퍼지 및 지능시스템학회 논문지 2007, Vol. 17. No. 2, pp. 276-283

Structure Identification of a Neuro-Fuzzy Model Can Reduce Inconsistency of Its Rulebase

Bo-Hyeun Wang(*) and Hyun-Joon Cho(**)

(*)Department of Electrical Engineering, Kangnung National University
(**)Department of Electrical and Computer Engineering, Purdue University

Abstract

It has been shown that the structure identification of a neuro-fuzzy model improves their accuracy performances in a various modeling problems. In this paper, we claim that the structure identification of a neuro-fuzzy model can also reduce the degree of inconsistency of its fuzzy rulebase. Thus, the resulting neuro-fuzzy model serves as more like a structured knowledge representation scheme. For this, we briefly review a structure identification method of a neuro-fuzzy model and propose a systematic method to measure inconsistency of a fuzzy rulebase. The proposed method is applied to problems of fuzzy system reproduction and nonlinear system modeling in order to validate our claim.

Key Words : Neuro-fuzzy modeling, structure identification, measure of inconsistency, input space partitioning, genetic algorithm.

1. Introduction

Neuro-fuzzy modeling is a problem to identify a fuzzy model of a system on the basis of input-output data by using the neuro-fuzzy systems [1]. Identification of neuro-fuzzy models consists of two sub problems: structure identification and parameter identification. The structure identification partitions the associated input space while the parameter identification optimizes the adjustable parameters of the neuro-fuzzy models.

For most practical modeling problems, there are two prime concerns: model's efficiency and model's accuracy. A common neuro-fuzzy modeling practice uses numerical data for parameter identification, while employing an arbitrary number of fuzzy rules and random initial parameters. Without an efficiency requirement, this is a feasible approach at least in practical viewpoint. However, when we must deal with both the model's efficiency and the model's accuracy simultaneously, we inevitably raise the following question: what is the minimum number of fuzzy rules to achieve a certain level of accuracy? This challenging problem is dealt by the structure identification of neuro-fuzzy models. A number of structure identification methods have been proposed and applied to various modeling problem with improved performances [2,3,4].

Although we know the minimum number of fuzzy rules *a priori*, some neuro-fuzzy models which consist of that minimum number of rules do not show an acceptable accuracy, since the initial parameters significantly affect the performance of parameter identification [5]. Furthermore, among the models of the acceptable accuracy which are built through repeated trials, most of them have some inconsistent rules, i.e., rules that have the same premise part but different consequent parts. We can view the identified neuro-fuzzy models of some inconsistent rules as something more like black-box models such as neural networks.

In this paper, we address that the structure identification method can build a more consistent neuro-fuzzy model which satisfies the predetermined accuracy requirement, while using the least possible number of fuzzy rules. The structure identification method under consideration is based on clustering, cell map approximation, and shape refinement using GA. In order to highlight the performance of the structure identification, its modeling performance is compared with that obtained without structure identification.

This paper is organized as follows: Section 2 reviews the neuro-fuzzy systems. In Section 3, we address the structure identification of a neuro-fuzzy model using GA and present a method to measure the degree of inconsistency of a fuzzy rulebase. Section 4 presents the simulation results using a couple of illustrative examples. Section V provides concluding remarks.

2. Neuro-Fuzzy Systems

Fuzzy inference systems provide a computing framework based on the concepts of fuzzy sets, fuzzy rules, and fuzzy reasoning. The basic structure of the fuzzy inference systems

접수일자 : 2006년 11월 30일
완료일자 : 2007년 4월 4일
감사의글 : 이 논문은 2006년도 강릉대학교 학술 연구조성비 지원에 의하여 수행되었으며 이에 감사드립니다.

consists of a fuzzy rulebase, a reasoning mechanism, and a defuzzification. A fuzzy rulebase is a set of fuzzy if-then rules that are expressed in the following form:

$$\begin{aligned} &\text{Rule 1: If } (x_1 \text{ is } A_1^1) \text{ and ... and } (x_n \text{ is } A_n^1), \text{ then } y \text{ is } q^1, \\ &\quad\vdots \\ &\text{Rule } p\text{: If } (x_1 \text{ is } A_1^p) \text{ and ... and } (x_n \text{ is } A_n^p), \text{ then } y \text{ is } q^p, \end{aligned} \tag{1}$$

where x_j $(1 \le j \le n)$ are the input variables, y is the output variable

Suppose that we have two fuzzy rules where each fuzzy rule has two inputs and a single output. The architecture of the corresponding neuro-fuzzy system is shown in Fig. 1. The input term node denoted as A_j^i accepts x_j as the input and produces the output which is the degree of matching between x_j and its corresponding membership function. If we use the Gaussian membership functions for A_j^i, the output of the input term node is given by

