that there are no definite criteria for the distinguishability of a fuzzy partition. Generally, it is dependant on the similarity of the fuzzy subsets and the number of fuzzy subsets in the partition. A distinguishable fuzzy partition should not have a lot of very similar fuzzy subsets and the number of

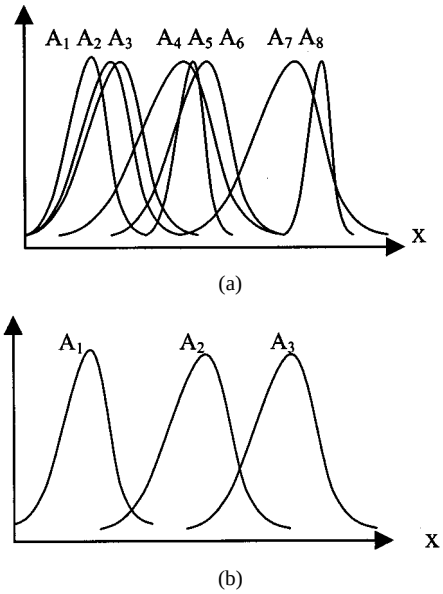


Fig. 2. (a) Indistinguishable and (b) distinguishable fuzzy partitions.

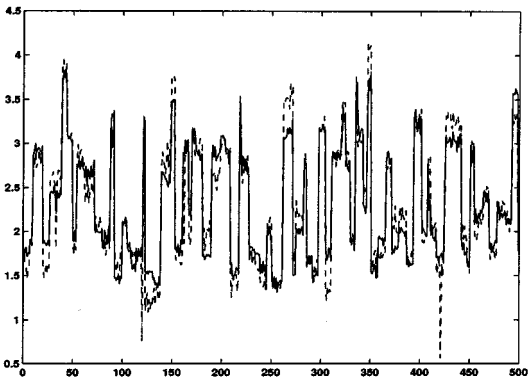


Fig. 3. Approximation results on part of the test data.

fuzzy subsets should not exceed nine [19]. Second, the fuzzy rules in the rule base should be consistent. If there are rules that are strongly contradictory to each other, it is also hard to understand the fuzzy rules. Other factors may include the number of variables in the rule premise and the total number of fuzzy rules in the rule base. The smaller the number of combinations in the rule premise, the easier it is to understand the fuzzy rule. So is the number of fuzzy rules in the rule base. In this section, we focus on the distinguishability of the fuzzy subsets.

B. Finding the Similar Subsets

There are several fuzzy similarity measures, one of which is based on the distance measure

$$S(A, B) = \frac{1}{1 + d(A_1, A_2)}, \qquad S(\cdot) \in (0, 1]. \tag{14}$$

If Gaussian membership functions are involved, the following simple expression can be used to approximate the distance between two fuzzy subsets:

$$d(A_1, A_2) = \sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2} \tag{15}$$

