2006 IEEE International Conference on Fuzzy Systems
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Learning for Hierarchical Fuzzy Systems Based on the Gradient-Descent Method

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

***Abstract*—Standard fuzzy systems suffer the "curse of dimensionality" which has become the bottleneck when applying fuzzy systems to solve complex and high dimensional application problems. This curse of dimensionality results in a lager number of fuzzy rules which reduces the transparency of fuzzy systems. Furthermore too many rules also reduce the generalization capability of fuzzy systems. Hierarchical fuzzy systems have emerged as an effective alternative to overcome this curse of dimensionality and have attracted much attention. However, research on learning methods for hierarchical fuzzy systems and applications is rare. In this paper, we propose a scheme to construct general hierarchical fuzzy systems based on the gradient-descent method. To show the advantages of the proposed method (in terms of accuracy, transparency, generalization capability and fewer rules), this method is applied to a function approximation problem and the result is compared with those obtained by standard (flat) fuzzy systems.**

I. INTRODUCTION

Standard fuzzy systems have been applied in many fields, such as approximation [1, 2, 3], control [4, 5], classification [6] and clustering [7]. Indeed, fuzzy systems are universal approximators which can approximate arbitrary continuous functions to any accuracy [1, 2, 3, 4, 8]. The success of these applications is due to the flexibility and expressive ease of fuzzy systems, and the related theoretical results [1, 2, 4, 8] for fuzzy logic obtained during the last four decades.

However, as fuzzy systems have been applied to more complex and high dimensional systems, the "curse of dimensionality" has become increasingly apparent as the bottleneck to wider application. The curse of dimensionality of fuzzy systems is that the total number of rules and parameters, and the data required to identify it increase exponentially with the number of input variables.

In addition, as is well known, an important advantage of fuzzy systems is their transparency and interpretability. Unfortunately, as a consequence of the curse of dimensionality, transparency and interpretability are reduced as humans are incapable of understanding hundreds or thousands of fuzzy rules and parameters. Further, as there is often only limited data available in an application, a large number of rules and parameters result in over-fitting, which reduces the generality of fuzzy systems.

To overcome the curse of dimensionality of fuzzy systems, hierarchical fuzzy systems were proposed in the early 90s by Raju, Zhou and Kisner [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 Sub-Fuzzy Systems (SFS) are linked in a hierarchical manner.

In this paper, a learning algorithm based on the gradient-descent method is proposed to identify general hierarchical fuzzy systems. The algorithm avoids the curse of dimensionality problem and shows advantages in accuracy, transparency and generalization capability. This paper only considers the case of Multiple Inputs and Single Output (MISO). This is without loss of generality as systems with Multiple Inputs and Multiple Outputs (MIMO) can be represented as several MISO systems and solved by using the methods and results based on MISO systems.

This paper is organized as follows: Section II analyses the curse of dimensionality problem in standard fuzzy systems; Section III gives an introduction to hierarchical fuzzy systems; Section IV introduces the proposed hierarchical structure for fuzzy systems and the learning algorithm; Section V applies the proposed learning algorithm to a function approximation problem to show its advantages; Section VI discusses existing difficulties in hierarchical structure design for hierarchical fuzzy systems; finally Section VII presents conclusions. .

II. EXPONENTIAL GROWTH PROBLEMS IN STANDARD FUZZY SYSTEMS

In a standard fuzzy system, a grid-based definition of fuzzy terms and their membership functions is usually used to partition the input space and all possible combinations of these fuzzy terms and their membership functions form the fuzzy rule base. As a consequence, the total number of rules grows exponentially as the number of input variables increases. Fig.1. shows a grid-based definition of the membership functions with two input variables, X_1 and X_2, each having m fuzzy values (A_1^1, A_1^2, …, A_1^m and A_2^1, A_2^2, …, A_2^m). A fuzzy rule can be defined as:

IF X_1 is A_1^i and X_2 is A_2^j, THEN Y is B^{ij}

where i=1…m and j=1…m

Then the total number of fuzzy rules is m^2.

This work is supported by U.K. EPSRC under Grant EP/C513355/1.

The authors are with the School of Informatics, the University of Manchester, Manchester, Sackvill Street, M60 1QD, UK (phone: 44-0161-3062063; fax: 44-0161-3063346; email: di.wang@manchester.ac.uk; x.zeng@manchester.ac.uk; john.keane@manchester.ac.uk).