$$\mu_{A_j^i}(x_j) = \exp\left[-\left(\frac{x_j - c_j^i}{\sigma_j^i}\right)^2\right], \tag{2}$$

where c_j^i and σ_j^i are called the premise parameters. The ith node in the second layer produces the output that represents the firing strength of the ith rule:

$$R_i(x) = \prod_{j=1}^{n} \mu_{A_j^i}(x_j) \tag{3}$$

The output term node in the third layer computes the normalized firing strength and provides the computed value to the last layer that acts as the defuzzifier. If we use the center of gravity defuzzification from local centroids, the output of this layer can be written as

$$y = \sum_{i=1}^{p} q^i R_i \Big/ \sum_{j=1}^{n} R_j , \tag{4}$$

where p denotes the number of fuzzy rules and n represents the number of input variables.

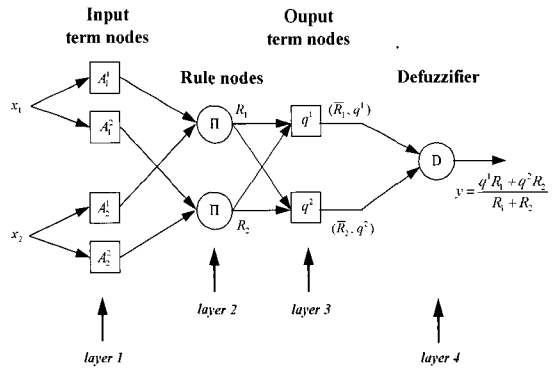


Fig. 1. Neuro-fuzzy system

3. Neuro-Fuzzy Modeling

3.1 Structure identification using genetic algorithm

Structure identification of the neuro-fuzzy model, which determines a global function structure for a nonlinear process to be modeled, deals with two problems: input variables selection and input space partitioning. The problem of input variables selection is to select a set of input variables that affect the output of a system. In most cases, a finite number of possible candidates are assumed to be given. The problem of input space partitioning, on the other hand, is to find a set of fuzzy partitions that are either overlapped or disjoint or both. Although both problems are equally important, we mainly deal with the problem of input space partitioning in this paper.

The structure identification in general and the input space partitioning in particular are imperative for improving learning/operation efficiency of the neuro-fuzzy models. Without solving the problem of input space partitioning, what we can do is to make an educational guess on the number of fuzzy partitions and to initialize the associated parameters either intuitively or randomly. However, in this way, some neuro-fuzzy models do not exhibit an acceptable accuracy. Furthermore, among the models which have acceptable accuracies, most of them have many inconsistent rules. The inconsistent rules sacrifice one of the advantages of the neuro-fuzzy model, since they make it difficult to further use the extracted rules after the modeling process.

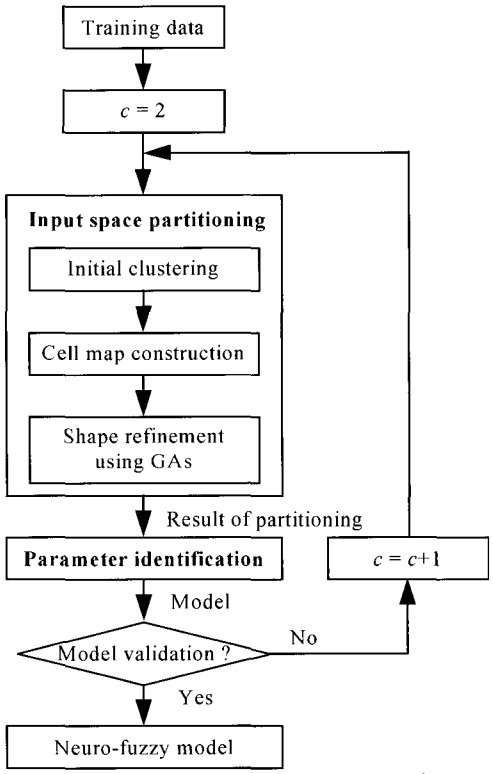


Fig. 2. Procedure of the structure identification of neuro-fuzzy modeling

In this section we shall review a method to partition the input space using genetic algorithm which was proposed in [4]. Fig. 2 illustrates the flow chart of the proposed input space partitioning method that consists of three steps: initial clustering, cell map construction, and shape refinement. As shown in Fig. 2, the method starts with the minimum number of clusters, i.e., c=2 and then increments it until the identified model satisfies the prespecified conditions.