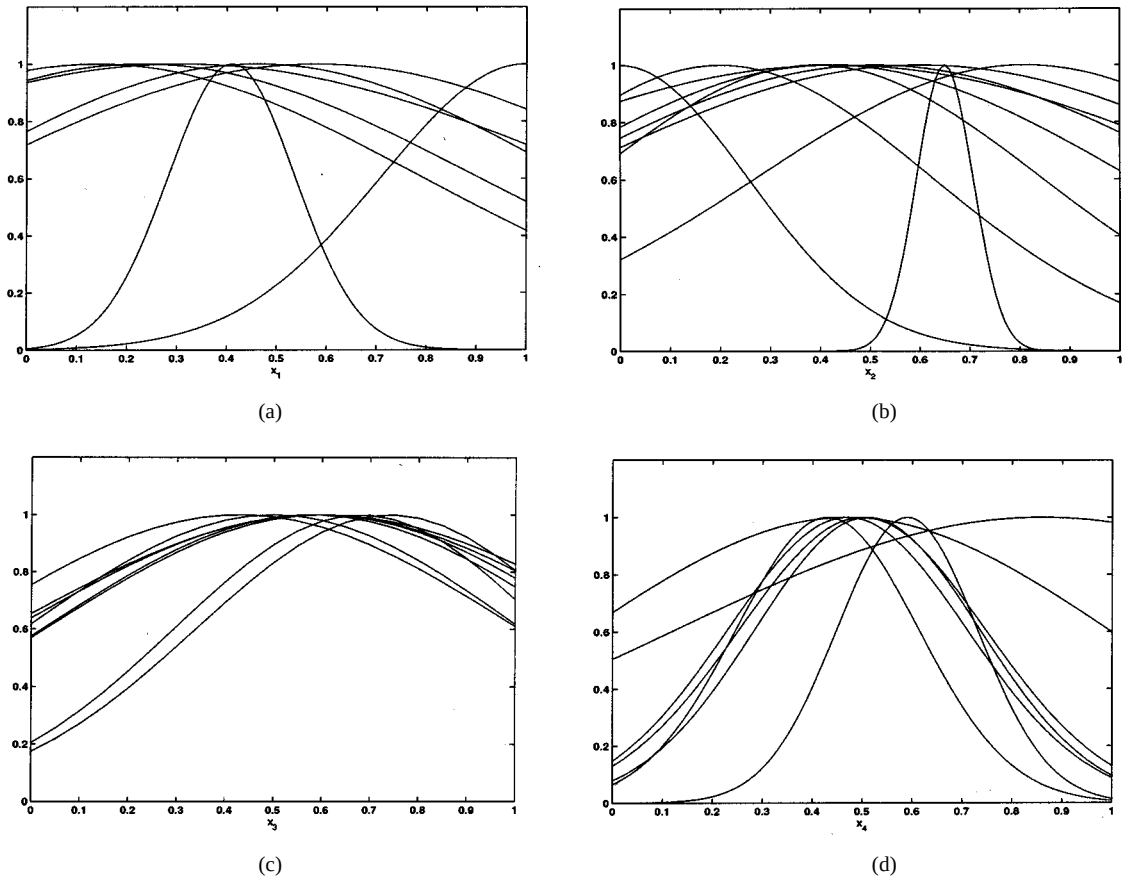


Fig. 4. (a)–(d) Membership functions before interpretability improvement.

assuming the Gaussian membership function has the following form:

$$A_i(x) = \exp\left(-\frac{(x-a_i)^2}{b_i^2}\right). \tag{16}$$

It is shown in our simulation that this simplified measure works well for finding the similar fuzzy sets.

Now we are ready to apply the fuzzy similarity measure to find the similar fuzzy subsets within a fuzzy partition. Assume variable x_i has L_i fuzzy subsets $A_i,\ i = 1,2,\cdots,L_i$. Then they can be divided into m_i groups using a prescribed similarity threshold δ

$$U_{ik} = \{A_i \mid S(A_i, A_{k0}) \geq \delta\}; \quad 1 \leq k \leq L_i \tag{17}$$

where U_{ik} denotes a group of fuzzy subsets that are considered to be similar, A_{k0} is the reference fuzzy set for the group. For example, the eight fuzzy subsets illustrated in Fig. 2(a) can be divided into three groups of similar fuzzy sets: $U_1 = \{A_1, A_2, A_3\}, U_2 = \{A_4, A_5, A_6\}$ and $U_3 = \{A_7, A_8\}$.

The goal of regularization is to drive the similar fuzzy sets in U_{ik} to a same fuzzy set. However, it should be pointed out that during regularization, the fuzzy subsets must be regrouped in each cycle of learning. More discussions on this issue will be provided in the next subsection.

C. Regularized Learning

One of the neural network regularization techniques is called formal regularization, in which an extra penalty term is added to the conventional error function during learning [20]. The penalty term can be determined based on a variety of criteria from the heuristic smoothness method to the information-theoretic approach. In this paper, the aim of regularization is to drive the similar fuzzy sets in set U_{ik} to a same fuzzy set during gradient learning so that the interpretability of the fuzzy system can be greatly improved without seriously deteriorating the system performance. To achieve this, we use the following cost function in regularized learning:

