Short Papers

Intermediate Variable Normalization for Gradient Descent Learning for Hierarchical Fuzzy System

Di Wang, Xiao-Jun Zeng, and John A. Keane

***Abstract*—When applying gradient descent learning methods to hierarchical fuzzy systems, there is great difficulty in handling the intermediate variables introduced by the hierarchical structures, as the intermediate variables may go outside their definition domain that makes gradient descent learning invalid. To overcome this difficulty, this paper proposes a learning scheme that integrates a normalization process for intermediate variables into gradient descent learning. This ensures that gradient descent methods are applicable to, and correctly used for, learning general hierarchical fuzzy systems. Benchmark datasets are used to demonstrate the validity and advantages of the proposed learning scheme over other existing methods in terms of better accuracy, better transparency, and fewer fuzzy rules and parameters.**

***Index Terms*—Fuzzy systems, gradient descent method, hierarchical fuzzy systems, learning.**

I. Introduction

Standard fuzzy systems have been widely and successfully applied in function approximation [1]–[3], system control [4]–[6], classification [7], and clustering [8]. However, when fuzzy systems are applied to more complex and high-dimensional systems, the "curse of dimensionality" becomes increasingly apparent as a bottleneck to their wider application. To overcome this, hierarchical fuzzy systems were proposed in the early 1990s by Raju and Zhou [9] and have attracted much attention in recent years. In hierarchical fuzzy systems, instead of using a standard (flat) high-dimensional fuzzy system, a number of lower dimensional fuzzy subsystems are linked in a hierarchical manner.

The main topics of hierarchical fuzzy systems research have been construction, learning, and analysis (see the survey by Torra [10]). Wang and his colleagues [5], [11] proposed a kind of hierarchical fuzzy system with one fuzzy subsystem in each layer that has one original input and one input from the output of the lower subsystem. They [5], [11] applied the gradient descent method for learning and analyzed the relative importance of input variables as a criterion for the hierarchical structure construction. Although their proposed hierarchical fuzzy systems managed to decrease the exponential growth of fuzzy rules, the exponential growth of parameters remains inherent. Chung and Duan [12] discussed how to design a hierarchical structure based on the correlated or coupled relationship between input variables and showed the applicability of their approach by simulation. Campello and Amaral [13] developed a method to construct hierarchical fuzzy systems by using Gaussian membership functions. Joo and Lee [14], [15] proposed a scheme to construct another kind of general hierarchical fuzzy system, in which the outputs of lower layers are used only in the THEN part instead of the IF part of upper layers. However, their method needs many parameters while obtaining less accurate results. Zeng and Keane [16] investigated the approximation capability and have theoretically shown that hierarchical fuzzy systems can achieve better approximation accuracy with fewer parameters and rules for a large class of systems.

Gradient descent methods are widely used in parameter learning. Despite the aforementioned research results for hierarchical fuzzy systems, there is no effective scheme to handle intermediate variables—a feature that does not occur in gradient descent learning for flat fuzzy systems. However, in order to make the gradient decent learning applicable to, and effective for, hierarchical fuzzy systems, it is necessary and important to handle these intermediate variables properly for two reasons. First, in order to define the membership functions of the upper-layered fuzzy subsystems, it is necessary to know the definition domain of intermediate variables which are the outputs of the lower-layered fuzzy subsystems and the inputs to the upper-layered fuzzy subsystems. Unfortunately, it is impossible to know the domain of intermediate variables before the final hierarchical fuzzy system is determined. Second, after the initial definition of membership functions in the gradient descent learning process, often the intermediate variables go outside their definition domain, which results in no activation, and hence, no parameter updating is done for the current training instance. If many training instances cause no parameter updating in this way, then the gradient descent learning method is invalid. Hence, the purpose of this research is to address this gap by introducing a normalization process to handle intermediate variables in the gradient descent learning method to make it applicable to, and effective for, hierarchical fuzzy systems.

In this paper, a gradient descent learning scheme integrated with the normalization of intermediate variables for general hierarchical fuzzy systems is proposed to overcome the particular difficulty in determining the definition domain of intermediate variables. Further, it is shown that, although some extra errors may be introduced by the normalization process for intermediate variables, such errors can be corrected in the following learning iterations. The proposed learning scheme leads to advantages such as better accuracy, better transparency, and fewer rules and parameters. These advantages are validated both theoretically in the Appendix and experimentally through simulation in Section III.

This paper is organized as follows. The learning algorithm with the normalization of intermediate variable is proposed and validated in Section II. In Section III, the proposed algorithm is applied to benchmark problems to demonstrate its advantages over existing methods for hierarchical fuzzy systems. Then, conclusions are given in Section IV.

II. Gradient Descent Learning With Intermediate Variable Normalization

A. Overview of Gradient Descent Learning for General Hierarchical Fuzzy Systems

In a general multiple-input and single-output (MISO) hierarchical fuzzy system (see Fig. 1), there may be multiple layers and multiple fuzzy subsystems in each layer. Outputs of lower-layered fuzzy subsystems form as inputs to their neighboring upper-layered fuzzy subsystems. These are termed as intermediate variables. The inputs to the lowest layer are all original input variables. The inputs to the lth ($l>1$) layer are the combination of its lower-layered (the $(l-1)$th layer) outputs and some (or none) of the original input variables.

Manuscript received April 27, 2007; revised April 4, 2008 and September 29, 2008; accepted December 16, 2008. First published February 10, 2009; current version published April 1, 2009. This work is supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/C513355/1.

D. Wang is with ThinkAnalytics Ltd., Glasgow G3 7QF, U.K. (e-mail: dwang@thinkanalytics.com).

X.-J. Zeng and J. A. Keane are with the School of Computer Science, University of Manchester, Manchester M13 9PL, U.K. (e-mail: x.zeng@manchester.ac.uk; john.keane@manchester.ac.uk).

Digital Object Identifier 10.1109/TFUZZ.2009.2014940

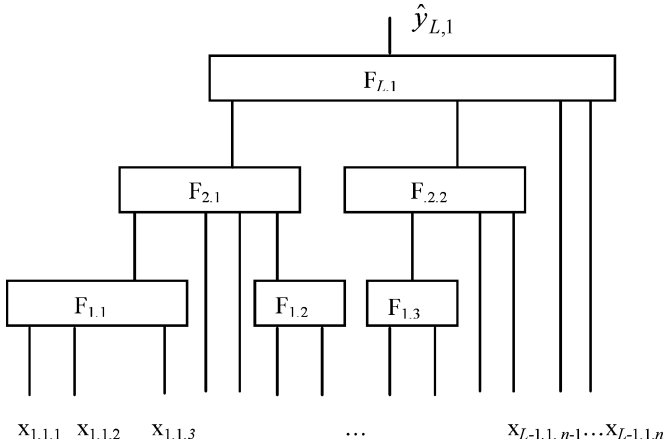

Fig. 1. General structure for hierarchical fuzzy systems.

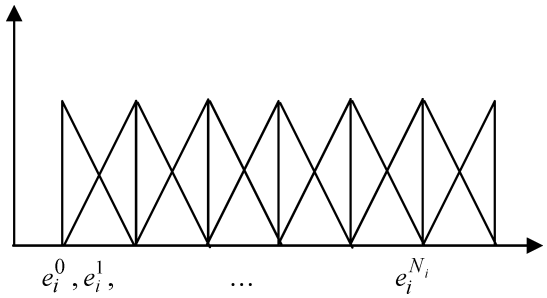

Fig. 2. Example of triangular membership function.