3.1.1 Initial Clustering and Projection

The initial clustering clusters the output data and projects the resultant clusters into the input space. For the initial clustering, we adopt the method presented in [1] which uses the fuzzy c-means algorithm [6]. As a result of the clustering, every output data y^i is associated with the grade of membership belonging to the fuzzy clusters $\tilde{O}_j$'s where i=1,2,...,N and j=1,2,...,c.

$$\left[y^i;\mu_{\tilde{O}_1}(y^i),\mu_{\tilde{O}_2}(y^i),\ldots,\mu_{\tilde{O}_c}(y^i)\right] \tag{5}$$

where $\mu_{\tilde{O}_j}(y^i)$ is the grade of the ith data belonging to the jth cluster, N is the number of data to be clustered, and c is the number of clusters.

Once we complete the clustering, we project the fuzzy clusters in the output space onto the input space. For this, we first transform the fuzzy clusters into the crisp clusters by taking the fuzzy cluster whose grade of membership of y^i is the maximum. A projected input cluster P_j is found by identifying a group of the input data which are associated with all the output data belonging to a crisp cluster O_j. This leads us to c groups of the input data.

3.1.2 Cell Map Construction

In order to perform the shape refinement, we construct a cell map from the groups of data in the input space. We decompose the space of interests $\boldsymbol{X}=X_1\times\cdots\times X_n$ into a finite collection of rectangular cells.

To label a cell, we inspect all training data that belong to the cell. If a cell contains at least one data which belongs to the input cluster P_j, then it is called P_j-labeled cell. When we label the cells, we may have three types of cells: empty cells, homogeneous cells, and nonhomogeneous cells. The empty cell does not contain any data at all and thus has no label. The homogeneous cell has only one label, whereas the nonhomogeneous cell has more than one label.

3.1.3 Shape Refinement using GAs

We view the shape refinement as an optimization problem. In order to build a cost function for the shape refinement, we first set up a partitioning strategy as follows:

Partitioning Strategy: Minimize the overlaps between the hyper-rectangular partitions.

Too much overlap between two hyper-rectangles implies that they are either inconsistent or redundant with each other. The partitioning strategy leads us to the following simple cost function:

$$g(H_1,H_2,\ldots,H_p)=\sum_{i=1}^{p}\sum_{j=i+1}^{p}V(H_i\cap H_j) \tag{6}$$

where p is the number of hyper-rectangles, H_i is the ith hyper-rectangle, and $V(H_i\cap H_j)$ is the volume of $H_i\cap H_j$.

In order to refine the shapes of the input clusters, the resulting hyper-rectangles have to satisfy the following constraints:

Constraint 1 (VPC): The P_j-labeled hyper-rectangles must include all P_j-labeled cells.

Constraint 2 (NIC): A hyper-rectangle must include at least one nonempty cell.

The problem of shape refinement is to find a set of p hyper-rectangles which satisfy both VPC and NIC so that they minimize the cost function given in (6). The penalty method associates a cost with the constraint violations:

$$f(H_1,\ldots,H_p)=g(H_1,\ldots,H_p)+\gamma\Psi(H_1,\ldots,H_p) \tag{7}$$

where γ is a penalty coefficient and

$$\Psi(H_1,\ldots,H_p)=\begin{cases}0, & \text{if } \{H_1,\ldots,H_p\} \text{satisfies VPC,}\\ 1, & \text{otherwise}\end{cases} \tag{8}$$

In order to apply GAs to this optimization problem, the hyper-rectangles that are the variables of the problem have to be represented as a string. Suppose that we have p hyper-rectangles in the n dimensional input space. Since a hyper-rectangle can be uniquely defined by two pointers on each of the input variables, p such vectors are concatenated to form a string of $2np$ length.

Now, we apply GA search in order to solve the problem. The detailed procedure of shape refinement that includes the GA cycle is shown in Fig. 3. The GA cycle involves initialization of a population, evaluation of the strings in a population, and genetic operations on a population.

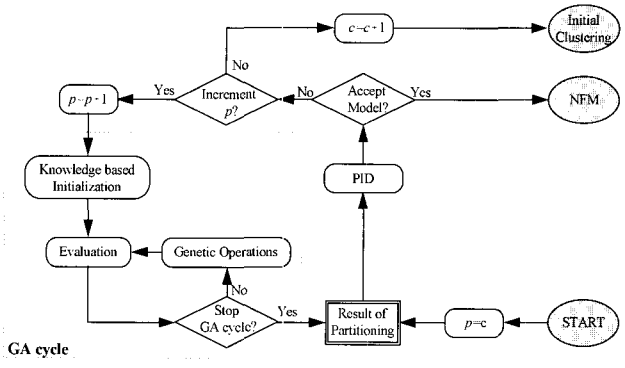


Fig. 3. The GA cycle

As shown in Fig. 2, the whole procedure of the input space partitioning starts with the smallest c. With a given c, we set p to c. When we obtain the results of partitioning for p=c, we perform the parameter identification, while the parameters of the neuro-fuzzy model are initialized by using the results of partitioning. After the parameter identification, we validate the identified neuro-fuzzy model. If we can accept the model, then we stop the procedure. Otherwise, we increment p to p+1 and go to the GA cycle again.

