International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:10, No:3, 2016

Learning Algorithms for Fuzzy Inference Systems Composed of Double- and Single-Input Rule Modules

Hirofumi Miyajima, Kazuya Kishida, Noritaka Shigei, Hiromi Miyajima

Abstract—Most of self-tuning fuzzy systems, which are automatically constructed from learning data, are based on the steepest descent method (SDM). However, this approach often requires a large convergence time and gets stuck into a shallow local minimum. One of its solutions is to use fuzzy rule modules with a small number of inputs such as DIRMs (Double-Input Rule Modules) and SIRMs (Single-Input Rule Modules). In this paper, we consider a (generalized) DIRMs model composed of double and single-input rule modules. Further, in order to reduce the redundant modules for the (generalized) DIRMs model, pruning and generative learning algorithms for the model are suggested. In order to show the effectiveness of them, numerical simulations for function approximation, Box-Jenkins and obstacle avoidance problems are performed.

Keywords—Box-Jenkins's problem, Double-input rule module, Fuzzy inference model, Obstacle avoidance, Single-input rule module.

I. Introduction

MANY studies on fuzzy inference systems and their learning methods have been made [1]–[5]. Their aim is to construct self-tuning fuzzy systems from learning data based on SDM. The obvious drawbacks of the method are its large computational complexity and getting stuck in a shallow local minimum. There are two approaches to overcome these drawbacks. One approach is to develop effective learning methods for fuzzy inference systems. In the approach, some methods have been developed which 1) use GA (Genetic Algorithm) and PSO (Particle Swarm Optimization) to determine the structure of the fuzzy model [6], 2) use a self-organization or a vector quantization technique to construct fuzzy inference systems with a small number of rules [7], [8], and 3) use generalized objective functions [9]. The other approach is to develop fuzzy inference models, which are advantageous in terms of computational complexity and global convergence. In such an approach, an SIRMs (Single-Input Rule Modules) model has been proposed that aims to obtain a better solution by using a simple model [10]. Although the SIRMs model can easily apply to the problems with a large number of variables, it does not always show good performance in nonlinear problems [11], [12]. Therefore, SNIRMs (Small Number of Input Rule Modules) models have been proposed as a generalized SIRMs model [11]–[13]. The DIRMs model is one of such models and each module involves two input variables. It has been shown that the DIRMs model outperforms the SIRMs model in terms of accuracy [11]. Using the conventional DIRMs model, we can know only the important pairs of variables but not a single variable. Let us consider how we can know important variables and modules while keeping a high accuracy.

In this paper, we consider a (generalized) DIRMs model composed of double and single-input rule modules. Further, in order to reduce the redundant modules for the DIRMs model, pruning and generative learning algorithms are suggested. In order to show the effectiveness of them, numerical simulations for function approximation, Box-Jenkins and obstacle avoidance problems.

H. Miyajima is with the Graduate School of Science and Engineering, Kagoshima University, 1-21-40 Korimoto, Kagoshima 890-0065, Japan (e-mail: k3768085@kadai.jp).

K. Kishida is with the National Institute of Technology, Kagoshima College, 1460-1 Shinko, Hayato, Kirishima 899-5102, Japan (e-mail: kishida@kagoshima-ct.ac.jp)

N. Shigei and H. Miyajima are with the Graduate School of Science and Engineering, Kagoshima University, 1-21-40 Korimoto, Kagoshima 890-0065, Japan.

II. Fuzzy Inference Model and Its Learning

A. Fuzzy Inference Model

The conventional fuzzy inference model based on steepest descent method (SDM) is described in [1]–[3]. Let $Z_j = \{1, \cdots, j\}$ for the positive integer j. Let $\boldsymbol{x} = (x_1, \cdots, x_m)$ and y be input and output data, respectively, where x_i for $i \in Z_m$ and y are real numbers. Then the rule of simplified fuzzy inference model is expressed as

$$R_j : \text{if } x_1 \text{ is } M_{1j} \text{ and } \cdots \; x_m \text{ is } M_{mj} \text{ then } y \text{ is } w_j, \quad (1)$$

where $j \in Z_n$ is a rule number, M_{ij} is a membership function, and w_j is the weight [2], [3].

A membership value of the antecedent part μ_j for input $\boldsymbol{x}$ is expressed as:

$$\mu_j = \prod_{i=1}^{m} M_{ij}(x_i) \quad (2)$$

Let c_{ij} and b_{ij} denote the center and width values of M_{ij}, respectively.

Using Gaussian membership function, M_{ij} is expressed as:

$$M_{ij} = \exp\left(-\frac{1}{2}\left(\frac{x_j - c_{ij}}{b_{ij}}\right)^2\right) \quad (3)$$

where c_{ij} and b_{ij} are the center and the width parameters, respectively.

In the following, Gaussian function is used as the membership function M_{ij}.