In general, a fuzzy subsystem $F_{l,p}$ (the pth fuzzy subsystem in the lth layer) is represented in the form as

$$\hat{y}_{l,p} = f_{l,p}(\hat{y}_{l-1,p,1}, \hat{y}_{l-1,p,2}, \ldots, \hat{y}_{l-1,p,P_{l-1,p}}, x_{l-1,p,1}, \ldots, x_{l-1,p,Q_{l-1,p}}) \quad (1)$$

where $\hat{y}_{l,p}$ is the output of fuzzy subsystem $F_{l,p}$, $\hat{y}_{l-1,p,j}$ is the output from fuzzy subsystem $F_{l-1,p,j}$ (the jth fuzzy subsystem of the $(l-1)$th layer) to $F_{l,p}$, where $P_{l-1,p}$ is the total number of outputs from the $(l-1)$th layer to $F_{l,p}$, $x_{l-1,p,j}$ is the jth original input variable to $F_{l,p}$, and $Q_{l-1,p}$ is the total number of original input variables to $F_{l,p}$.

Assuming L is the total number of layers (there is only one fuzzy subsystem in the top layer, the Lth layer, for an MISO system), the final output (the output of the Lth layer) is represented by

$$\hat{y} = \hat{y}_{L,1} = f_{L,1}(\hat{y}_{L-1,1,1}, \hat{y}_{L-1,1,2}, \ldots, \hat{y}_{L-1,1,P_{L-1,1}}, x_{L-1,1,1}, \ldots, x_{L-1,1,Q_{L-1,1}}). \quad (2)$$

Uniquely, for the first (lowest) layer, the output of the pth fuzzy subsystem $F_{1,p}$ is

$$\hat{y}_{1,p} = f_{1,p}(x_{0,p,1}, \ldots, x_{0,p,Q_{0,p}}). \quad (3)$$

In the proposed algorithm, for each fuzzy subsystem $F_{l,p}$, a commonly used defuzzifier, center-average defuzzifier, is applied based on Mamdani reasoning [17] (4) as shown at the bottom of this page, where $j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}$ is the index of rules for the $F_{l,p}$, $\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k})$ is the membership function for $\hat{y}_{l-1,p,k}$, $v_{l,p,k}^{i_k}(x_{l-1,p,k})$ is the membership function for $x_{l-1,p,k}$, and $y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}}$ is the parameters to be learned during gradient descent learning.

If

$$U_{l,p} = \prod_{k=1}^{P_{l,1,p}} u_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) \quad (5)$$

and

$$V_{l,p} = \prod_{k=1}^{Q_{l-1,p}} v_{l,p,k}^{i_k}(x_{l-1,p,k}) \quad (6)$$

then (4) can be represented as (7) as follows. For a simpler representation, we use $U_{l,p}$ and $V_{l,p}$ in the remainder of this paper

$$\hat{y}_{l,p} = \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left[\frac{U_{l,p} V_{l,p}}{\sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} U_{l,p} V_{l,p}} \right] \times y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}}. \quad (7)$$

In the aforementioned discussion, $\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k})$ and $v_{l,p,k}^{i_k}(x_{l-1,p,k})$ in (5) and (6) are the membership functions, which can be defined by many types of functions, such as Gaussian, triangular, trapezoid, or bell-shape. Specially, triangular functions are chosen as the membership functions in this paper. Furthermore, the edge of one triangular membership function is chosen to intersect to the middle point of its neighboring triangular membership functions (as shown in Fig. 2). Detailed reasons for the use of triangle membership functions are given in Remark 2.1 at the end of this section. Based on the analysis before, the membership functions $\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k})$ are designed as $\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) : \Gamma_{l-1,p,k} \subset R \to \Gamma_{l,p} \subset R$, where $\Gamma_{l-1,p,k} = [\alpha_{l-1,p,k}, \beta_{l-1,p,k}]$. If $\hat{y}_{l-1,p,k}$ is evenly partitioned with $N_{l-1,p,k}$, then $\Gamma_{l-1,p,k} = [\alpha_{l-1,p,k}, \beta_{l-1,p,k}] = \bigcup_{i=0}^{N_{l,1,p,k}-1} [e_{l-1,p,k}^i, e_{l-1,p,k}^{i+1}]$, and $e_{l-1,p,k}^{j_k+1} - e_{l-1,p,k}^{j_k} = (\alpha_{l-1,p,k} - \beta_{l-1,p,k})/N_{l-1,p,k}$.

$$\hat{y}_{l,p} = f_{l,p}(\hat{y}_{l-1,p,1}, \hat{y}_{l-1,p,2}, \ldots, \hat{y}_{l-1,p,P_{l-1,p}}, x_{l-1,p,1}, \ldots, x_{l-1,p,Q_{l-1,p}}) = \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left[\frac{\prod_{k=1}^{P_{l-1,p}} \mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) \prod_{k=1}^{Q_{l-1,p}} v_{l,p,k}^{i_k}(x_{l-1,p,k})}{\sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \prod_{k=1}^{P_{l-1,p}} \mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) \prod_{k=1}^{Q_{l-1,p}} v_{l,p,k}^{i_k}(x_{l-1,p,k})} \right] y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \quad (4)$$

The triangular membership function $\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k})$ for $\hat{y}_{l-1,p,k}$ is defined as follows:

$$\mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) = \begin{cases} \dfrac{\hat{y}_{l-1,p,k} - e_{l-1,p,k}^{j_k-1}}{e_{l-1,p,k}^{j_k} - e_{l-1,p,k}^{j_k-1}}, & e_{l-1,p,k}^{j_k-1} \le \hat{y}_{l-1,p,k} < e_{l-1,p,k}^{j_k} \\ \dfrac{e_{l-1,p,i}^{j_k+1} - \hat{y}_{l-1,p,k}}{e_{l-1,p,k}^{j_k+1} - e_{l-1,p,k}^{j_k}}, & e_{l-1,p,k}^{j_k} \le \hat{y}_{l-1,p,k} < e_{l-1,p,k}^{j_k+1} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

In a similar way, $v_{l,p,k}^{i_k}(x_{l-1,p,k})$ is defined as

$$v_{l,p,k}^{i_k}(x_{l-1,p,k}) = \begin{cases} \dfrac{x_{l-1,p,k} - e_{l-1,p,k}^{i_k-1}}{e_{l-1,p,k}^{i_k} - e_{l-1,p,k}^{i_k-1}}, & e_{l-1,p,k}^{i_k-1} \le x_{l-1,p,k} < e_{l-1,p,k}^{i_k} \\ \dfrac{e_{l-1,p,i}^{i_k+1} - x_{l-1,p,k}}{e_{l-1,p,k}^{i_k+1} - e_{l-1,p,k}^{i_k}}, & e_{l-1,p,k}^{i_k} \le x_{l-1,p,k} < e_{l-1,p,k}^{i_k+1} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

For the aforementioned membership functions, only 2^n rules are activated each time, where n is the number of input variables. As shown by [3, Th. 2]

$$\sum_{j_1 j_2 \cdots j_{P_{l-1,p}}\, i_1 i_2 \cdots i_{Q_{l-1,p}}} U_{l,p} V_{l,p} = 1 \tag{10}$$

then (7) can be simplified as

$$\hat{y}_{l,p} = \sum_{j_1 j_2 \cdots j_{P_{l-1,p}}\, i_1 i_2 \cdots i_{Q_{l-1,p}}} U_{l,p} V_{l,p} y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}}\, i_1 i_2 \cdots i_{Q_{l-1,p}}}. \tag{11}$$