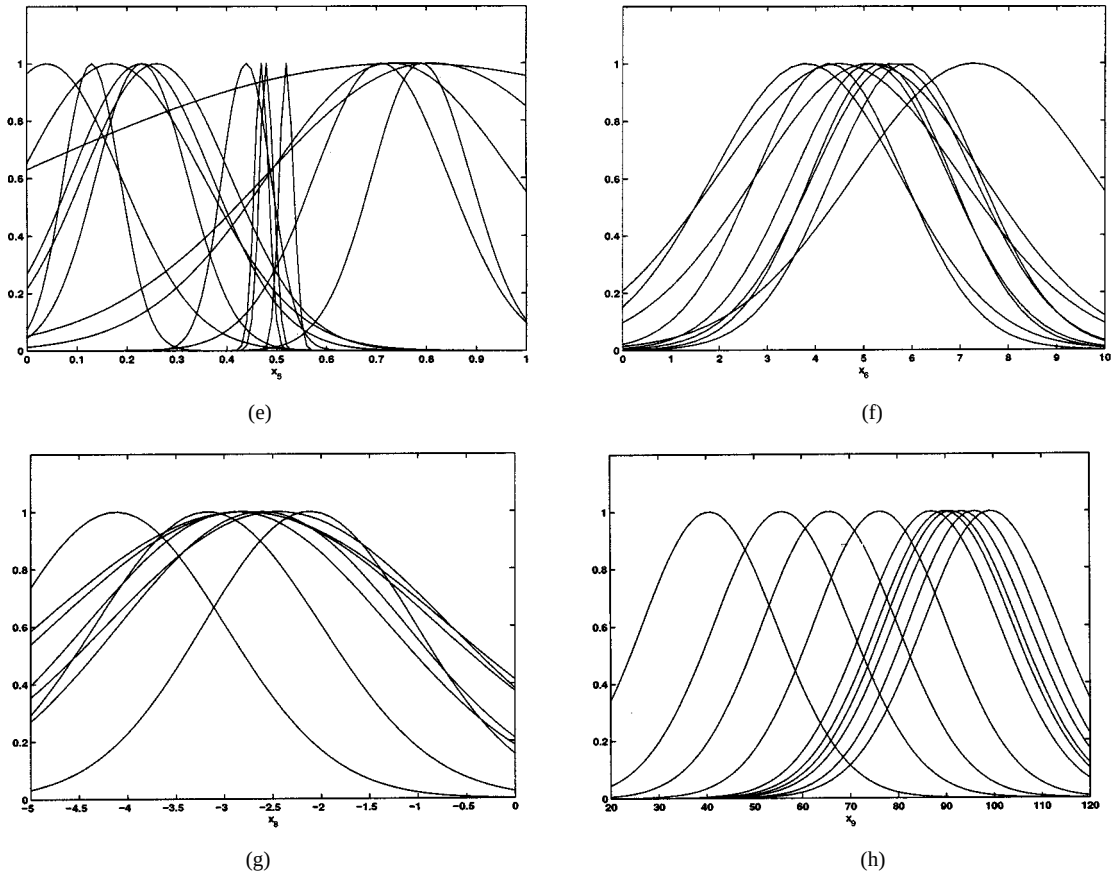
$$J = E + \gamma\Omega \tag{18}$$

where E is the conventional error function defined in (11), γ is called the regularization coefficient $(0 \leq \gamma < 1)$, and Ω is the regularization term for merging the similar fuzzy membership functions

$$\Omega = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m_i}\sum_{A_{ij}\in U_{ik}}(a_{ij} - \bar{a}_{ik})^2 + \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{m_i}\sum_{A_{ij}\in U_{ik}}(b_{ij} - \bar{b}_{ik})^2 \tag{19}$$

where $\bar{a}_{ik}$ and $\bar{b}_{ik}$ are the two parameters in the Gaussian function shared by all the fuzzy subsets in group U_{ik}. In regularized learning, the initial values of $\bar{a}_{ik}$ and $\bar{b}_{ik}$ can be the average of the a_{ij} and b_{ij} of subset A_i in the same group U_{ik}

$$\bar{a}_{ik} = \frac{1}{I_{ik}}\sum_{A_i\in U_{ik}}^{I_{ik}} a_{ij} \tag{20}$$



(e) (f)

(g) (h)

Fig. 4. (*Continued.*) (e)–(h) Membership functions before interpretability improvement.