The output y^* of fuzzy inference is calculated by:

$$y^* = \frac{\sum_{j=1}^{n} \mu_j \cdot w_j}{\sum_{j=1}^{n} \mu_j} \tag{4}$$

The objective function E is defined to evaluate the inference error between the desirable output y and the inference output y^*.

$$E = \frac{1}{2}\,(y^* - y)^2 \tag{5}$$

In order to minimize the objective function E, each parameter $\beta \in \{c_{ij}, b_{ij}, w_j\}$ is updated based on SDM [3].

$$\beta(t+1) = \beta(t) - K_\beta \frac{\partial E}{\partial \beta} \tag{6}$$

where t is iteration time and K_β is a constant. Then, the following equations are obtained:

$$\frac{\partial E}{\partial w_j} = \frac{\mu_j}{\sum_{j=1}^{n} \mu_j} \cdot (y^* - y) \tag{7}$$

$$\frac{\partial E}{\partial c_{ij}} = \frac{\mu_j}{\sum_{j=1}^{n} \mu_j} \cdot (y^* - y) \cdot (w_j - y^*) \cdot \frac{x_j - c_{ij}}{b_{ij}^2} \tag{8}$$

$$\frac{\partial E}{\partial b_{ij}} = \frac{\mu_j}{\sum_{j=1}^{n} \mu_j} \cdot (y^* - y) \cdot (w_j - y^*) \cdot \frac{(x_j - c_{ij})^2}{b_{ij}^3} \tag{9}$$

B. The Conventional Leaning Method

In this section, we describe a learning algorithm for the conventional model described in the previous section. The set of learning data $D = \{(x_1^p, \cdots, x_m^p, y_p) | p \in Z_P\}$ is given in advance. The objective of learning is to minimize the following error.

$$E = \frac{1}{P} \sum_{p=1}^{P} (y_p^* - y_p)^2. \tag{10}$$

The conventional learning algorithm is shown below [3].

Learning Algorithm A

Step 1: The initial numbers of rules, c_{ij}, b_{ij} and w_j are set randomly. The threshold Θ_1 for inference error is given. Let T_{max} be the maximum number of learning time. The learning coefficients K_c, K_b and K_w for c_{ij}, b_{ij} and w_j are set.

Step 2: Let $t = 1$.

Step 3: Let $p = 1$.

Step 4: An input and output data $(x_1^p, \cdots, x_m^p, y_p)$ is given.

Step 5: Membership value of each rule is calculated by 2) and (3).

Step 6: Inference output y_p^* is calculated by 4).

Step 7: Real number w_j is updated by (7).

Step 8: Parameters c_{ij} and b_{ij} are updated by (8) and (9).

Step 9: If $p < P$ then $p \leftarrow p + 1$ and go to Step 4.

Step 10: Inference error $E(t)$ is calculated by (10). If $E(t) \leq \Theta_1$ then learning is terminated.

Step 11: If $t \neq T_{max}$ then $t \leftarrow t + 1$ and go to Step 3. Otherwise learning is terminated.

III. The SNIRMs Models

Generalized SNIRMs, SIRMs and DIRMs models are defined. Let U_r^m be the set of all ordered r-tuples of Z_m, that is

$$U_r^m = \{l_1 \cdots l_r | l_i < l_j \text{ if } i < j\}. \tag{11}$$

Then, let $U_{m,k}^* = \bigcup_{r=1}^{k} U_r^m$. Each rule of SNIRMs model for $U_{m,k}^*$ is defined as:

$$\begin{aligned}&\text{SNIRM}{-}l_1 \cdots l_r :\\ &\quad \{R_i^{l_1 \cdots l_r} : \text{if } x_{l_1} \text{ is } M_i^{l_1} \text{ and } \cdots \text{ and } x_{l_r} \text{ is } M_i^{l_r}\\ &\qquad\qquad \text{then } y_{l_1 \cdots l_r} \text{ is } w_i^{l_1 \cdots l_r}\}_{i=1}^{n}\end{aligned} \tag{12}$$

for $1 \leq r \leq k$.

Example 1. For $U_{3,2}^* = U_1^3 \cup U_2^3 = \{1, 2, 3, 12, 13, 23\}$, the obtained system is as follows:

$$\begin{aligned}&\text{SNIRM} - 1 : \{R_i^1 : \text{if } x_1 \text{ is } M_i^1 \text{ then } y_1 \text{ is } w_i^1\}_{i=1}^{n}\\ &\text{SNIRM} - 2 : \{R_i^2 : \text{if } x_2 \text{ is } M_i^2 \text{ then } y_2 \text{ is } w_i^2\}_{i=1}^{n}\\ &\text{SNIRM} - 3 : \{R_i^3 : \text{if } x_3 \text{ is } M_i^3 \text{ then } y_3 \text{ is } w_i^3\}_{i=1}^{n}\end{aligned}$$