We terminate the execution of the GA cycle after the number of executions of the GA cycle reaches a prespecified value. After termination, we select the string that has the minimum cost function value. Using the string chosen, we initialize the premise parameters of the neuro-fuzzy model. The centers of the Gaussian membership functions for the ith hyper-rectangle (fuzzy rule) are simply given by

$$c_j^i = x_{j\min}^i + \frac{x_{j\max}^i - x_{j\min}^i}{2}, \text{ for all } i \text{ and } j, \quad (9)$$

where $x_{j\min}^i$ and $x_{j\max}^i$ are the values of the left edge and the right edge in the x_j-direction, respectively. The width of the membership function is given by

$$\sigma_j^i = \frac{x_{j\max}^i - c_j^i}{[\ln(1/\alpha)]^{1/2}}, \text{ for } i = 1,\ldots,p \text{ and } j = 1,\ldots,n \quad (10)$$

where α is the value of the α-cut fuzzy set which is used for constructing the hyper-rectangle. The consequent parameters associated with the hyper-rectangles are also initialized by using the centers of the output fuzzy clusters:

$$q^i = v_j, \text{ if } H_i \text{ is the } P_j \text{ labeled hyper - rectangle} \quad (11)$$

We perform the parameter identification with the initial parameters determined as above. If the identified model is accurate enough, we stop the modeling process. Otherwise, we increment p and repeat the GA cycle

3.2 Measure of inconsistency of a fuzzy rulebase

Information (rules) encoded in a neuro-fuzzy system is interpreted as a set of partitions in the input and output space. Suppose that a neuro-fuzzy system acquires a fuzzy rule:

$$\text{If } (x_1 \text{ is } A_1) \text{ and } (x_2 \text{ is } A_2), \text{ then } y \text{ is } q, \quad (12)$$

where two fuzzy sets A_1 and A_2 of X_1 and X_2 are shown in Fig. 4. The associated α-cut sets $A_{1\alpha}$ and $A_{2\alpha}$ are the crisp sets of elements which belong to the fuzzy sets A_1 and A_2, respectively:

$$A_{1\alpha} = \{x_1 \in X_1 \mid \mu_{A_1}(x_1) \geq \alpha\}, \quad (13.a)$$

$$A_{2\alpha} = \{x_2 \in X_2 \mid \mu_{A_2}(x_2) \geq \alpha\}, \quad (13.b)$$

where $0 \leq \alpha < 1$. To obtain the hyper-rectangular partition which represents the premise part of the fuzzy rule, we first construct the cylindrical extension $c(A_{1\alpha})$ and $c(A_{2\alpha})$ with the bases $A_{1\alpha}$ and $A_{2\alpha}$, respectively, and then find the intersection of the supports of the cylindrical extensions:

$$H(\alpha) = S(c(A_{1\alpha})) \cap S(c(A_{2\alpha})). \quad (14)$$

It should be noted that the shape of the partition defined in $X_1 \times X_2$ is hyper-rectangular.

Fuzzy rules are allowed to be inconsistent to some degree in nature. Then to what degree are a set of fuzzy rules inconsistent with each other? To answer this question we need to define a measure of inconsistency. Such measure was first given by Pedrycz [7] in which a degree of difference between two fuzzy sets was expressed as the possibility measure of the fuzzy sets. Suppose that a neuro-fuzzy system implements a fuzzy rulebase given in (1). The possibility measure of two fuzzy sets which reflects the extent to which they coincide or overlap is given by

$$Poss\left(A_k^i \middle| A_k^j\right) = \sup_{x_k \in X_k}\left[\min\left(\mu_{A_k^i}(x_k), \mu_{A_k^j}(x_k)\right)\right], \quad (15)$$

where X_k is the universe of discourse of A_k^i and A_k^j. The degree of inconsistency between two rules is defined as

$$DIC_r(i,j) = \begin{cases} \min\limits_k\left[Poss\left(A_k^i \middle| A_k^j\right)\right], & \text{if } \left|q^i - q^j\right| \geq \Delta_y \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

where $\Delta_y = (q_{\max} - q_{\min})/p$ and k = 1,2,...,n. In addition, the degree of inconsistency for a fuzzy rulebase is given by

$$DIC_{rb}(\beta) = \sum_{i=1}^{p}\sum_{j=i+1}^{p} DIC_n(i,j:\beta), \quad (17.a)$$

where DIC_n is given by

$$DIC_n(i,j:\beta) = \begin{cases} 1, \text{ if } DIC_r(i,j) \geq \beta \\ 0, \text{ otherwise.} \end{cases} \quad (17.b)$$