Remark 2.1: There are several important reasons which motivate the use of triangular membership functions in this paper. First, fuzzy systems with triangular membership functions are more transparent and interpretable than fuzzy systems with other types of membership functions. Second, fuzzy systems with triangular membership functions lead to a much simpler gradient decent learning algorithm for two reasons: one is the much simpler mathematical expression possible, as shown in (11) where the complicated denominator terms in (7) disappear based on (10); the other is that only 2^n rules are activated for each training sample that leads to much simpler calculation in each iteration of the training process. This is significantly different to fuzzy systems with other types of membership functions, especially Gaussian membership functions where all rules are active for each training sample. Due to these two reasons, the corresponding gradient decent algorithm takes much less time to learn and is much quicker to converge. Such simplification and improved performance are especially important for hierarchical fuzzy systems that have a much more complex system structure than (single layered) flat fuzzy systems and are thus much more complicated to learn. Third, fuzzy systems with triangular membership functions are one of the most widely used classes of fuzzy systems and have shown better performance than other types of membership functions in many applications (for example, [19]). In particular, a valuable property of fuzzy systems with triangular membership functions is that they can reproduce (i.e., represent with no error) many important and widely used functions and systems such as linear functions, multilinear functions, and piecewise linear functions, as shown in [18]. Fourth, fuzzy systems with triangular membership functions have the decomposition property [18], i.e., by properly dividing the input space into subinput spaces, a general fuzzy system is decomposed into several fuzzy subsystems that are the simplest fuzzy systems in the subinput spaces. This feature is particular useful in advanced techniques for fuzzy control design and analysis such as piecewise Lyapounov methods [20].

In the remainder of this section, a gradient descent learning scheme integrated with normalization of intermediate variables for general hierarchical fuzzy systems is proposed. For this purpose, it is assumed that the hierarchical structure has been determined, and the discussion will focus on the parameter learning process for general hierarchical fuzzy systems.

B. Normalization for the Intermediate Variables

The definition domain for each variable must be known in order to define triangular membership functions given in (8) and (9). The definition domain of the original input variables can be obtained directly from the training data. However, the possible values of the intermediate variables are unknown. As a result, it is impossible to define the domain of $[\alpha_{l-1,p,k}, \beta_{l-1,p,k}]$ for intermediate variable $\hat{y}_{l-1,p,k}$ in hierarchical fuzzy systems before training.

In Wang's research [11], the domain for the intermediate variable (y_1 in his simulation example) is defined as [0, 1] during training. However, the value of the intermediate variable y_1 is, in fact, not always between zero and one during gradient descent learning. On the contrary, y_1 might be outside of the range [0, 1]. A value of y_1 outside of the range [0, 1] will not activate any membership function. In such cases, the parameters (d_{lij} and $\bar{y}_1^{pq}$ [11]) will not be updated during the gradient descent learning process. As a result, such gradient descent learning is not valid for hierarchical fuzzy systems with triangular membership functions. This might be the reason why Wang's system in [11] is less accurate than ours, despite using more parameters. This is illustrated by simulation comparison in Section III.

To bridge this gap, a normalization process is proposed to overcome this particular difficulty in determining the definition domain of intermediate variables. Our research has found that there is a valuable property related to intermediate variables: if a hierarchical fuzzy system can achieve a desired approximation accuracy within a definition domain $[\alpha_{l-1,p,k}, \beta_{l-1,p,k}]$ for an intermediate variable $\hat{y}_{l-1,p,k}$, then by choosing its definition domain to be any interval $[\alpha'_{l-1,p,k}, \beta'_{l-1,p,k}]$, the same approximation accuracy can be achieved by an associated hierarchical fuzzy system. In other words, the approximation accuracy of hierarchical fuzzy systems is independent of the choice of the definition domains for intermediate variables. As a result, we can always find the optimal hierarchical fuzzy system by restricting the intermediate variables to any definition domain of $[\alpha'_{l-1,p,k}, \beta'_{l-1,p,k}]$. Particularly, in our work, optimal hierarchical fuzzy systems are constructed by restricting the definition domain of all intermediate variables to [0, 1]. A normalization process is introduced to restrict the definition domain of intermediate variables to [0, 1], which is integrated into the gradient descent learning algorithm. We now give the following theorem (Theorem 2.1) that provides the theoretical justification of this normalization process of intermediate variables to [0, 1].

Theorem 2.1 (Approximation Accuracy Preservation Property of Normalization): Define, in a hierarchical fuzzy system $H(X)$, the pth fuzzy subsystem in the lth layer as $F_{l,p}$:$\hat{y}_{l,p} = f_{l,p}(Y_{l-1,p}, X_{l-1,p})$, where $\hat{y}_{l,p}$ is the output, $X_{l-1,p} = (x_{l-1,p,1}, x_{l-1,p,2}, \ldots, x_{l-1,p,Q_{l-1,p}})$ are original inputs to $F_{l,p}$, and $Y_{l-1,p} = (\hat{y}_{l-1,p,1}, \hat{y}_{l-1,p,2}, \ldots, \hat{y}_{l-1,p,P_{l-1,p}})$ are outputs of the connected fuzzy subsystems in the $(l-1)$th layer and $\hat{y}_{l,p,j} \in [\alpha_{l-1,p,i}, \beta_{l-1,p,i}]$ $(i = 1, \ldots, Q_{l,p})$. Under this definition, for a

given function $G(X)$, if there exists a hierarchical fuzzy system $H(X)$ defined by (1)–(4)

$$\begin{aligned} H(X) = f_{L,1}(f_{L-1,1}\{\ldots[f_{2,1}(f_{1,1}(X_{0,1}) \\ ,\ldots,f_{1,P_1}(X_{0,P_1}),X_{1,1}),\ldots]\ldots\} \\ ,\ldots,f_{L-1,P_{L-1}}\{\ldots\},X_{L-1,1}) \end{aligned} \tag{12}$$

such that, for any given input vector X, $|G(X) - H(X)| \leq \varepsilon$ (where ε is a small constant representing the approximation error), then there exists an associated hierarchical fuzzy system $H'(X)$ defined by (1)–(4), where $\hat{y}'_{l,p,j} \in [0,1], \quad (l < L)$, which has the same hierarchical structure as $H(X)$ and is given by

$$\begin{aligned} H'(X) = f'_{L,1}(f'_{L-1,1}\{\ldots[f'_{2,1}(f'_{1,1}(X_{0,1}) \\ ,\ldots,f'_{1,P_1}(X_{0,P_1}),X_{1,1}),\ldots]\ldots\} \\ ,\ldots,f'_{L-1,P_{L-1}}\{\ldots\},X_{L-1,1}) \end{aligned} \tag{13}$$

such that $H(X) = H'(X)$ and $|G(X) - H'(X)| \leq \varepsilon$.

The proof of Theorem 2.1 can be found in Appendix A.

Remark 2.2: As shown in Theorem 2.1, for a given hierarchical fuzzy system $H(X)$ with the approximation error ε, there exists an associated hierarchical fuzzy system $H'(X)$with its all intermediate variables $y_{l,p} = f'_{l,p} \in [0,1](l < L)$ that can achieve the same accuracy as $H(X)$. This is why the aforementioned theorem is called the *approximation accuracy preservation property of normalization.* Further, by the same proof, it can be shown that the theorem still holds if [0, 1] is replaced by any interval. For simplicity, in our research, the definition domain of all intermediate variables is restricted to [0, 1].