$$\begin{aligned}&\text{SNIRM} - 12 :\\ &\{R_i^{12} : \text{if } x_1 \text{ is } M_i^1 \text{ and } x_2 \text{ is } M_i^2 \text{ then } y_{12} \text{ is } w_i^{12}\}_{i=1}^{n}\\ &\text{SNIRM} - 13 :\\ &\{R_i^{13} : \text{if } x_1 \text{ is } M_i^1 \text{ and } x_3 \text{ is } M_i^3 \text{ then } y_{13} \text{ is } w_i^{13}\}_{i=1}^{n}\\ &\text{SNIRM} - 23 :\\ &\{R_i^{23} : \text{if } x_2 \text{ is } M_i^2 \text{ and } x_3 \text{ is } M_i^3 \text{ then } y_{23} \text{ is } w_i^{23}\}_{i=1}^{n}\end{aligned}$$

Let $\boldsymbol{x} = (x_1, \cdots, x_m)$. The output for SNIRM$-l_1 \cdots l_r$ is as:

$$\mu_i^{l_1 \cdots l_r} = M_i^{l_1}(x_{l_1}) \cdots M_i^{l_r}(x_{l_r}), \tag{13}$$

$$y_{l_1 \cdots l_r}^0 = \frac{\sum_{i=1}^{n} \mu_i^{l_1 \cdots l_r} w_i^{l_1 \cdots l_r}}{\sum_{i=1}^{n} \mu_i^{l_1 \cdots l_r}}. \tag{14}$$

In this model, in addition to the conventional parameters c, b and w, the importance degree h is used. Let h_L be the importance degree of each module L.

$$y^* = \sum_{L \in U_{m,k}^*} h_L \cdot y_L^0 \tag{15}$$

where $L = l_1 \cdots l_r$ for $1 \leq r \leq k$.

From (2) to (5), $\frac{\partial E}{\partial \beta}$'s are calculated as:

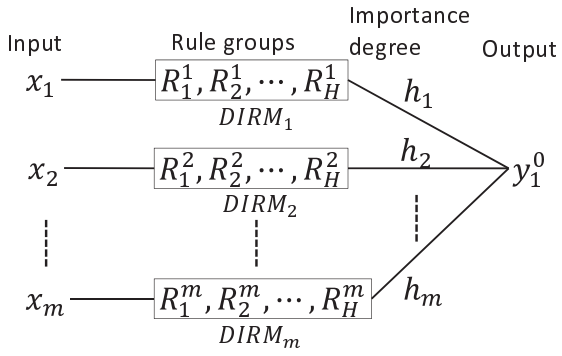
$$\frac{\partial E}{\partial h_L} = (y^* - y) y_L^0, \tag{16}$$

$$\frac{\partial E}{\partial w_i^L} = h_L \cdot \frac{\mu_i^L}{\sum_{i=1}^{n} \mu_i^L} (y^* - y), \tag{17}$$

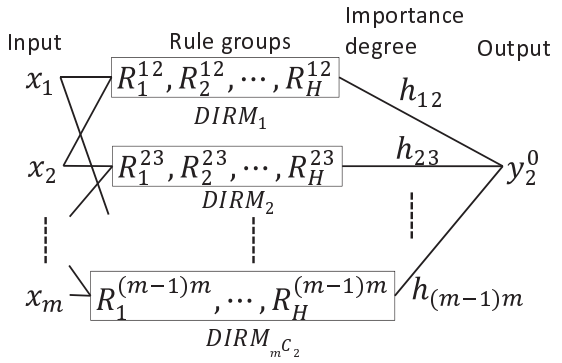
$$\frac{\partial E}{\partial c_i^L} = h_L \cdot (y^* - y) \frac{w_i^L - y_L^0}{\sum_{i=1}^{n} \mu_i^L} \frac{x_i - c_i^L}{(b_i^L)^2} \tag{18}$$

$$\frac{\partial E}{\partial b_i^L} = h_L \cdot (y^* - y) \frac{w_i^L - y_L^0}{\sum_{i=1}^{n} \mu_i^L} \frac{(x_i - c_i^L)^2}{(b_i^L)^3} \tag{19}$$

The cases of $k = 1$ and $k = 2$ are called SIRMs and DIRMs models, respectively. Fig. 1 shows DIRMs model and the conventional DIRMs model [13] has only U_2^m. Therefore, the conventional DIRMs model is a special case of the



(a)DIRMs for U_1^m



(b)DIRMs for U_2^m

Fig. 1 The (generalized) DIRMs model with the output $y^* = y_1^0 + y_2^0$

(generalized) DIRMs models. Examples 1 is DIRMs model for m=3. It is known that the SIRMs model does not always achieve good performance in non-linear systems [11], [12]. On the other hand, when the number of input variables is large, the conventional fuzzy inference model defined equation (11), requires a large time complexity and tends to easily get stuck into a shallow local minimum.

The DIRMs model can achieve good performance in non-linear systems compared to the SIRMs model and is simpler than the conventional fuzzy model.