$DIC_{rb}(\beta)$ in (17.a) simply implies the number of pairs of fuzzy rules whose degree of inconsistency is larger than β. Here, we normally choose β which should be larger than 0.5, since some inconsistencies between fuzzy rules are natural phenomena.

It is obvious that, as $DIC_{rb}(\beta)$ becomes larger, the neuro-fuzzy model is less of a knowledge representation scheme, rather it is more of a black-box model such as neural networks. It should be also noted that there might be many other ways to define a measure of inconsistency for a fuzzy rulebase. One example is that 1 in (17.b) can be replaced with $DIC_r(i,j)$. The similarity measure of fuzzy rules can provide another alternative to define a measure of inconsistency [8].

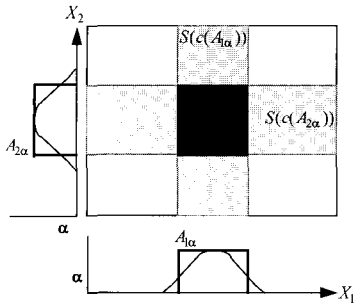


Fig. 4. A fuzzy rule as a hyper-rectangle

4. Simulation Results

This section presents simulation results of the proposed method in order to validate our claim that the structure identification method presented can reduce the degree of inconsistency of the fuzzy rulebase of a neuro-fuzzy model. For this, we use two examples: fuzzy system reproduction [1] and modeling of the process of a gas furnace [9].

4.1 Fuzzy system reproduction

In this example, we consider a reproduction problem which builds a neuro-fuzzy model of a fuzzy system [1]. The fuzzy system under consideration has 9 fuzzy rules as shown in Fig. 5. Each input variable has three linguistic values which are represented by the Gaussian membership functions and the output variable takes singletons.

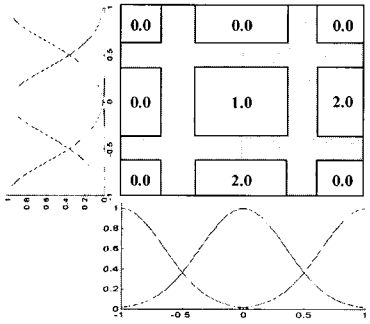


Fig. 5. The fuzzy rulebase of the fuzzy system to be identified

For comparison purpose, we first build neuro-fuzzy models without structure identification. Thus, we have to determine a type of fuzzy partitions, the number of fuzzy partitions (fuzzy rules), and a set of initial parameters through trial and error.

To collect a set of training data, we apply the random inputs which are uniformly distributed in the range of [-1,1]×[-1,1]. We adjust the parameters in a neuro-fuzzy model using the back propagation method presented in [10,11]. The initial parameters (c_j^i's, σ_j^i's and q^i's) are randomly chosen and the back propagation algorithm employs a step size of 0.02 and is terminated after 1000 epochs.

Using the same training data and different sets of initial parameters, we build 100 neuro-fuzzy models in which 50 models have 5 fuzzy rules and the remaining 50 are of 7 rules. The accuracy of an identified neuro-fuzzy model is measured by the mean absolute error (MAE) of the output of the model:

$$e_1 = \frac{1}{N}\sum_{i=1}^{N}\left|y^i - \hat{y}^i\right|, \tag{18}$$

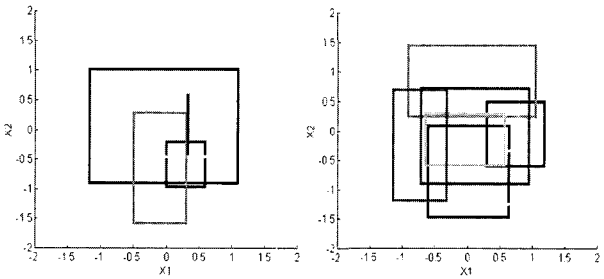
where N is the number of data, y^i is the ith actual output, and $\hat{y}^i$ is the ith model output. Table 1 lists the number of the identified models whose accuracies are in a particular range. As expected, it is likely that we can build a more accurate fuzzy model when the number of rules becomes larger. The accuracy of a model is very sensitive to the initial parameters, even though the sensitivity becomes less as the number of rules increases. As the number of fuzzy rules increases, it becomes more difficult to have a model whose rules are consistent with each other. It can be also seen from this table that DIC_{rb} is not directly related to the accuracy of the model; some accurate models may have a larger DIC_{rb} and *vice versa*.