C. Learning Based on Gradient Descent Algorithm

In this section, the formula of gradient descent learning for hierarchical fuzzy systems with the integrated normalization process is given.

In gradient descent learning, the error of the final output is propagated back from upper layers to lower ones. The parameter updating of the lower layers is based on the errors propagated back from their upper layers. The objective is to minimize total error.

The error between the actual output $y(k)$ and the model output $\hat{y}_L(k)$ at time k is defined as

$$e_L(k) = \hat{y}_L(k) - y(k). \tag{14}$$

For simplicity, consider two connected fuzzy subsystems in two consequent layers, fuzzy subsystem q and fuzzy subsystem p, where q is the neighboring upper-layered fuzzy subsystem of fuzzy subsystem p (as shown in Fig. 3).

Further, for simplifying the expressions, in this section, the variables in discussion and equations are omitted. For example, we represent $\mu^{j_p}_{q,p}(\hat{y}_{q,p,j},k)$ by $\mu^{j_p}_{q,p}(k)$, where k means "at time k."

Then, the error propagated back from fuzzy subsystem q to p is defined as

$$e_p(k) = e_q(k) \times \frac{\partial \hat{y}_q(k)}{\partial \hat{y}_p(k)} \tag{15}$$

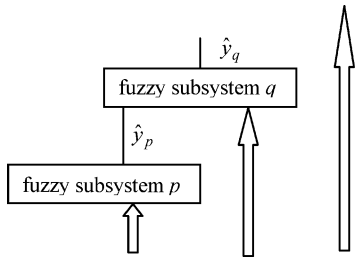

Fig. 3. Illustration of two neighboring fuzzy subsystems.

where $e_p(k)$ is the error of fuzzy subsystem p, which is propagated from its neighboring upper-layered fuzzy subsystem q, and $e_q(k)$ is the propagated error of fuzzy subsystem q.

Similar to (5) and (6), if we define

$$U_q(k) = \prod_{i=1}^{P_q} u^{j_i}_{q,i}(k) \tag{16}$$

$$V_q(k) = \prod_{i=1}^{Q_q} \nu^{j_i}_{q,i}(x_{p,i}) \tag{17}$$

then,

$$\frac{\partial \hat{y}_q(k)}{\partial \hat{y}_p(k)} = \sum_{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} y_q^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} \frac{\partial (U_q(k) V_q(k))}{\partial \hat{y}_p(k)} \tag{18}$$

where

$$\frac{\partial (U_q(k) V_q(k)}{\partial \hat{y}_p(k)} = \frac{U_q(k) V_q(k)}{\mu^{j_p}_{q,p}(k)} \times \frac{\partial (\mu^{j_p}_{q,p}(k))}{\partial \hat{y}_p(k)}. \tag{19}$$

From (5), we have

$$\frac{\partial (\mu^{j_p}_{q,p}(k))}{\partial \hat{y}_p(k)} = \begin{cases} 1, & e_p^{j_p-1}(k) \leq \hat{y}_p(k) < e_p^{j_p}(k) \\ -1, & e_p^{j_p}(k) \leq \hat{y}_p(k) < e_p^{j_p+1}(k) \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Hence, (21) shown at the bottom of this page.

The formula to update the parameters $y_p^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}(k)$ in gradient descent learning is

$$\begin{aligned} & y_p^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}(k+1) \\ & \quad = y_p^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}(k) - \eta \times U_q(k) V_q(k) \times e_p \end{aligned} \tag{22}$$

where η is the learning rate.

As discussed, we aim to assign the definition domain for the intermediate variables as [0, 1]. However, the outputs of a fuzzy subsystem $\hat{y}_{l,p}$ computed by (11) are dependent on the parameters $y_{l,p}^{j_1 j_2 \cdots j_{n_q}}$ and cannot always be guaranteed to be between zero and one. To force the

$$e_p(k) = \begin{cases} e_q(k) \times \displaystyle\sum_{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} \left(\frac{y_q^{j_1, j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}(k) U_q(k) V_q(k)}{\mu^{j_p}_{q,p}(k)} \right), & \text{if} \quad e_p^{j_p-1}(k) \leq \hat{y}_p(k) < e_p^{j_p}(k) \\ e_q(k) \times \displaystyle\sum_{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} \left(\frac{y_q^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}(k) U_q(k) V_q(k)}{\mu^{j_p}_{q,p}(k)} \right), & \text{if} \quad e_p^{j_p}(k) \leq \hat{y}_p(k) < e_p^{j_p+1}(k) \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

intermediate variables to be in the range [0, 1], normalization has to be applied by using (23), shown at the bottom of this page.

After this normalization, all $\hat{y}_{l,p}$ are always between zero and one. This can be formally expressed in Theorem 3.1.

Theorem 3.1: After the normalization process by (23), for any fuzzy subsystem $F_{l,p}$ (i.e., the pth fuzzy subsystem of the lth layer), all the THEN part $y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}$ (say, the $j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i^{\text{th}}_{Q_{q,p}}$ rule of $F_{l,p}$) is between zero and one (i.e., $y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} \in [0,1]$). Thus, the output of $F_{l,p}$, $\hat{y}_{l,p}$, is also between zero and one (i.e., $\hat{y}_{l,p} \in [0,1]$).

The proof of Theorem 3.1 can be found in Appendix B.

However, the normalization process, according to (23), introduces extra error; hence, the parameters may no longer be optimal after normalization. The error introduced by this normalization process can be efficiently corrected in the following gradient descent learning iterations. This is shown by the following theorem.

Theorem 3.2: In the proposed learning algorithm, the extra error introduced in the normalization process [according to (23)] can be eliminated by its following gradient descent training process.

The proof of Theorem 3.2 is given in Appendix C.

III. Simulation and Comparison

In this section, we illustrate the advantages of our proposed method over other methods for hierarchical fuzzy system learning using two simulation examples.

Simulation 1: This is a function approximation problem referred by Wang [11]

$$g(x_1, x_2, x_3) = \frac{1}{1 + \sin^2(\pi x_1) + \sin^2(\pi x_2) + \sin^2(\pi x_3)} \quad \text{on} \quad U = [0,1]^3.$$

For comparison, 6^3 input–output pairs $(x_0^{ijk}; g(x_0^{ijk}))$, where $x_0^{ijk} = [0.2(i-1), 0.2(j-1), 0.2(k-1)]$ for $i, j, k = 1, 2, \ldots, 6$, are created for training [11], and 6^3 uniformly distributed input–output pairs are used for testing. Wang used six equally spaced triangular membership functions over [0, 1] on both the inputs x_1, x_2, x_3, and the intermediate variable.

In Wang's method, there is one fuzzy subsystem in each layer, and there are two inputs to each fuzzy subsystem created by using a grid partition. Hence, there are $(n-1)$ layers altogether, and N^2 fuzzy rules for each layer, where N is the partition for each variable (both the original and intermediate variables) and n is the input dimension (i.e., the total number of the original input variables). In Wang's scheme, the conclusion part for lowest layer is a constant, i.e., the number of parameters for the lowest layer is equal to the number of fuzzy rules N^2. For the other layers, the conclusion part of a fuzzy rule is an $(N-1)$ order polynomial (hence, N parameters for each fuzzy rule) of its lower layer outputs. Therefore, the total number of parameters used in Wang's method is $(n-2) * N^3 + N^2$.