A learning algorithm for SNIRMs model is given as:

Learning Algorithm B(k)

Step 1: The initial parameters, c_i^L, b_i^L, w_i^L, Θ_1, T_{max}, K_c, K_b and K_w are set, where $L \in U_{m,k}^*$.

Step 2: Let $t = 1$.

Step 3: Let $p = 1$.

Step 4: An input and output data $(x_1^p, \cdots, x_m^p, y_p)$ is given.

Step 5: Membership value of each rule is calculated by (13).

Step 6: Inference output y_p is calculated by (15).

Step 7: Importance degree h_L is updated by (16).

Step 8: Real number w_i^L is updated by (17).

Step 9: Parameters c_i^L and b_i^L are updated by (18) and (19).

Step 10: If $p < P$ then $p \leftarrow p + 1$ and go to Step 4.

Step 11: Inference error $E(t)$ is calculated by (10). If $E(t) < \Theta_1$ then learning is terminated.

Step 12: If $t \neq T_{max}$, $t \leftarrow t + 1$ and go to Step 3. Otherwise learning is terminated.

Note that the numbers of rules for the conventional model, DIRMs and SIRMs models are $O(H^m)$, $O(m^2H^2)$ and $O(mH)$, respectively, where H is the number of partitions for fuzzy inference rules.

The conventional DIRMs model outperforms SIRMs model in terms of accuracy, but not the number of rules. Therefore, we consider a (generalized) DIRMs model. It seems that the (generalized) DIRMs model is superior in interpretability to the conventional DIRMs model, but it also includes the redundant modules (or parameters). Therefore, in order to reduce redundant modules, we suggest pruning and generative learning algorithms as:

Learning Algorithm C (Generative Learning Algorithm for DIRMs Model)

Step 1: Threshold value Θ for learning is set. Algorithm B(1) for U_1^m is performed. Let $U = U_1^m$ and $t = 1$.

Step 2: Select a pair of variables x_{i_1} and x_{i_2} with highest importance degree in U_1^m and add a new module composed of two input variables x_{i_1} and x_{i_2} to the system. If the module composed of x_{i_1} and x_{i_2} is already included in the system, a pair of variables x_{i_3} and x_{i_4} with the next highest importance degree is selected. Let the new module be $\{i_1 i_2\}$.

Step 3: In order to adjust the parameters of the system, algorithm B for $U = U \cup \{i_1 i_2\}$ is performed.

Step 4: Let $E(t)$ be defined by (10). If $|E(t) - E(t-1)| < \Theta$ then go to Step 2 with $t \leftarrow t + 1$, else algorithm terminates.

Learning Algorithm D (Pruning Learning Algorithm for DIRMs model)

Step 1: Threshold value Θ for learning is set. Let $U = U_{m,2}^*$ and $t = 1$.

Step 2: Learning algorithm B for the set U is performed. Let $E(t)$ be defined by (10).

Step 3: If $|E(t) - E(t-1)| \geq \Theta$, then the algorithm terminates.

Step 3: Select one module L_0 with the lowest importance degree in U, set $U = U - \{L_0\}$ and go to Step 2 with $t \leftarrow t + 1$.

IV. Numerical Simulations

In order to show the performance of DIRMs model and pruning and generative learning algorithms, numerical simulations are performed. In IV.A, the relation among suggested, conventional and SIRMs models is shown. In Sections IV B, IV C, and IV D, numerical simulations for function approximation, Box-Jenkins problem, and obstacle avoidance problems are performed, respectively.

A. The Class of DIRMs Model

The suggested model includes the conventional SIRMs and DIRMs models. Further, the following results are obtained.

The EX-OR problem with two variables is defined as:

$$y = x_1 \oplus x_2 \tag{20}$$

where x_1, x_2 and $y \in \{0, 1\}$ and $\oplus$ means the Exclusive OR operation [3]. Then the following result holds.

[Proposition 1] [14] The EX-OR problem cannot be implemented by any SIRMs model.

[Proposition 2] EX-OR problem can be approximated by DIRMs model with U_2^2.
(Proof) Let $\varepsilon > 0$. Then, it is shown that there exists $\delta > 0$ for a DIRMs model such that

$$E = \frac{1}{4}\sum_{p=1}^{4}(y_p^* - y_p)^2 < \varepsilon \tag{21}$$

and DIRMs system is satisfied with the condition.

$$y_p = h_{12}\frac{\sum_{i=1}^{2} M_i^1(x_1)M_i^2 w_i^{12}}{\sum_{i=1}^{2} M_i^1(x_1)M_i^2(x_2)} \tag{22}$$

where $h_{12} = 1$, $c_1^1 = c_1^2 = c_2^2 = c_3^1 = 0$, $c_2^1 = c_3^2 = c_4^1 = c_4^2 = 1$, $b_i^j = \delta$, $w_1^{12} = w_4^{12} = 0$, $w_2^{12} = w_3^{12} = 1$ and δ is sufficiently small. □

Fig. 2 shows the relation between MSE for learning and the number of parameters. As shown in Fig. 2, the MSE for learning is no longer changed where the number of parameters exceeds 248. Therefore, pruning and generative algorithms are needed.