Table 1. Performance of the neuro-fuzzy models identified from random initial parameters

Range of Accuracy	(0.0, 0.01]	(0.01, 0.02]	(0.02, 0.03]	(0.03, 0.04]	(0.04, 0.05]	(0.05, 0.06]	(0.06, 0.20]
5 rules	0 (0)	2 (2.5)	10 (2.9)	3 (4.0)	5 (2.0)	15 (2.7)	15 (2.2)
7 rules	0 (0)	12 (4.3)	14 (5.4)	11 (4.6)	4 (4.5)	3 (2.7)	6 (2.3)

The number in parenthesis indicates the average DIC_{rb}(β=0.6) of the models whose accuracy is in the range.

In order to examine how the parameter identification procedure has partitioned the input space, we display the layouts of resultant partitions of some models in Fig. 6. In this figure, all hyper-rectangles are constructed by setting α in (14) to 0.2. Fig. 6(a) and Fig. 6(b) are the partitioning results of the 5-rule model of the best MAE and the 7-rule model of the best MAE, respectively.



(a) (b)

Fig. 6. Partitions of neuro-fuzzy models whose initial parameters are randomly chosen. (a) A 5-rule fuzzy model (MAE=0.0165, DIC_{rb}(β=0.6)=3). (b) A 7-rule fuzzy model (MAE=0.0127, DIC_{rb}(β=0.6)=5).

We can make several interesting observations from Fig. 6. First, the layout of the partitions depends heavily on the initial parameters. The fuzzy rules identified from one set of initial parameters are quite different from those of the other sets of initial parameters. Second, some rules cover the whole region on which all input training data occur. The property of the local representation of fuzzy rules disappears in this case. Finally, there may be some inconsistent rules. For instance, a couple of hyper-rectangles contained in the largest hyper-rectangle in Fig. 6(a) are potentially inconsistent.

The observation that we often end up with some inconsistent rules when random initial weights are used for parameter identification of the neuro-fuzzy models is not new. In fact, there has been a debate on the inconsistent rules. Bien and Yu [12] showed that a fuzzy model with some inconsistent rules could still be acceptable as far as the accuracy was the only concern. We may say that we can confirm Bien and Yu's claim through the above simulation. Wang and Mendel [13] proposed a method to select a rule among a group of inconsistent rules when they built a neuro-fuzzy model from both linguistic and numerical information. Jang in [14] suggested that the constrained gradient descent could be employed to maintain the ε-completeness for the grid partitions.

Next, we build neuro-fuzzy models using the proposed structure identification based on GA. We have a set of training data which consists of 200 samples collected above. In order to construct the cell map, we have to cluster the output data by using FCM algorithm. In this example, we use c=2 and m=2 for the FCM parameters. After clustering, we project the output fuzzy clusters into the input space. Considering the space of interests $\boldsymbol{X}$=[-0.9,0.9]×[-0.9,0.9], we construct the cell map of 10×10 cells.

Next stage is for the GA cycle to provide a set of scatter partitions from a given cell map. Since the cell map has been constructed by setting c to 2, we start with p=2. For each GA cycle, we set the number of strings in the population to 200. The length of each string is $4p$ since the input dimension is 2. To evaluate the strings, we use the cost function given in (7). For reproduction, either the generational replacement technique or the steady state reproduction without duplicates is employed depending on the property of the initial population. The generational replacement uses 0.78 and 0.1 for the crossover rate and the mutation rate, respectively, while in the steady state reproduction 1.0 and 0.1 are employed for the crossover rate and the mutation rate. For all the p's, the GA cycles are terminated after 20000 generations.

Once we obtain the results of partitioning, we next identify the parameters of the neuro-fuzzy models using the back propagation algorithm. The associated parameters with the back propagation are set to the same as in above. In this procedure, we initialize the parameters of the neuro-fuzzy model by using (9)-(11). After the parameter identification, we investigate the consistency of the rulebase. Fig. 7 shows the fuzzy partitions after the parameter identification for p=5 and p=7. It can be seen from this figure that the degrees of inconsistency of the models identified by the proposed method are significantly decreased, i.e., $DIC_{rb}(\beta=0.6) = 1$ for p=5 and $DIC_{rb}(\beta=0.6) = 0$ for p=7. When compared with the partitioning results shown in Fig. 6, the partitions obtained by the proposed method are more like a structured knowledge representation in which each fuzzy rule describes each local region.