In this simulation, $N = 6$ and $n = 3$, so the total number of parameters used is $6^3 + 6^2 = 252$ for Wang's method. On the contrary, our proposed method is based on Mamdani reasoning and the number of parameters is equal to the number of fuzzy rules, i.e., only $6^2 + 6^2 = 72$ parameters are needed in our proposed scheme for the same hierarchical structure and partitions. Therefore, far fewer parameters are used in our scheme than Wang's, and, in turn, much better accuracy is obtained, as shown in Table I, where it can be seen that by using 72 parameters, we obtain an accuracy of average percentage error (APE) = 0.44067% for the training data, and an accuracy of APE = 2.9721% for the testing data in comparison with 10% by Wang's method [11]. More results in terms of APE by using different number of parameters and fuzzy rules in our scheme are shown in Table I. All these results show that our proposed scheme constructs a hierarchical fuzzy system with far fewer parameters and better accuracy than Wang's method.

TABLE I
Comparison With Wang's Method [11]

Method	Number of membership functions	Number of parameters	Training accuracy (relative error)	Testing accuracy (relative error)
Our scheme	((6,6) 3,6)	54	0.3767%	2.8126%
Our scheme	((6,6) 4,6)	60	0.1153%	2.7475%
Our scheme	((6,6) 5,6)	66	0.1266%	2.8448%
Our scheme	((6,6) 6,6)	72	0.44067%	2.9721%
Wang's scheme	((6,6) 6,6)	252	Not Available	10%

TABLE II
Comparison With Joo's Work 2002 (300 cases)

Method	Number membership of functions	Number of fuzzy rule	Number of parameters	J_u
Our scheme	((2,2),2, 2, 2)	12	12	0.0167
Our scheme	((2,2),2, 3, 3)	22	22	0.0141
Our scheme	((2,2),2, 5, 3)	34	34	0.0046
Our scheme	((2,2),2, 5, 5)	54	54	0.0033
Joo's scheme	((2,2), (2,2) 2,2)	12	20	Not Available
Joo's scheme	((3,3), (3,3) 3,3)	27	45	0.02

Joo and Lee [14], [15] have proposed a hierarchical fuzzy system [14], where the outputs of the previous layer are not used in the IF parts, but only the THEN parts of the fuzzy rules of the current layer. Using Simulation 2 (used in [14] and [15]), we illustrate the better performance of our proposed algorithm over Joo and Lee's approach.

$$y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} = \frac{y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}} - \min\limits_{j_1 j_2 \cdots j_{n_q}} (y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}})}{\max\limits_{j_1 j_2 \cdots j_{n_q}} (y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}}) - \min\limits_{j_1 j_2 \cdots j_{n_q}} (y_{l,p}^{j_1 j_2 \cdots j_{P_{q,p}} i_1 i_2 \cdots i_{Q_{q,p}}})} \tag{23}$$

TABLE III
COMPARISON WITH JOO'S WORK 2005 (3000 CASES)

Method	Number of membership functions	Number of fuzzy rule	Number of parameters	J_u
Our scheme	((2,2),2, 2)	12	12	0.0050
Our scheme	((2,2),2, 3)	22	22	0.0045
Our scheme	((2,2),2, 3)	34	34	0.0024
Our scheme	((2,2),2, 5)	54	54	0.0021
Joo's scheme	((2,2), (2,2) 2,2)	12	20	0.0080
Joo's scheme	((3,3), (3,3) 3,3)	27	54	0.0035

Simulation 2: This simulation is to stabilize a ball and beam control system. The control law obtained is as follows:

$$u*(x)=\frac{4BGx_4\cos x_3+6BG\sin x_3-4x_2-x_1-BGx_4^2\sin x_3}{-BG\cos x_3}$$

with B and G being 0.7143 and 9.81, respectively. So, the task of this simulation is to find a hierarchical fuzzy system to approximate the aforementioned control function.

The simulation results are compared with Joo and Lee's method [14], [15], as shown in Table II [14] and Table III [15] in terms of Ju, which is defined as $J_u=(1/N)\sqrt{\sum_{i=1}^{N}(u_k^*-\hat{u}_k)^2}$, where u_k^* is the kth objective output and $\hat{u}_k$ is the kth model output. Our proposed scheme obtains better accuracy of Ju $=0.0167$ while also using fewer fuzzy rules and fewer parameters (12 fuzzy rules and 12 parameters) than Joo and Lee's method with an accuracy of Ju $=0.02$ (27 fuzzy rules and 45 parameters). When more fuzzy rules are used in our proposed scheme, better accuracy is obtained, e.g., 54 fuzzy rules and parameters result in an accuracy of Ju $=0.0035$. So, this shows that our proposed scheme has better performance than Joo and Lee's method, in terms of both accuracy and the number of parameters.

IV. CONCLUSION

Hierarchical fuzzy systems have been shown by a number of papers to be an effective approach to overcoming the "curse of dimensionality" in flat grid-based fuzzy systems. When applying gradient descent methods to learn hierarchical fuzzy systems, there is a great difficulty in handling the intermediate variables introduced by the hierarchical structures, as the intermediate variables often go outside their definition domain that makes gradient descent learning invalid. In order to overcome this difficulty, this paper has proposed a gradient descent learning scheme integrated with the normalization process of intermediate variables for general hierarchical fuzzy systems.

The main contribution of this paper is the introduction of a normalization process for the intermediate variables during the learning iterations, to make the gradient descent learning method applicable to, and effective for, general hierarchical fuzzy systems. We theoretically proved the validity of this normalization process and verified that the proposed normalization process does not damage the optimization of the final solution. The advantages of the proposed learning scheme over existing methods [5], [11], [14], [15] are shown by benchmark simulations in terms of better accuracy, better transparency, and fewer fuzzy rules and parameters.

Nonetheless, there remain some related and open problems for learning hierarchical fuzzy systems, such as how to identify and learn their associated hierarchical structure. Developments in these areas should result in wider application of hierarchical fuzzy systems and, in turn, help to extend fuzzy systems to successfully solve more complicated and high-dimensional problems.

APPENDIX

A. Proof of Theorem 2.1

To prove Theorem 2.1, a lemma is proved first as follows.

Lemma A.1: Let $F(X)=f_1[f_{0,1}(X_1),f_{0,2}(X_2),\ldots,f_{0,p}(X_p),X_0]$ be a function defined on $X=[X_1,X_2,\ldots,X_p,X_0]\in U=U_1\times U_2\times\cdots\times U_p\times U_0$ and $\hat{y}_i=f_{0,i}(X_i)\in[\alpha_i,\beta_i]$ for $X_i\in U_i$ with $\alpha_i=\min_{X_i\in U_i}f_{0,i}(X_i)$ and $\beta_i=\max_{X_i\in U_i}f_{0,i}(X_i)$ $(i=1,2,\ldots,p)$. Then, there exists an associated function $F'(X)=f_1'[f_{0,1}'(X_1),f_{0,2}'(X_2),\ldots,f_{0,p}'(X_p),X_0]$ defined on $U=U_1\times U_2\times\cdots\times U_p\times U_0$ such that $F(X)=F'(X)$ for all $X\in U$ and $\hat{y}_i'=f_{0,i}'(X_i)\in[0,1]$ for $X_i\in U_i$ with $\min_{X_i\in U_i}f_{0,i}'(X_i)=0$ and $\max_{X_i\in U_i}f_{0,i}'(X_i)=1$ $(i=1,2,\ldots,p)$.