It is not clear if the (generalized) DIRMs model includes truly the conventional DIRMs model.

B. Function Approximation

This simulation uses four systems specified by the following functions with four dimensional input domains $[0, 1]^4$ for (23) and (24) and $[-1, 1]^4$ for (25) and (26).

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{37.21} \times \frac{(4\sin(\pi x_3) + 2\cos(\pi x_4) + 6)}{12} \tag{23}$$

$$y = \frac{(\sin(2\pi x_1) \times \cos(x_2) \times \sin(\pi x_3) \times x_4 + 1.0)}{2.0} \tag{24}$$

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{74.42} + \frac{(4\sin(\pi x_3) + 2\cos(\pi x_4) + 6)}{446.52} \tag{25}$$

$$y = \frac{(2x_1 + 4x_2^2 + 0.1)^2}{74.42} + \frac{(3e^{3x_3} + 2e^{-4x_4})^{-0.5} - 0.077}{4.68} \tag{26}$$

The simulation condition is shown in Table I. The numbers of learning and testing data are 512 and 6400, respectively. All learning and testing data are uniformly and randomly selected from the input space, respectively. In Table II, A, B(1), C and D mean Learning Algorithms A, B(1), C and D. Further, B*(2) means Learning Algorithm B with only U_2^m. Table II shows the results. In each box, three numbers from the top to the bottom show MSE of training($\times 10^{-3}$), MSE of testing($\times 10^{-3}$) and the number of parameters, respectively. The result of simulation is the average value from ten trials.

The results in Table II show that B*(2) and D are the same ability as A in terms of accuracy with the exception of (24), B(1) is inferior in terms of accuracy to other methods and C is inferior in terms of accuracy to D. As a result, D is more effective in terms of accuracy and the number of parameters compared with other methods.

TABLE I
THE INITIAL CONDITIONS FOR SIMULATIONS OF FUNCTION APPROXIMATION

	A	B	C	D
K_c	0.001	0.001	0.001	0.001
K_b	0.001	0.001	0.001	0.001
K_w	0.01	0.01	0.01	0.01
K_h		0.01	0.01	0.01
H	3	3	3	3
T_{max}	50000	50000	50000	50000
Initial c_{ij}	Equal intervals			
Initial b_{ij}	$\frac{1}{2(H-1)} \times (the\ domain\ of\ input)$			
Initial w_{ij}	Random on [0,1]			
Initial h_j		Random on [0,1]		

TABLE II
THE RESULT OF SIMULATION FOR FUNCTION APPROXIMATION

	(23)	(24)	(25)	(26)
A	0.01 0.03 (729)	0.07 0.14 (729)	0.08 0.14 (729)	0.07 0.14 (729)
B(1)	1.60 1.80 (40)	10.69 10.71 (40)	1.40 1.62 (40)	3.34 3.75 (40)
B*(2)	0.02 0.06 (276)	2.29 7.80 (276)	0.01 0.01 (276)	0.01 0.02 (276)
C	0.09 0.12 (283.8)	8.85 9.43 (49.2)	0.05 0.06 (201)	0.06 0.08 (306.8)
D	0.04 0.06 (267)	3.63 4.76 (86.8)	0.03 0.04 (104.6)	0.02 0.03 (128.2)

C. Box-Jenkins Gas Furnace Problem

In order to show the effectiveness of suggested methods, Box-Jenkins problem as one of benchmark ones is used. The problem is a prediction one with six input and one output and is known as one used to compare the ability among conventional fuzzy models [15]–[17]. The data set of Box-Jenkins gas furnace modeling consists of 296 input and output data of a gas furnace process collected using a sampling of 9 seconds [15]. In each sampling instant k, the input x_k is the gas flow into the furnace and the output y_k

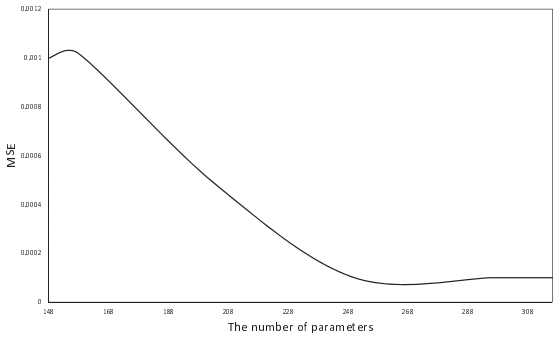


Fig. 2 The relation between MSE for learning and the number of parameters for (23)

TABLE III
SIMULATION RESULT FOR SCENARIO 1

Model	# parameters	# rules	Training Error (MSE)
Box and Jenkins [15]	10	-	0.202
Kim et. al. [16]	38	2	0.055
Pedrycz et. al. [8]	64	8	0.061
B(1)	114	6	0.059
B(2)	1347	23	0.039
D	208	-	0.043