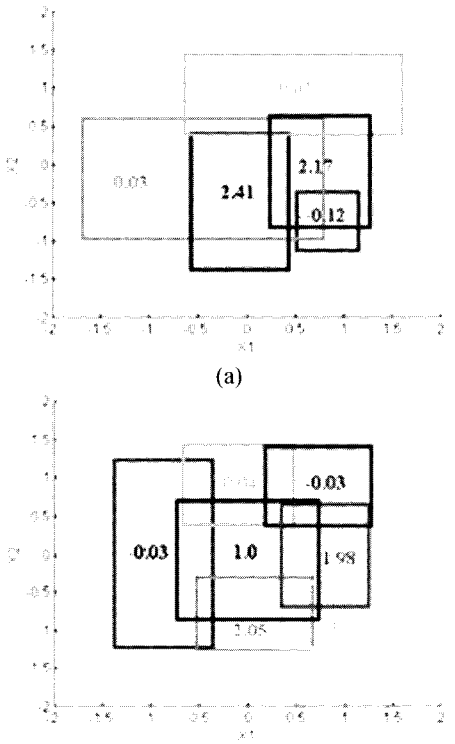


Fig. 7. Results of partitioning after parameter identification (α=0.2). (a) p=5. (b) p=7.

The accuracies of the neuro-fuzzy models for each p in terms of MAE are assessed after the parameter identification. They are listed in Table 2. As expected, the accuracy of the model increases as the number of rules increases in most cases except for p=4. If the given accuracy requirement is not severe, say, MAE=0.03, then the obtained results suggest that the random initialization is a feasible approach. A reasonable number of trials can provide a model of a better performance compared with the model built on the proposed method. However, if the accuracy requirement is getting stricter, for instance, MAE is less than 0.01, then improvement obtained by the proposed method becomes clear. None of 100 models initialized randomly is as accurate as the model for c=2 and p=7.

Table 2. Accuracies of the neuro-fuzzy models identified from the initial parameters obtained by the input space partitioning

	p=2	p=3	p=4	p=5	p=6	p=7
MAE	0.1822	0.0581	0.0808	0.0354	0.0100	0.0086

4.2 Box and Jenkins' Gas Furnace

In this example, we apply the proposed structure identification method to a nonlinear dynamical process modeling using the gas furnace data of Box and Jenkins [9]. The data set consists of 296 pairs of input and output measurements. The input $u(k)$ is the gas flow rate into the furnace and the output $y(k)$ is the concentration of CO_2 in the outlet gas. The sampling interval is 9 seconds.

We are assumed to know that $u(k\text{-}4)$ and $y(k\text{-}1)$ are the input variables and the structure of the process is as follows:

$$y(k) = F_1\big(u(k-4), y(k-1)\big). \tag{19}$$

In this structure, we could generate 292 training samples and performed the fuzzy clustering on the output training data, while setting m to 2 in the FCM algorithm. After clustering, we projected the output clusters into the input space and constructed the cell map.

To run each GA cycle, we set the number of strings in a population to 200. For all the GA cycles, the steady state reproduction without duplicates with 1.0 as the crossover rate and 0.1 as the mutation rate was used. We terminated the GA cycle after 20,000 generations. Each GA cycle for a particular c and p took less than five minutes on a Sun Sparc 20 workstation.

The GA cycles produced the results of partitioning for different c's and different p's. Each result of partitioning from a pair of c and p made us to initialize the parameters of the neuro-fuzzy model of the corresponding structure. Using the initial parameters obtained, we identified the parameters of the model. The mean square error (MSE) was used for measuring the accuracy of the model:

$$e_2 = \frac{1}{N}\sum_{k=1}^{N}\big[y(k) - \hat{y}(k)\big]^2, \tag{20}$$

where N=292. For comparison purpose, we built 50 models with 8 rules using the random initialization scheme. None of them satisfied the accuracy requirement of MSE less than 0.11 in this case, and the best MSE was 0.1148. To examine the degree of inconsistency of these models, we evaluated $DIC_{rb}(\beta$ =0.8)'s by using (17). $DIC_{rb}(\beta$=0.8) of the model with the best MSE was 4 and the average of $DIC_{rb}(\beta$=0.8) of those models was 4.3. A comparison of these models with the model for c=4 and p=8 clearly indicates that the proposed method can build a more accurate model with more consistent rules. In Fig. 8, we show the layouts of the partitioning of the input space for c=4 and p=8 after the parameter identification.