Proof: Define $\hat{y}_i'=f_{0,i}'(X_i)=(f_{0,i}(X_i)-\alpha_i)/(\beta_i-\alpha_i)$ $(i=1,2,\ldots,p)$ for $X_i\in U_i$, then $\hat{y}_i'=f_{0,i}'(X_i)\in[0,1]$ for $X_i\in U_i$ with $\min_{X_i\in U_i}f_{0,i}'(X_i)=0$ and $\max_{X_i\in U_i}f_{0,i}'(X_i)=1$ $(i=1,2,\ldots,p)$.

Now, define

$$f_1'(\hat{y}_1',\hat{y}_2',\ldots,\hat{y}_p',X_0)=f_1[(\beta_1-\alpha_1)\hat{y}_1'+\alpha_1,(\beta_2-\alpha_2)\hat{y}_2'+\alpha_2,\ldots,(\beta_p-\alpha_p)\hat{y}_p'+\alpha_p,X_0]$$

for $Y'=[\hat{y}_1',\hat{y}_2',\ldots,\hat{y}_p']\in[0,1]\times[0,1]\times\cdots\times[0,1]$, then for all $X=[X_1,X_2,\ldots,X_p,X_0]\in U=U_1\times U_2\times\cdots\times U_p\times U_0$,

$$\begin{aligned}F'(X)&=f_1'[f_{0,1}'(X_1),f_{0,2}'(X_2),\ldots,f_{0,p}'(X_p),X_0]\\&=f_1[(\beta_1-\alpha_1)f_{0,1}'(X_1)+\alpha_1,(\beta_2-\alpha_2)f_{0,2}'(X_2)+\alpha_2,\ldots,(\beta_p-\alpha_p)f_{0,p}'(X_p)+\alpha_p,X_0]\\&=f_1\left[(\beta_1-\alpha_1)\frac{f_{0,1}(X_1)-\alpha_1}{\beta_1-\alpha_1}+\alpha_1,(\beta_2-\alpha_2)\frac{f_{0,2}(X_2)-\alpha_2}{\beta_2-\alpha_2}+\alpha_2,\ldots,(\beta_p-\alpha_p)\frac{f_{0,p}(X_p)-\alpha_p}{\beta_p-\alpha_p}+\alpha_p,X_0\right]\\&=f_1[f_{0,1}(X_1),f_{0,2}(X_2),\ldots,f_{0,p}(X_p),X_0]=F(X).\end{aligned}$$

This ends the proof.

Proof of Theorem 2.1: Based on Lemma 1, which is the case for a two-level hierarchical fuzzy system and mathematical induction, Theorem 2.1 can be obtained immediately. Here, we omit the detailed steps to apply mathematical induction due to the space limitation.

B. Proof of Theorem 3.1

Proof of Theorem 3.1: We prove that the conclusion part for any fuzzy subsystem $\hat{y}_{l,p}=f_{l,p}(Y_{l-1,p},X_{l-1,p})$ is between zero and one

(i.e., $\hat{y}_{l,p} \in [0,1]$). First, we have from (4) that

$$\widehat{y}_{l,p} = \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left\{ \frac{U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})}{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})} \times y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \right\} \tag{B1}$$

where $U(\hat{\mathbf{y}}_{l-1,p})$ and $V(\mathbf{x}_{l-1,p})$ are defined as

$$U(\hat{\mathbf{y}}_{l-1,p}) = \prod_{k=1}^{P_{l-1,p}} \mu_{l,p,k}^{j_k}(\hat{y}_{l-1,p,k}) \tag{B2}$$

$$V(\mathbf{x}_{l-1,p}) = \prod_{k=1}^{Q_{l-1,p}} v_{l,p,k}^{i_k}(x_{l-1,p,k}). \tag{B3}$$

For a simple representation, we define a normalization factor as

$$N(\hat{\mathbf{y}}_{l-1,p}, \mathbf{x}_{l-1,p}) = \sum_{j_1 j_2 \cdots j_{n_{l,p}}} U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p}). \tag{B4}$$

Then, the following two inequalities are obtained as shown (B5) and (B6) at the bottom of this page.
This immediately implies $\hat{y}_{l,p} \in [0,1]$.

C. Proof of Theorem 3.2

Proof of Theorem 3.2: Consider the hierarchical fuzzy system given in (4) or (11). To simplify the expression, in the following, we omit the index of layers and the index of fuzzy subsystems in that layer. Then, for a given training instance, the output of any given fuzzy subsystem can be presented as

$$\hat{y} = o = \sum_{r=1}^{R} \left(y^r \times \prod_{p=1}^{m} \mu^r(y_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \right) \tag{C1}$$

where $y_p (p = 1, 2, \ldots, m)$ are the associated intermediate variables, $x_q (q = 1, 2, \ldots, n_0)$ are the original input variables, $\hat{y}$ is the output, y^r is the THEN part of the rth rule, $\mu^r(y_p)$ is the membership of the pth intermediate variable, and $\mu^r(x_q)$ is membership of the qth original variable. There are m intermediate variables and n_0 original input variables for the given subsystem.

Now, define the output for the rth rule for the given fuzzy subsystem as o^r

$$o^r = y^r \times \prod_{p=1}^{m} \mu^r(y_p) \times \prod_{q=1}^{n_0} \mu^r(x_q). \tag{C2}$$

Then,

$$\hat{y} = o = \sum_{r=1}^{R} o^r. \tag{C3}$$

The overall error of the given fuzzy subsystem caused by normalization of the intermediate variables y_p (where $p = 1,\ldots, m$) is

$$\begin{aligned} e &= \sum_{r=1}^{R} \left(y^r \times \prod_{p=1}^{m} \mu^r(\bar{y}_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \right) - \sum_{r=1}^{R} \left(y^r \times \prod_{p=1}^{m} \mu^r(y_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \right) \\ &= \sum_{r=1}^{R} \left\{ y^r \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \left[\prod_{p=1}^{m} \mu^r(\bar{y}_p) - \prod_{p=1}^{m} \mu^r(y_p) \right] \right\} \end{aligned} \tag{C4}$$

where

$$\bar{y}_p = \frac{y_p - \min_p}{\max_p - \min_p}. \tag{C5}$$

If we define the error caused by the rth rule by the normalization of the intermediate variables as e^r

$$e^r = y^r \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \left[\prod_{p=1}^{m} \mu^r(\bar{y}_p) - \prod_{p=1}^{m} \mu^r(y_p) \right]. \tag{C6}$$

Then,

$$e = \sum_{r=1}^{R} e^r. \tag{C7}$$

As (C6) and (C7) show that the overall error e is the sum of the error caused by each fuzzy rule, this implies that, to prove that the overall error caused by the normalization process is eliminated, it is sufficient to prove the error caused by each fuzzy rule is eliminated, which is given later.

$$\begin{aligned} \hat{y}_{l,p} &= \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left\{ \left[\frac{U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})}{N(\hat{\mathbf{y}}_{l-1,p}, \mathbf{x}_{l-1,p})} \right] y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \right\} \\ &\leq \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left\{ \left[\frac{U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})}{N(\hat{\mathbf{y}}_{l-1,p}, \mathbf{x}_{l-1,p})} \right] \max_{j_1 j_2 \cdots j_{n_{l,p}}} \left(y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \right) = 1 \right\} \end{aligned} \tag{B5}$$

$$\begin{aligned} \hat{y}_{l,p} &= \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left\{ \left[\frac{U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})}{N(\hat{\mathbf{y}}_{l-1,p}, \mathbf{x}_{l-1,p})} \right] y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \right\} \\ &\geq \sum_{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \left\{ \left[\frac{U(\hat{\mathbf{y}}_{l-1,p}) V(\mathbf{x}_{l-1,p})}{N(\hat{\mathbf{y}}_{l-1,p}, \mathbf{x}_{l-1,p})} \right] \min_{j_1 j_2 \cdots j_{n_{l,p}}} \left(y_{l,p}^{j_1 j_2 \cdots j_{P_{l-1,p}} i_1 i_2 \cdots i_{Q_{l-1,p}}} \right) \right\} = 0 \end{aligned} \tag{B6}$$