$$\bar{b}_{ik} = \frac{1}{I_{ik}} \sum_{A_i \in U_{ik}}^{I_{ik}} b_{ij} \tag{21}$$

where I_{ik} is the number of subsets in group U_{ik}. According to the gradient method, the regularized learning algorithm can be developed for the fuzzy system described by (12) and (13)

$$\left.\frac{\partial J}{\partial a_{ij}}\right|_{A_{ij} \in U_{ik}} = \frac{\partial E}{\partial a_{ij}} + \gamma(a_{ij} - \bar{a}_{ik}) \tag{22}$$

$$\left.\frac{\partial J}{\partial b_{ij}}\right|_{A_{ij} \in U_{ik}} = \frac{\partial E}{\partial b_{ij}} + \gamma(b_{ij} - \bar{b}_{ik}) \tag{23}$$

$$\frac{\partial J}{\partial \bar{a}_{ik}} = -\gamma \sum_{A_{ij} \in U_{ik}} (a_{ij} - \bar{a}_{ik}) \tag{24}$$

$$\frac{\partial J}{\partial \bar{b}_{ik}} = -\gamma \sum_{A_{ij} \in U_{ik}} (b_{ij} - \bar{b}_{ik}) \tag{25}$$

$$\frac{\partial E}{\partial a_{ij}} = (y(t) - y_d(t))(w_j - y(t))(x_i - a_{ij}) w_j / b_{ij}^2 \tag{26}$$

$$\frac{\partial E}{\partial b_{ij}} = (y(t) - y_d(t))(w_j - y(t))(x_i - a_{ij})^2 w_j / b_{ij}^3. \tag{27}$$

It is seen that the modification of the parameters depends not only on the system error, but also on how the similar fuzzy subsets converge to the same one. The regularization parameter γ plays a very important role in regularization. If γ is too large, then the similar fuzzy subsets will be merged quickly but the system performance may becomes seriously worse. On the contrary, if γ is too small, then the system performance will be good, but the similar fuzzy subsets may remain indistinguishable and the interpretability of the fuzzy system is not good. It is also necessary to point out that a fuzzy subset A_{ij} that is initially in group U_{ik} may be put in another group and be merged with the subsets in that group. For example, in Fig. 2(a), A_6 is initially in group U_2. During regularized learning, A_6 may move toward group U_3 and finally be merged with the fuzzy sets in U_3. This is due to the fact that to merge A_6 with A_4 and A_5 causes a much larger error than to merge A_6 with A_7 and A_8. This is the reason why regularized learning instead direct merging is used to improve the interpretability of the fuzzy system.

V. Simulation Study

A. Fuzzy Rule Generation and Optimization

The proposed method is applied to an example that contains 20 000 training data sets and 80 000 test sets. The system has 11 input variables and one output variable. For such a high-dimensional system, it is very difficult to adopt a grid partition for