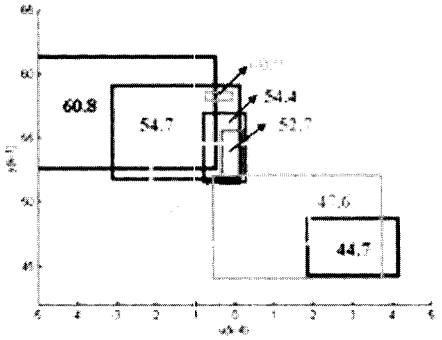


Fig. 8. Partitions of the model (c=4 and p=8) after the parameter identification (α=0.2)

Conclusions

In this paper, we reviewed the structure identification of neuro-fuzzy modeling using genetic algorithms, which guided us to build a better model in terms of the model's efficiency and the model's accuracy. In addition, we addressed the role of structure identification of neuro-fuzzy modeling as a tool to reduce the degree of inconsistency of the fuzzy rulebase learned by the modeling process. Maintaining the measure of inconsistency is especially important when the identified fuzzy rules should be articulated for further uses in practice. In this case, we prefer our neuro-fuzzy model to be more like a structured knowledge representation scheme. In Section 4, we presented two modeling problems in order to justify our claims.

References

[1] M. Sugeno and T. Yasukawa, "A fuzzy-logic-based approach to qualitative modeling," *IEEE Trans. Fuzzy Systems*, vol. 1, no. 1, pp. 7-31, Feb. 1993.

[2] M. Kubat, "Decision trees can initialize radial basis function networks," *IEEE Trans. Neural Networks*, Vol. 9, No. 5, pp. 813-821, Sept. 1998.

[3] Y. J. Park, H. J. Shim, and B. H. Wang, "Short-term electrical load forecasting using neuro-fuzzy models," *The Transaction of KIEE*, Vol. 49A, No. 3, pp. 107-117, 2000.

[4] B. H. Wang and H. J. Cho, "Structure identification of neuro-fuzzy models using genetic algorithms," in *Proc. 8th Int. Symp. Artificial Life and Robotics*, (Oita, Japan) pp. 573-576, Jan. 2003.

[5] A. Lotfi and A. C. Tsoi, "Learning fuzzy inference systems using an adaptive membership function scheme," *IEEE Trans. Syst., Man, and Cybern.*, vol. SMC-26, pp. 326-331, April 1996.

[6] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York: Plenum Press, 1981.

[7] W. Pedrycz, *Fuzzy Control and Fuzzy Systems*. New York: Wiley, 1989.

[8] X. Wang, B. D. Baets, and E. Kerre, "A comparative study of similarity measures," *Fuzzy Sets and Syst.*, vol. 73, pp. 259-268, 1995.

[9] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, San Francisco: Holden Day, 1976.

[10] S. Lee and R. M. Kil, "A Gaussian potential function network with hierarchically self-organizing learning," *Neural Networks*, vol. 4, pp. 207-224, 1991.

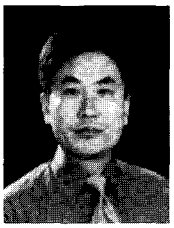
[11] P. D. Wasserman, *Advanced Methods in Neural Computing*, Van Nostrand Reinhold, New York, 1993.

[12] Bien and Yu, "Extracting core information from inconsistent fuzzy control rules," *Fuzzy Sets and Syst.*, vol. 71, no. 1, pp. 95-111, 1995.

[13] L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from example," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-22, no. 6, pp. 1414-1427, 1992.

[14] J. S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-23, no. 3, pp. 665-684, May/June 1993.

저 자 소 개



Bo-Hyeun Wang received the B.S. degree in Electrical Engineering from Yonsei University in 1987 and the M.S. and the Ph.D. degrees in Electrical Engineering both from Georgia Institute of Technology in 1990 and 1991, respectively. From 1991 to 1998, he was a Senior Research Scientist at LG Electronics Research Center in Seoul, Korea. Since 1998, he has been a faculty member of the Department of Electrical Engineering at Kangnung National University. His research interests include artificial intelligence and machine learning.

Phone : +82-33-640-2384
Fax : +82-33-640-2244
E-mail : bhw@kangnung.ac.kr

Hyun-Joon Cho received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Korea, in 1993 and 1995, respectively. From 1995 to 2000, he served as a research engineer at LG Electronics Research Center, in Seoul, Korea. From 2000, he has been working on his Ph.D. degree at Purdue University. His recent research interests include computer network, data mining, and artificial intelligence.