Consider the error caused by the rth rule as given in (C6) and define

$$\Delta = \prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p). \tag{C8}$$

Then, (C6) can be rewritten as

$$\begin{aligned} e^r &= y^r \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \left[\prod_{p=1}^{m} \mu^r(\bar{y}_p) - \prod_{p=1}^{m} \mu^r(y_p) \right] \\ &= y^r \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta \end{aligned} \tag{C9}$$

which implies

$$\frac{\partial o^r}{\partial y^r} = \prod_{p=1}^{m} \mu^r(y_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \tag{C10}$$

$$\begin{aligned} y^r(k+1) &= y^r(k) - \eta \times \frac{\partial o^r}{\partial y^r} \times e^r \\ &= y^r(k) - \eta \times \prod_{p=1}^{m} \mu^r(y_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times e^r. \end{aligned} \tag{C11}$$

Equation (C9) is the error caused by the rth rule of the given fuzzy subsystem and (C11) is used for parameter updating. Based on (C8)–(C11), we consider each of the three possible cases as follows:

1) If $\prod_{p=1}^{m} \mu^r(\bar{y}_p) = 0$ and $\prod_{p=1}^{m} \mu^r(y_p) = 0$, or $\prod_{p=1}^{m} \mu^r(\bar{y}_p) = \prod_{p=1}^{m} \mu^r(y_p) > 0$, i.e., $\Delta = \prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p) = 0$, then based on (C9), $e^r = 0$, i.e., no error is introduced by the normalization for this rule. Further, no action is taken to adjust the parameters, as $y^r(k+1) = y^r(k) - \eta \times (\partial o^r / \partial y^r) \times e^r = y^r(k)$ based on (C11).
2) If $(\prod_{i=1}^{m} \mu^r(y_p) = 0$ and $\prod_{i=1}^{m} \mu^r(\bar{y}_p) > 0)$, or $(\prod_{i=1}^{m} \mu^r(y_p) > 0$, $\prod_{i=1}^{m} \mu^r(\bar{y}_p) > 0$ and $\prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p) > 0)$, then $\Delta = \prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p) > 0$. Then, there are two possible cases that are considered next.
   a) If $y^r(k) > 0$, then from (C9), $e^r(k) > 0$, which implies $\eta \times \prod_{p=1}^{m} \mu^r(\bar{y}_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times e^r(k) > 0$ and $y^r(k+1) < y^r(k)$ from (C11). Given a proper small value of the learning rate η in (C11), then $y^r(k)$ and $y^r(k+1)$ have the same sign, i.e., $y^r(k+1) > 0$. Then, based on (C9), $e^r(k+1) = y^r(k+1) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta > 0$. Further, $\Delta e^r = e^r(k+1) - e^r(k) = [y^r(k+1) - y^r(k)] \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta < 0$ as $y^r(k+1) < y^r(k)$. Based on $e^r(k+1) > 0$, $e^r(k) > 0$, and $e^r(k+1) < e^r(k)$, we have $|e^r(k+1)| < |e^r(k)|$.
   b) Else if $y^r(k) < 0$, then from (C9), $e^r(k) < 0$, which implies $\eta \times \prod_{p=1}^{m} \mu^r(\bar{y}_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times e^r(k) < 0$ and $y^r(k+1) > y^r(k)$ from (C11). Given a proper small value of the learning rate η in (C11), then $y^r(k)$ and $y^r(k+1)$ have the same sign, i.e., $y^r(k+1) < 0$. Then, based on (9), $e^r(k+1) = y^r(k+1) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta < 0$. Further, $\Delta e^r = e^r(k+1) - e^r(k) = [y^r(k+1) - y^r(k)] \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta > 0$ as $y^r(k+1) > y^r(k)$. Based on $e^r(k+1) < 0$, $e^r(k) < 0$, and $e^r(k+1) > e^r(k)$, we have $|e^r(k+1)| < |e^r(k)|$.
3) If $(\prod_{i=1}^{m} \mu^r(y_p) > 0$ and $\prod_{i=1}^{m} \mu^r(\bar{y}_p) = 0)$ or $(\prod_{i=1}^{m} \mu^r(y_p) > 0$, $\prod_{i=1}^{m} \mu^r(\bar{y}_p) > 0$, and $\prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p) < 0)$, then $\Delta = \prod_{i=1}^{m} \mu^r(\bar{y}_p) - \prod_{i=1}^{m} \mu^r(y_p) < 0$. Then, there are two possible cases that are considered next.
   a) If $y^r(k) > 0$, then from (C9), $e^r(k) < 0$, which implies $\eta \times \prod_{p=1}^{m} \mu^r(\bar{y}_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times e^r(k) < 0$ and $y^r(k+1) > y^r(k)$ from (C11). Given a proper small value of the learning rate η in (C11), then $y^r(k)$ and $y^r(k+1)$ have the same sign, i.e., $y^r(k+1) > 0$. Then, based on (C9), $e^r(k+1) = y^r(k+1) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta < 0$. Further, $\Delta e^r = e^r(k+1) - e^r(k) = [y^r(k+1) - y^r(k)] \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta < 0$ as $y^r(k+1) > y^r(k)$. Based on $e^r(k+1) > 0$, $e^r(k) > 0$, and $e^r(k+1) < e^r(k)$, we have $|e^r(k+1)| < |e^r(k)|$.
   b) Else if $y^r(k) < 0$, then from (C9), $e^r(k) > 0$, which implies $\eta \times \prod_{p=1}^{m} \mu^r(\bar{y}_p) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times e^r(k) > 0$ and $y^r(k+1) < y^r(k)$ from (C11). Given a proper small value of the learning rate η in (C11), then $y^r(k)$ and $y^r(k+1)$ have the same sign, i.e., $y^r(k+1) < 0$. Then, based on (9), $e^r(k+1) = y^r(k+1) \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta > 0$. Further, $\Delta e^r = e^r(k+1) - e^r(k) = [y^r(k+1) - y^r(k)] \times \prod_{q=1}^{n_0} \mu^r(x_q) \times \Delta > 0$ as $y^r(k+1) < y^r(k)$. Based on $e^r(k+1) < 0$, $e^r(k) < 0$, and $e^r(k+1) > e^r(k)$, we have $|e^r(k+1)| < |e^r(k)|$.

By combining the conclusion of 1)–3), it is implied that $|e^r(k+1)| \leq |e^r(k)|$, where $e^r(k)$ and $e^r(k+1)$ are errors caused by the normalization process. Then, we say the error introduced by the normalization process for the considered fuzzy subsystem can be corrected in the following training process. This conclusion can then be applied to arbitrary fuzzy subsystems involving intermediate variables.

Remark C.1: The aforementioned proof is based on that, for a given training data point, the error introduced by the normalization for this data point in some iteration will be corrected by the same data point in subsequent iterations. In the practical execution of the proposed algorithm, the error introduced by the normalization of a training data point, in fact, can occur with the same iteration, if, in the training dataset, there are input–output pairs which are very similar to the given data point. This can be verified by using similar arguments to those given in the aforementioned proof.