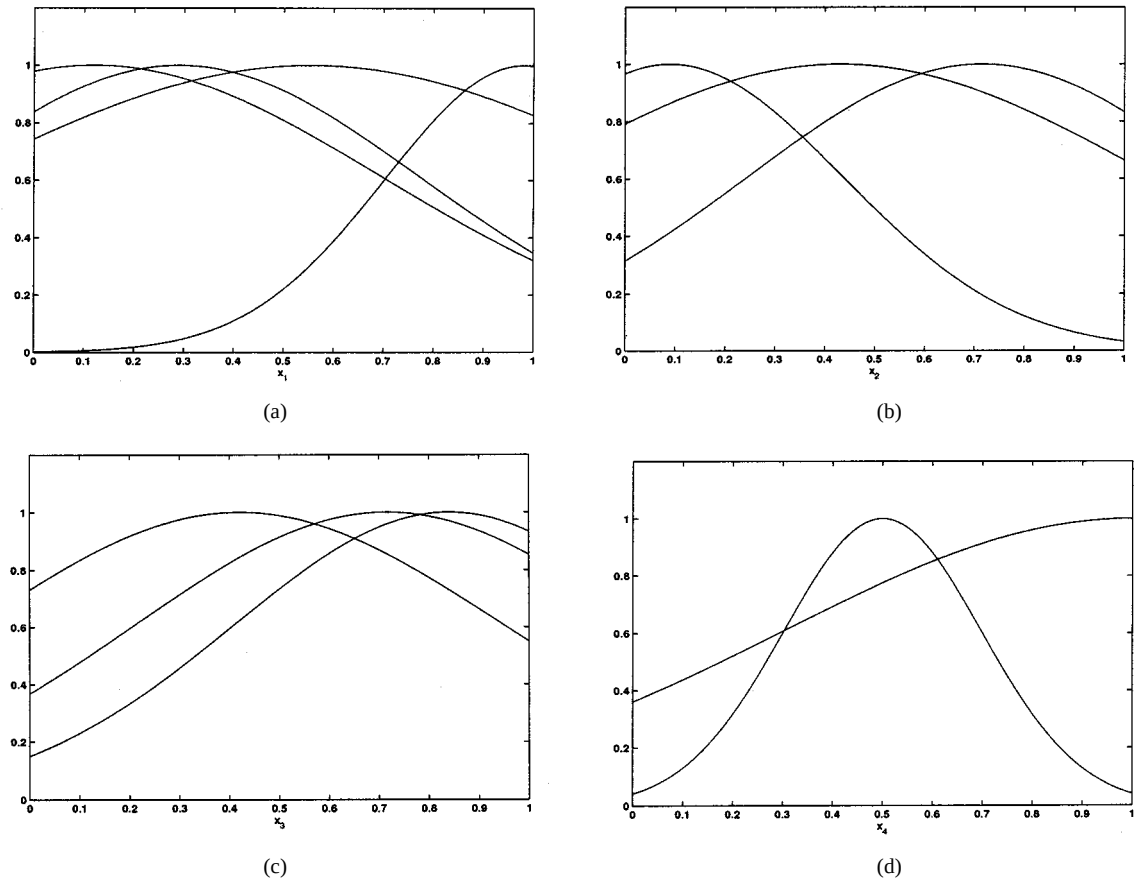


Fig. 5. (a)–(d) Membership functions after interpretability improvement.

fuzzy rule generation. Imagine that we would like to generate a fuzzy system with genetic algorithms. In the simplest case, we would partition each variable into three fuzzy subsets. If the grid fuzzy rule structure is adopted, the total number of fuzzy rules will be 3^{11}. For such a large system, the genetic algorithm will become very inefficient and the optimization results are unacceptable. Since there are 20 000 training data sets, it is difficult to directly employ the method proposed in [13].

According to the method proposed in the previous sections of this paper, we first generate fuzzy rules on the extrema of each patch. To achieve a good balance between the system performance and rule complexity, different patch sizes and similarity thresholds are tried. The resulting fuzzy systems using different patch sizes are evaluated according to rule complexity and the local approximation error. To obtain a reasonable number of fuzzy rules without losing too much information about the system, we finally adopt a patch size of 15 and a similarity threshold of 0.45. As a result, the generated fuzzy system has 1232 fuzzy rules. Recall that without similarity checking, the total number of fuzzy rules will be around 2666 for 20 000 data sets when the patch size is 15. The average local error of the system is 0.167 and the global root mean square (rms) errors on training and test data are 0.259 and 0.305, respectively.

In the fuzzy system generated previously, each variable appears in the premise part of the fuzzy rule. We first attempt to train the fuzzy system without optimization of the rule structure. Unfortunately, we find the fuzzy system has two problems. First, when training proceeds, the error on the test data is not satisfactory. In our simulation, after 300 iterations of gradient learning, the training root mean square (rms) error is 0.127 and the test rms error is 0.233. It is seen that although the training error has been reduced greatly, the test error is almost twice of the training error. Second, we find that the performance of the fuzzy system degrades seriously when we try to reduce the rule complexity. For example, when the rules with an average firing strength lower than 0.055 are deleted, 682 rules remain in the rule base. Even with such a large number of fuzzy rules, the training and test errors increase to 0.252 and 0.301, respectively. Obviously, such a fuzzy rule system is not satisfactory.