TABLE IV
RESULT FOR SCENARIO 2

Model	# parameters	# rules	Training error (MSE)	Testing error (MSE)
Kim et. al. [16]	38	2	0.034	0.244
Li et. al. [17]	57	3	0.015	0.147
Pedrycz et. al. [8]	32	4	0.022	0.173
B(1)	78	6	0.017	0.161
B(2)	1293	20	0.013	0.197
D	130	-	0.016	0.157

is CO_2 concentration. We employed the suggested technique to construct a fuzzy model with six inputs x_k , x_{k-1}, x_{k-2}, y_{k-1}, y_{k-2}, and y_{k-3}, and one output y_k.

We investigate two scenarios when constructing fuzzy models. In the first scenario (scenario 1), the objective is to evaluate the ability of the suggested method to fit the learning data. In this case all the data are used as the learning data.

In the second scenario (Scenario 2), we consider the first 148 samples as learning data while the remaining data form testing data.

Tables III and IV show the results for scenarios 1 and 2, respectively, where #parameters means the number of parameters. Simulations of them are performed for some systems with different numbers of parameters (or modules) and best results selected from their results are shown. In the scenario 1, the suggested pruning method(D) is superior in terms of accuracy to DIRMs model(B(2)). In the scenario 2, SIRMs model(B(1)) is superior in terms of accuracy to other suggested methods. Note that both DIRMs and SIRMs models are constructed from modules of two and one variables, and compared results are showed as the method with six input rules. Therefore,it is considered that the problem strongly depends on one variable y_{k-1}.

D. Obstacle Avoidance Problem

This problem is also used in the previous paper [13]. As shown in Fig. 3, the distance d_1 and the angle θ_1 between the mobile object and the obstacle and the distance d_2 and the angle θ_2 between the mobile object and the destination are selected as input variables. The problem is to construct a fuzzy inference system that the mobile object avoids the obstacle and arrives at the destination. From learning data (200 points shown in Fig. 4), fuzzy inference rules for the model are constructed. The number of partitions for each model is 3. The mobile object moves with the vector $\mathbf{A} = (A_x, A_y)$ at each step, where A_x is a fixed value and A_y is determined by learning. The simulation condition is shown in Table V. Simulation for learning is successful for Algorithm B(2) and D. Further, following tests are performed.

TABLE V
THE INITIAL CONDITIONS FOR SIMULATIONS OF OBSTACLE AVOIDANCE AND ARRIVING AT THE DESIGNATED PLACE

	D
K_c	0.001
K_b	0.001
K_w	0.05
K_h	0.05
H	3
T_{max}	50000
Initial c_{ij}	Equal intervals
Initial b_{ij}	$\frac{1}{2(H-1)} \times (the\ domain\ of\ input)$
Initial w_{ij}	Random on [0,1]
Initial h_j	Random on [0,1]

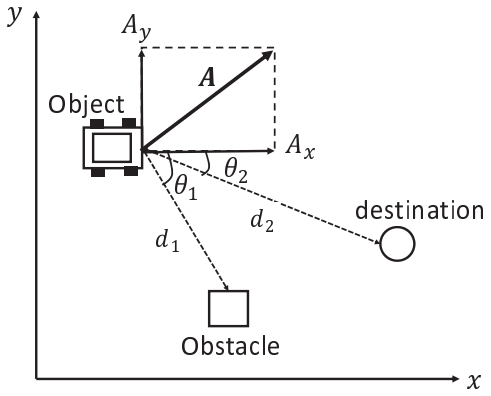


Fig. 3 Simulation on obstacle avoidance and arriving at the goal.

- T1: Test 1 is simulation for obstacle avoidance and arriving at the destination when the mobile object starts from various places. Fig. 5 shows the results of moves of mobile object for starting places at $(0.0, 0.1), (0.0, 0.2), \cdots, (0.0, 0.8), (0.0, 0.9)$. Simulations with the obstacle placed at the place (0.4, 0.4) and arriving at the destination (1.0, 0.6) are

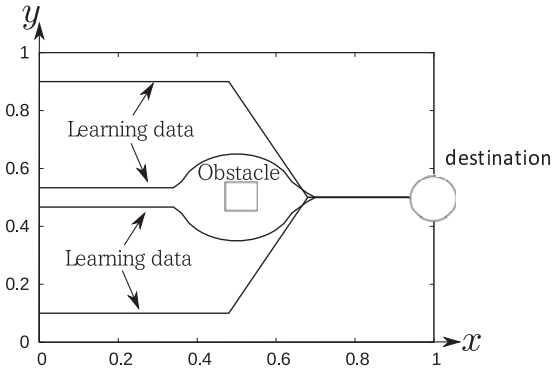


Fig. 4 Learning data to avoid obstacle and arrive at the destination (1.0, 0.5).