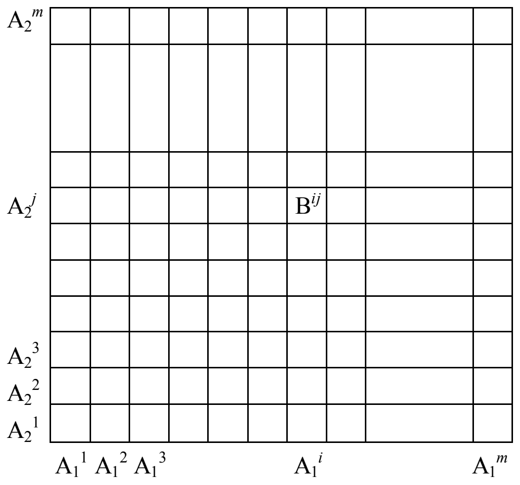

Fig. 1. An example of grid definition for rules and membership functions with two input variables

This set of fuzzy rules covers the entire definition domain and is termed a complete rule set [10]. Assume that there are n variables, and each variable has m membership functions (fuzzy terms), a complete set of rules should have m^n different fuzzy rules. For a problem with 10 input variables each of which has 10 membership functions, there will be 10^{10} fuzzy rules altogether.

The number of parameters also increases exponentially corresponding to the number of input variables. Assume p parameters are needed for each fuzzy rule, a problem with 10 input variables and 10 membership functions for each input variable needs $p \times 10^{10}$ parameters.

The data needed to train standard fuzzy systems also increases exponentially corresponding to the number of input variables [11, 12]. The number of training samples in the training dataset should at least be the same as the number of parameters. As has been discussed that the number of parameters increases exponentially corresponding to the number of input variables; hence the data needed increases exponentially corresponding to the number of input variables as well.

It is infeasible to design a standard fuzzy system for a complex application with a large number of input variables. This exponential growth of fuzzy rules, parameters and data is termed the "curse of dimensionality" problem [9]. In addition, this curse of dimensionality results in several negative consequences:

- Transparency and interpretability (important advantages when fuzzy systems are applied) is reduced as humans are incapable of understanding hundreds or thousands of fuzzy rules and parameters.
- Over-fitting learning often occurs. As there is often only limited data available to an application, a large number of rules and parameters result in over-fitting, which reduces the generality of fuzzy systems. In other words, learning from data becomes difficult or impossible.
- The requirements for more computation and more memory become excessive [12].

III. Hierarchical Fuzzy Systems

To overcome the curse of dimensionality, hierarchical fuzzy systems were proposed in the early 90s by Raju, Zhou and Kisner [9]. In hierarchical fuzzy systems, instead of using a flat high-dimensional fuzzy system, a number of lower-dimensional Sub-Fuzzy Systems (SFS: defined in terms of standard fuzzy systems) are linked in a hierarchical manner. A general structure for hierarchical fuzzy systems is shown in Fig. 2. There may be multiple levels, and multiple sub-fuzzy systems in each level. The outputs of lower levelled sub-fuzzy systems are used as the inputs to their neighbouring upper levelled sub-fuzzy systems. The inputs to the lowest level are all original input variables. The inputs to the lth (l>1) level are the combination of its lower levelled outputs and some (or none) of the original input variables.

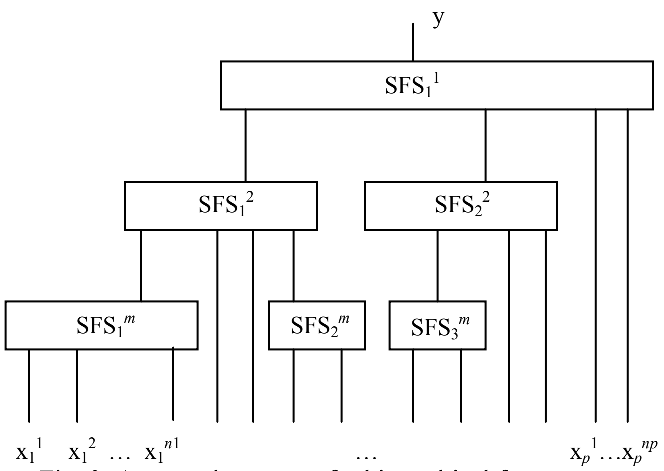

Fig. 2. A general structure for hierarchical fuzzy systems

Some research has focused on the design and analysis of hierarchical fuzzy systems. Wang and colleagues [13] had analyzed the relative importance of input variables and concluded that the most important input variable should be assigned to the lowest level, and that the lest important input variables should be assigned to the highest level. Similar work was also done by Chung and Duan [14]. Another key problem for hierarchical structure design is that correlated or coupled input variables should be assigned to the same sub-fuzzy system. Chung and Duan [14] designed a method to determine the correlated or coupled input variables by introducing a correlation matrix.

Schemes for constructing hierarchical fuzzy systems have been investigated as well. Wang and colleagues [8, 13] proposed a hierarchical structure for fuzzy systems with triangular membership functions, which was proven to be a universal approximator. However, even though Wang and colleagues managed to decrease the exponential growth of fuzzy rules, the exponential growth problem of parameters remains inherent. Campello and Amarel [15] proposed a method to construct hierarchical fuzzy systems by using Gaussian membership functions. In Joo and Lee's work [16], they proposed a general hierarchical fuzzy structure, in which the outputs of the lower levels are used as the THEN part instead of the IF part (the inputs) of the upper levels. However, their scheme involved more parameters to compute in the THEN part for each sub-fuzzy system, which to some degree reduced the transparency of fuzzy systems.