Due to the previous problems, we decide to use genetic algorithms to optimize the rule structure. The optimization is carried out in a local sense. Since there are 11 input variables, the length of code for each rule in the individual is 11 with each bit representing whether a variable appears in the rule. It is obvious that if the ith $(i = 1, 2, \cdots, 11)$ bit is zero, x_i will disappear from the premise part of the rule. In the simulation, the population size is 100, crossover probability is 0.75 and mutation rate is 0.01. Elitism selection is adopted. After 100 generations of evolution, the average local error decreases from 0.176 to 0.09. It is conceivable that the decrease of the local error does not necessarily mean the decrease of the global error. In fact, the global training and test errors increase to 0.472 and 0.474. The gradient method is then introduced. The learning result is very inspiring. After 100 iterations of training, the training

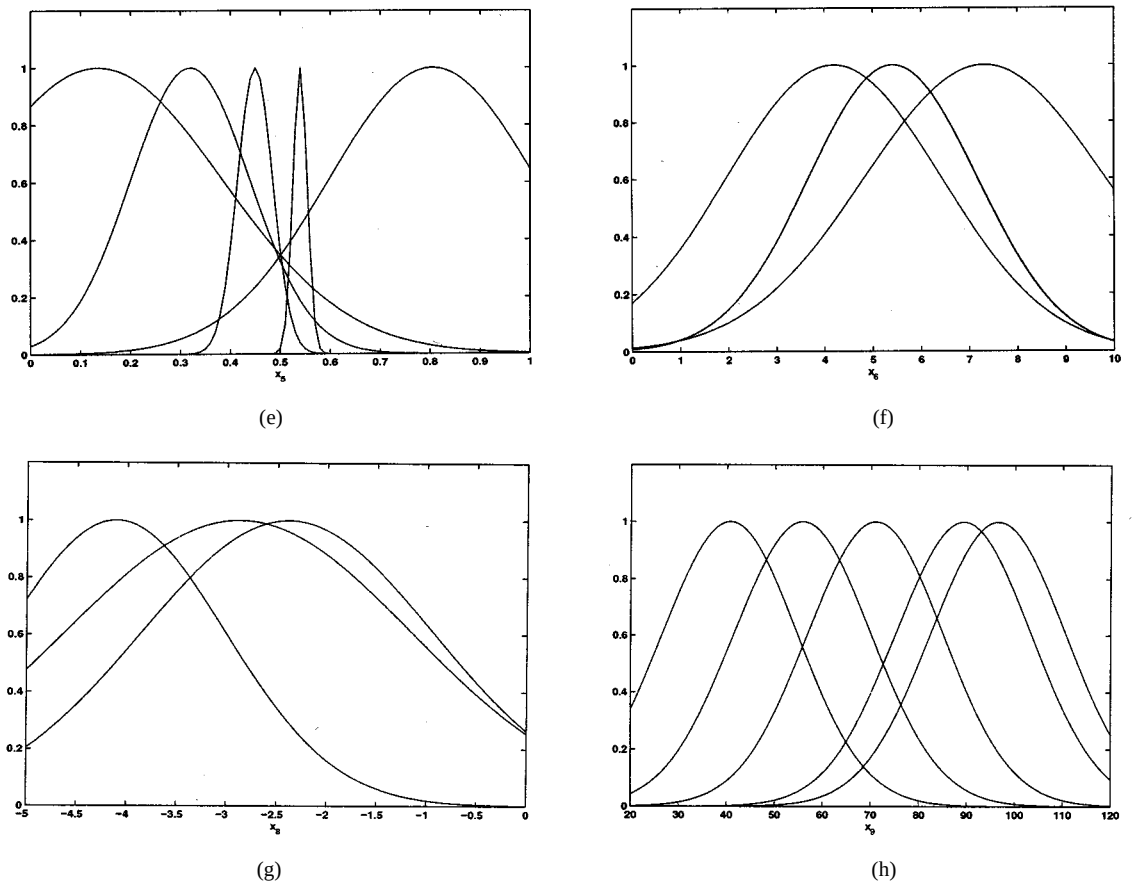


Fig. 5. (*Continued.*) (e)–(h) Membership functions after interpretability improvement.

error and test error are reduced to 0.154 and 0.197, respectively. At this time, the rule number is still 1232. Compared to the results without structure optimization, the test error is fairly reduced, although the training error is increased slightly. Most importantly, the complexity of the fuzzy system can be easily reduced. The same threshold (0.055) is used to delete the inactive rules. After rule deletion, there are only 27 fuzzy rules in the rule base. The training error is 0.189 and the test error is 0.207. This is a very inspiring result, taking into account the large number of input variables and the huge number of training data sets. For clarity, the approximation results of first 500 test data are shown in Fig. 3, where the dotted line denotes the desired values and the solid line is the output of the fuzzy model.