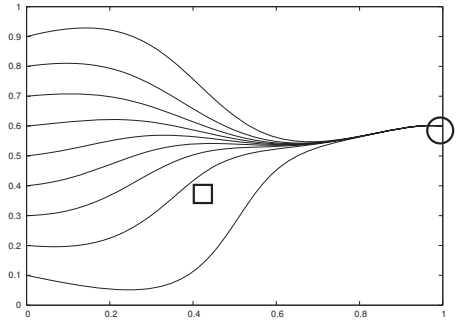


Fig. 5 Simulation for obstacle avoidance with the different destination (1.0, 0.6) from learning.

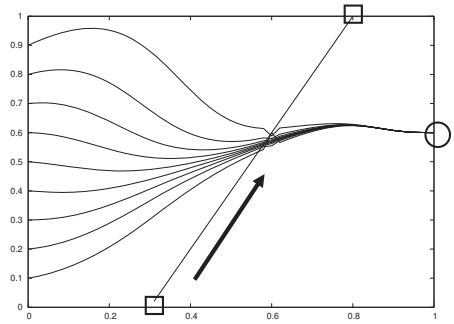


Fig. 6 Simulation for obstacle moving with fixed speed.

performed for suggested model (See Fig. 5). The results are successful as shown in Fig. 5.

- T2: Test 2 is simulation for the case where the obstacle moves with the fixed speed. That is a simulation to avoid the obstacle moving with the vector (0.01, 0.02) at each step, from the initial place (0.3, 0.0) and arrives at the place (0.8, 1.0) at step $T = 50$ as shown in Fig. 6. The destination is (1.0, 0.6). The results are successful as shown in Fig. 6.
- T3: Test 3 is simulation for the case where the obstacle moves randomly as shown in Fig. 7, where $|\boldsymbol{B}|$ is constant, and the θ_b is determined randomly at each step. Simulations with the obstacle moving randomly from the point $(0.5, 0.0)$ are performed. The destination is (1.0, 0.6). The results are successful as shown in Fig. 8.

From this simulation, we can get the following fuzzy rules. Assume that three attributes are "small", "middle" and "large" for d_1, θ_1, d_2 and θ_2, and left($A_y > 0$) and right($A_y < 0$) for the direction of A_y, respectively.

If d_2 is (small or middle), then move to right.
If d_2 is large, then move to left.

If d_1 is (small or middle) and d_2 is small, then move to left.
If d_1 is (small or middle) and d_2 is (middle or large), then move to right.
If d_1 is large and d_2 is small, then move to right.

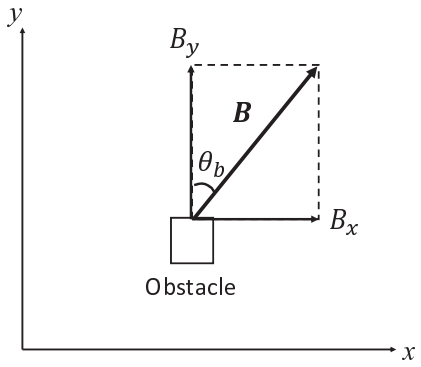


Fig. 7 The obstacle moves with the vector $\boldsymbol{B}$, where $|\boldsymbol{B}|$ is constant and θ_b is selected randomly.

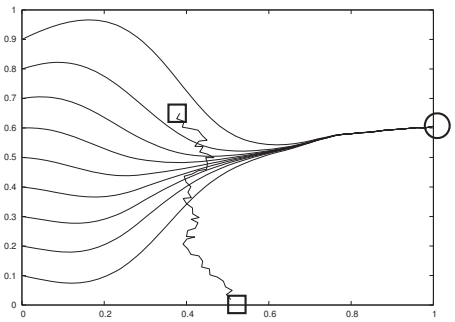


Fig. 8 Simulation for obstacle moving randomly.

If d_1 is large and d_2 is (middle or large), then move to left.

If d_1 is small and θ_2 is (small or middle), then move to right.
If d_1 is small and θ_2 is large, then move to left.
If d_1 is middle and θ_2 is small, then move to left.
If d_1 is middle and θ_2 is (middle or large), then move to right.
If d_1 is large and θ_2 is small, then move to right.
If d_1 is large and θ_2 is (middle or large), then move to left.

We have already performed the same simulations in the previous paper [13]. The simulation for B*(2) are successful and needed almost the numbers of modules 3. On the other hand, the fuzzy inference system composed of two modules of two variables and one module of one variable are constructed when using pruning learning algorithm D.

V. Conclusion

In this paper, we considered the (generalized) DIRMs model and suggested pruning and generative learning algorithms for the model. In order to show the performance of them, numerical simulations for function approximation, Box-Jenkins and obstacle avoidance problems are performed. In particular, it is shown that the model is superior in the number of parameters to the conventional models when using

pruning algorithm. Further, it seems that the same results hold for the cases of the SNIRMs models.