ACKNOWLEDGMENT

The authors would like to thank the Associate Editor and reviewers for their detailed constructive comments and valuable suggestions that have greatly helped improve the presentation and quality of the paper.

REFERENCES

[1] H. Ying, "Sufficient conditions on general fuzzy systems as function approximators," *Automatica*, vol. 30, pp. 521–525, 1994.
[2] X. J. Zeng and M. G. Singh, "Approximation theory of fuzzy systems—SISO case," *IEEE Trans. Fuzzy Syst.*, vol. 2, no. 2, pp. 162–176, May 1994.
[3] X. J. Zeng and M. G. Singh, "Approximation accuracy analysis of fuzzy system as function approximators," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 1, pp. 44–63, Feb. 1996.
[4] J. J. Buckley, "Universal fuzzy controller," *Automatica*, vol. 28, pp. 1245–1248, 1992.
[5] L. X. Wang, "Fuzzy systems are universal approximators," in *Proc. Conf. Fuzzy Syst.*, San Diego, CA, 1992, pp. 1163–1170.
[6] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modelling and control," *IEEE Trans. Syst., Man Cybern.*, vol. SMC-15, no. 1, pp. 116–132, Feb. 1985.

[7] G. Tsekourasa, H. Sarimveisb, and E. K. George, "A hierarchical fuzzy-clustering approach to fuzzy modelling," *Fuzzy Sets Syst.*, vol. 150, pp. 245–266, 2005.
[8] Y. EI-Sonbaty and M. A. Ismail, "Fuzzy clustering for symbolic data," *IEEE Trans. Fuzzy Syst.*, vol. 6, no. 2, pp. 195–204, May 1998.
[9] G. V. S. Raju and J. Zhou, "Adaptive hierarchical fuzzy controller," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 4, pp. 973–980, Jul./Aug. 1993.
[10] V. Torra, "A Review of the construction of hierarchical fuzzy systems," *Int. J. Intell. Syst.*, vol. 17, pp. 531–543, 2002.
[11] L. X. Wang, "Analysis and design of hierarchical fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 5, pp. 617–624, Oct. 1999.
[12] F. L Chung and J. C. Duan, "On multistage fuzzy neural network modeling," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 2, pp. 125–142, Apr. 2000.
[13] R. J. G. B. Campello and W. C. Amaral, "Optimization of hierarchical neural fuzzy models," in *Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Netw.*, Como, Italy, Jul. 2000, vol. 5, pp. 8–13.
[14] M. G. Joo and J. S. Lee, "Universal approximation by hierarchical fuzzy systems with constraints on the fuzzy rules," *Fuzzy Sets Syst.*, vol. 130, pp. 175–188, 2002.
[15] M. G. Joo and J. S. Lee, "A class of hierarchical fuzzy systems with constrains on fuzzy rules," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 2, pp. 194–203, Apr. 2005.
[16] X. J. Zeng and J. A. Keane, "Approximation capabilities of hierarchical fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 5, pp. 659–672, Oct. 2005.
[17] E. Mamdani, "Advances in the linguistic synthesis of fuzzy controller," *Int. J. Man-Mach. Stud.*, vol. 8, no. 6, pp. 669–678, 1976.
[18] X. J. Zeng and M. G. Singh, "Decomposition property of fuzzy systems and its applications," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 2, pp. 149–165, May 1996.
[19] V. Duraisamy, N. Devarajan, D. Somasundareswari, S. N. Sivanandam. (2004). Comparative study of membership functions for design of fuzzy logic fault diagnosis system for single phase induction motor. *Acad. Open Internet J.* [Online]. Available: http://www.acadjournal.com/2004/V13/Part6/p4/
[20] G. Feng, "A Survey on analysis and design of model-based fuzzy control systems," *IEEE Trans. Fuzzy Syst.*, vol. 14, no. 5, pp. 676–697, Oct. 2006.

Observer-Based Relaxed $\mathcal{H}_\infty$ Control for Fuzzy Systems Using a Multiple Lyapunov Function

Sung Hyun Kim and PooGyeon Park

***Abstract*—This short paper proposes a method of designing a fuzzy observer-based $\mathcal{H}_\infty$ controller for discrete-time Takagi–Sugeno (T–S) fuzzy systems. To enhance the applicability of the output-feedback controller and improve its performance, this short paper first builds a set of fuzzy control rules with premise variables different from those of the T–S fuzzy system, and sets the overall controller to be dependent on not only the current time but also the one-step-past information on the estimated fuzzy weighting functions. Then, based on the fuzzy control rules, this short paper establishes a less conservative $\mathcal{H}_\infty$ stabilization condition incorporated with a multiple Lyapunov function dependent on the estimated fuzzy weighting functions. Through a two-step design procedure, the $\mathcal{H}_\infty$ stabilization condition is formulated in terms of parameterized linear matrix equalities (PLMIs), which are reconverted into LMIs with the help of an efficient and effective relaxation scheme.**

***Index Terms*—Fuzzy weighting-dependent Lyapunov function (FWDLF), $\mathcal{H}_\infty$ performance, observer-based fuzzy control, non-parallel distributed compensation (non-PDC) scheme, parameterized linear matrix inequalities (PLMIs), relaxation scheme.**

I. Introduction

For a systematic control design of nonlinear systems, the Takagi–Sugeno (T–S) fuzzy model [1], [27] has been a popular choice not only in consumer products but also in industrial processes due to its ability to represent the nonlinear system only from input–output data without complex mathematical equations. Thus, based on the T–S fuzzy model, various kinds of fuzzy control methods have been developed under the so-called parallel distributed compensation (PDC) scheme [2], [4], [15]. Recently and notably, one has focused on developing the fuzzy control method associated with a fuzzy weighting-dependent (multiple) Lyapunov function (FWDLF) [11], [20], [24], which is because, for a large number of fuzzy rules, the use of the common quadratic Lyapunov function (CQLF) [2], [4], [22], [25] leads to overconservative design solutions. In spite of this trend, to the best of our knowledge, there has been almost no results using the FWDLF approach when designing fuzzy output-feedback controllers, except for [24].

Practically, all states are not fully measurable; hence, it is necessary in this area to design a fuzzy output-feedback controller, such as static output feedback [14], [22], [25], dynamic output feedback [15], [24], and observer-based output feedback [5], [7], [9], [12], [13], [16], [21], [23], [28]. Especially in the case where the premise variables of the T–S fuzzy system are related to the immeasurable state, one needs to design a fuzzy observer estimating the state. Of course, under the assumption that the premise variables of the fuzzy controller are same as those of the T–S fuzzy system, Ma *et al.* [5] and Yoneyama *et al.* [7] proved the separation principle for the observer-based fuzzy controller; Xiaodong and Qingling [13] proposed two sufficient linear matrix inequalities (LMI) conditions guaranteeing the existence of the observer-based $\mathcal{H}_\infty$ control; and Lo and Lin [16] proposed the method of designing an

Manuscript received February 11, 2008; revised May 30, 2008 and October 2, 2008; accepted December 6, 2008. First published December 22, 2008; current version published April 1, 2009. This work was supported in part by the Ministry of Knowledge Economy, Korea, under the Information Technology Research Center (ITRC) support program supervised by the Institute of Information Technology Advancement (IITA): IITA-2008-C1090-0801-0037 and IITA-2008-C1090-0801-0004.

The authors are with the Division of Electrical and Computer Engineering, Pohang University of Science and Technology, Pohang 790-784, Korea (ie222@postech.ac.kr; ppg@postech.ac.kr).

Digital Object Identifier 10.1109/TFUZZ.2008.2011136