To investigate the dependencies between the inputs and the output, let us have a look at the frequencies that the input variables show up in the rule premise. In the 27 fuzzy rules, we find that x_5 appears 19 times, x_9 appears 11 times, and x_4 appears eight times. In contrast, x_7 and x_{11} shows up only two times. This implies that in the system, x_5, x_9 and x_4 play a much more important role than x_7 and x_{11}. This conclusion coincides with the results obtained by using statistical correlation analyses.

We also found that the average number of input variables that appear in the rule premise is fairly small. In the rule base, the maximal number of variables appearing in the premise part is five and the minimal number is one with an average of 3.2 over the 27 fuzzy rules. It is believed that a simpler rule premise contributes to a better rule interpretability.

B. Interpretability Improvement

The fuzzy model is very good considering the system performance and rule size. However, when we have a look at the membership functions of the input variables, it is found that most of the variables have so many fuzzy subsets that it is almost impossible to assign proper linguistic terms to the fuzzy sets [Fig. 4(a)–(h)]. Due to space limit, the membership functions for x_7, x_{10} and x_{11} are not plotted in the figure, because their fuzzy membership functions in the fuzzy partition are fairly distinguishable. To improve interpretability of the fuzzy system, we first try to directly merge the similar fuzzy sets in the fuzzy partitions of x_1 to x_6, x_8 and x_9. The result is not good: the rms error for training data increases to 0.291 and the RMS error for test data to 0.324.

We then implement the regularization algorithm introduced in Section IV. The results are very inspiring. Although some of the fuzzy subsets do not converge to exactly the same value, the parameter difference between them are minor and direct merging of these subsets will not affect the system performance much. The final membership functions are plotted in Fig. 5(a)–(h) except for x_7, x_{10} and x_{11}. Compared to the fuzzy membership functions before regularization, they become fairly distinguishable and the number of fuzzy sets in the fuzzy partitions is significantly reduced. In this way, the interpretability of the fuzzy system is greatly improved. The RMS errors for training and test are 0.191 and 0.213, respectively. Compared to the errors

TABLE I
PERFORMANCE OF THE FUZZY SYSTEMS BASED ON DIFFERENT METHODS

	Rule number	Training error	Test error	Interpretability
Structure not optimized, inactive rules not deleted	1233	0.127	0.233	Bad
Structure not optimized, inactive rules deleted	682	0.252	0.301	Bad
Structure optimized, inactive rules not deleted	1233	0.154	0.197	Bad
Structure optimized, inactive rules deleted	27	0.189	0.207	Bad
Structure optimized, inactive rules deleted, direct merging	27	0.291	0.324	Good
Structure optimized, inactive rules deleted, regularized learning	27	0.191	0.213	Good

before regularization (0.189 and 0.207), we find the error increase can be neglected. Table I lists the system performance and the number of rules for the different systems discussed in the simulation.

C. Discussions

Through the simulation example, it is shown that the proposed method is able to build a compact and interpretable fuzzy rule system for high-dimensional systems with large number of experimental data. The effectiveness of the method can mainly be ascribed to the following reasons: 1) the fuzzy rules are generated on extrema; 2) the structure of the fuzzy rule system is optimized in a local sense; and 3) regularization rather than hard merging is used in improving the interpretability of the fuzzy system. Since no *ad hoc* assumptions on the system are made in this method, it is applicable to any systems. However, it should also be pointed out to what degree the rule system can be simplified and how many input variables appear in the rule premise, i.e., the optimal rule structure, are dependant to some extent on the training data and the nature of the original system. This agrees with the following two facts: 1) all knowledge of a statistical model comes solely from the data and 2) the optimal rule structure is determined by the inherent structure of the original system.