In the future work, we will consider the difference of the ability between the suggested and the conventional DIRMs models.

REFERENCES

[1] B. Kosko, Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence, Prentice Hall, Englewood Cliffs, NJ, 1992.
[2] C. Lin, and C. Lee, Neural Fuzzy Systems, Prentice Hall, PTR, 1996.
[3] M.M. Gupta, L. Jin, and N. Homma, Static and Dynamic Neural Networks, IEEE Press, 2003.
[4] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena, "Accuracy Improvements in Linguistic Fuzzy Modeling," Studies in Fuzziness and Soft Computing, Vol.129, Springer, 2003.
[5] B. Liu, Theory and Practice of Uncertain Programming, Studies in Fuzziness and Soft Computing, Vol.239, Springer, 2009.
[6] O. Cordon, "A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems," Journal of Approximate Reasoning, 52, pp.894-913, 2011.
[7] K. Kishida, H. Miyajima, M. Maeda, and S. Murashima, "A Self-tuning Method of Fuzzy Modeling using Vector Quantization," Proceedings of FUZZ-IEEE'97, pp.397-402, 1997.
[8] W. Pedrycz, and H. Izakian, "Cluster-Centric Fuzzy Modeling," IEEE Trans. on Fuzzy Systems, Vol.22, Issue 6, pp. 1585-1597, 2014.
[9] S. Fukumoto, and H. Miyajima, "Learning Algorithms with Regularization Criteria for Fuzzy Reasoning Model," Journal of Innovative Computing, Information and Control, vol.1, no.1, pp.249-263, 2006.
[10] N. Yubazaki, J. Yi, and K. Hirota, "SIRMS (Single Input Rule Modules) Connected Fuzzy Inference Model," J. Advanced Computational Intelligence, 1, 1, pp.23-30, 1997.
[11] N. Shigei, H. Miyajima, and S. Nagamine, "A Proposal of Fuzzy Inference Model Composed of Small-Number-of-Input Rule Modules," Proc. of Int. Symp. on Neural Networks: Advances in Neural Networks - Part II, pp.118-126, 2009.
[12] S. Miike, H. Miyajima, N. Shigei, and K. Noo, "Fuzzy Reasoning Model with Deletion of Rules Consisting of Small-Number-of-Input Rule Modules," Journal of Japan Society for Fuzzy Theory and Intelligent Informatics, pp.621-629, 2010 (in Japanese).
[13] H. Miyajima, N. Shigei, and H. Miyajima, "An Application of Fuzzy Inference System Composed of Double-Input Rule Modules to Control Problems," Proceedings of the International MultiConference of Engineers and Computer Scientists 2014, Vol I, pp.23-28, 2014.
[14] H. Miyajima, N. Shigei, and H. Miyajima, "Some Properties on Fuzzy Inference Systems Composed of Small Number of Input Rule Modules," Advances in Fuzzy Sets and Systems, Vol.20, pp.155-175, 2015.
[15] G.E.P. Box, G.M. Jenkins, Time series analysis, forecasting and control,second ed., Holden Day, San Francisco, CA, 1970.
[16] E. Kim, M. Park, S. Ji, and M. Park, "A new approach to fuzzy modeling," IEEE Trans. Fuzzy Systems, vol. 5, no. 3, pp. 328-337, 1997.
[17] C. Li, J. Zhou, B. Fu, P. Kou, and J. Xiao, "T-S fuzzy model identification with a gravitational search-based hyperplane clustering algorithm," IEEE Trans. Fuzzy Syst., vol.20, no.2, pp. 305-317, 2012.

Hirofumi Miyajima received the B.E. degree in Electronics and Information Engineering from Hokkaido University, Japan, in 2010, and the M.E. degree in Information Science and Technology from Osaka University, Japan, in 2012. He is currently working toward his Dr. Eng. Degree at Kagoshima University. His current research interests include fuzzy modeling, neural networks and learning algorithms for them.

Kazuya Kishida received the B.E., M.E., D.E. degrees from Kagoshima University, Japan, in 1993, 1995, and 1998, respectively. He is currently an Associate Professor in Electronic Control Engineering at National Institute of Technology, Kagoshima College. His current research interests include neural networks, fuzzy modeling, genetic algorithms.

Noritaka Shigei received the B.E., M.E., D.E. degrees from Kagoshima University, Japan, in 1992, 1994, and 1997, respectively. He is currently an Associate Professor in Graduate School of Science and Engineering at Kagoshima University. His current research interests include neural network, wireless sensor network, digital communication system, digital circuit design, and parallel computing system.

Hiromi Miyajima received the B.E. degree in electrical engineering from Yamanashi University, Japan, in 1974, and the M.E. and D.E. degrees in electrical and communication engineering from Tohoku University, in 1976 and 1979, respectively. He is currently a Professor in Graduate School of Science and Engineering at Kagoshima University. His current research interests include fuzzy modeling, neural networks, quantum computing, and parallel computing.