VI. CONCLUSION

An approach to fuzzy modeling of high-dimensional systems has been proposed in this paper. The proposed method can be divided into the following steps: 1) generation of fuzzy rules that cover the extrema directly from data; 2) rule similarity checking for deletion of the redundant and inconsistent rules; 3) optimization of the rule structure using genetic algorithms based on a local performance index; 4) further training of the rule parameters using gradient based learning method and deletion of the inactive rules; 5) interpretability improvement using regularization. In this way, a compact and interpretable fuzzy model can be obtained for a high-dimensional system. Through structure optimization, the relationship between the inputs and the output can also be revealed, which is very important for understanding an unknown system. The effectiveness of the method is shown by an example. With 20 000 training data and 11 input variables, the final fuzzy system has only 27 fuzzy rules with a very good performance on both training and test data sets.

REFERENCES

[1] M. Sugeno and K. Tanaka, "Successive identification of a fuzzy model and its application to prediction of a complex model," *Fuzzy Sets Syst.*, vol. 42, pp. 315–334, 1991.
[2] C. Sun, "Rule-base structure identification in an adaptive-network-based inference systems," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 64–73, Feb. 1994.
[3] T. Kavli, "ASMOD: An algorithm for adaptive spline modeling of observation data," *Int. J. Contr.*, vol. 58, no. 4, pp. 947–968, 1993.
[4] G. S. V. Raju and J. Zhou, "Adaptive hierarchical fuzzy controllers," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 973–980, Aug. 1993.
[5] Y. Jin, "Decentralized adaptive fuzzy control of robotic manipulators," *IEEE Trans. Syst., Man, Cybern.—Part B*, vol. 28, pp. 47–57, Jan. 1998.
[6] Y. Jin, W. von Seelen, and B. Sendhoff, "On generating flexible, complete, consistent and compact fuzzy rule systems from data using evolution strategies," *IEEE Trans. Syst., Man, Cybern.—Part B*, vol. 29, pp. 829–845, Dec. 1999.
[7] L. T. Koczy and K. Hirota, "Size reduction by interpolation in fuzzy rule bases," *IEEE Trans. Syst., Man, Cybern.—Part B*, vol. 27, pp. 14–25, Feb. 1997.
[8] B. Kosko, "Optimal fuzzy rules cover extrema," *Int. J. Intell. Syst.*, vol. 10, no. 2, pp. 249–255, 1995.
[9] W. E. Combs and J. E. Andrews, "Combinatorial rule explosion eliminated by a fuzzy rule configuration," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 1–11, Feb. 1998.
[10] A. Lotfi, H. C. Anderson, and A. C. Tsoi, "Interpretation preservation of adaptive fuzzy inference systems," *Int. J. Approxim. Reasoning*, vol. 15, no. 4, 1996.
[11] M. Setnes, R. Babuska, U. Kaymak, and H. R. van Nauta Lemke, "Similarity measures in fuzzy rule base simplification," *IEEE Trans. Syst., Man, Cybern.—Part B*, vol. 28, pp. 376–386, June 1998.
[12] J. Yen, L. Wang, and W. Gillespie, "A global-local learning algorithm for identifying Takagi–Sugeno–Kang fuzzy models," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 967–972.
[13] L. X. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 1414–1427, Dec. 1992.
[14] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. New York: Addison-Wesley, 1989.
[15] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control systems," *IEEE Trans. Comput.*, vol. 40, pp. 1320–1336, Dec. 1991.
[16] J. Nie and D. A. Linkens, "Fast self-learning multivariable fuzzy controllers constructed from a modified CPN network," *Int. J. Contr.*, vol. 60, no. 3, pp. 369–393, 1994.

[17] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy controllers through reinforcement," *IEEE Trans. Neural Networks*, vol. 3, pp. 724–739, Sept. 1992.

[18] Y. C. Jin, W. von Seelen, and B. Sendhoff, "An approach to rule-based knowledge extraction," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 1188–1193.

[19] J. V. de Oliveira, "On the optimization of fuzzy systems using bio-inspired strategies," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Anchorage, AK, May 1998, pp. 1129–1134.

[20] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.