Investigation on learning methods for general hierarchical fuzzy systems and applications is rare. In Section IV, we propose a learning algorithm for general hierarchical fuzzy systems based on the gradient-descent method. In our scheme, we use triangular membership functions, each of which intersects the central point of its neighbouring membership functions. This helps to retain the transparency of fuzzy systems. To show the advantages of the proposed scheme, its result for function approximation is compared with that of standard (flat) fuzzy systems.

IV. LEARNING ALGORITHM FOR HIERARCHICAL FUZZY SYSTEMS

The structure of a hierarchical system should be carefully considered before parameter optimization. Firstly, the input variables should be grouped into sub-groups. Zeng and Keane [17] introduced the concept of a natural hierarchical structure when a function can be decomposed as a composition of several lower dimensional functions or a system consists of several lower dimensional components. If there is a natural hierarchical structure for the function to be approximated, the model hierarchical structure should be consistent with this natural hierarchical structure. Otherwise, the structure of the hierarchical fuzzy system should be analysed by using human knowledge or by the grouping method proposed by Chung and Duan [14].

A major advantage of fuzzy systems is transparency. However, when the number of fired rules increases, the final output is less understandable. We wish fewer rules to fire at one time to keep the fuzzy rules transparent. Reducing the overlaps of different membership functions is an effective method. Gaussian and triangular functions are two popular membership functions used in fuzzy systems. By using Gaussian membership functions more rules fire due to the overlap of different membership functions. Triangular membership functions have comparatively better performance to overcome these overlaps if carefully designed. Specially, if the edge of one triangular membership function does not go across to the middle point of its neighbouring triangular membership functions, then only 2^n rules fire each time, where n is the number of input variables. In this case, the fuzzy rules are quite easy to understand.

A. Formulation for Hierarchical Fuzzy Systems

Based on the discussion above, we use triangular membership function based on Mamdani reasoning [18] in our algorithm. The edge of one triangular membership function intersects to the middle point of its neighbouring triangular membership functions (as shown in Fig. 3).

We consider a function $f: U \subset R^n \to V \in R$. $U = U_1 \times U_2 \times \ldots \times U_n$ is the definition domain, where $U_i \subset R$, and $V \subset R$. Specially, we define $U_i = [\alpha_i, \beta_i]$, where i = 1, 2…n. If each variable is evenly partitioned with N_i, then $U_i = [\alpha_i, \beta_i] = [e_i^0, e_i^1, \ldots, e_i^{N_i}]$. $e_i^{j+1} - e_i^j = (\alpha_i - \beta_i)/N_i$.

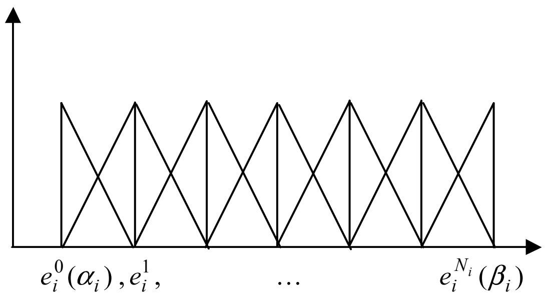

Fig. 3 . An example of triangular membership function

The triangular membership function for the ith input variable of the pth sub-fuzzy system in the lth level, $x_{l,p,i}$, is defined as follows.

If 0< $j_i < N_{l,p,i}$,

$$\mu_{l,p,i}^{j_i} = \begin{cases} (x_{l,p,i} - e_{l,p,i}^{j_i-1})/(e_{l,p,i}^{j_i} - e_{l,p,i}^{j_i-1}), & e_{l,p,i}^{j_i-1} \le x_{l,p,i} < e_{l,p,i}^{j_i} \\ (e_{l,p,i}^{j_i+1} - x_{l,p,i})/(e_{l,p,i}^{j_i+1} - e_{l,p,i}^{j_i}), & e_{l,p,i}^{j_i} \le x_{l,p,i} < e_{l,p,i}^{j_i+1} \\ 0, & othersize \end{cases} \tag{1}$$

where $N_{l,p,i}$ is the number of partitions for $x_{l,p,i}$. $e_{l,p,i}^{j_i}$ is the central point of the j_i–th fuzzy term for $x_{l,p,i}$ and $\mu_{l,p,i}^{j_i}$ is the membership for this fuzzy term.

If j_i =0,

$$\mu_{l,p,i}^{0} = \begin{cases} (e_{l,p,i}^{1} - x_{l,p,i})/(e_{l,p,i}^{1} - e_{l,p,i}^{0}) & e_{l,p,i}^{0} \le x_{l,p,i} < e_{l,p,i}^{1} \\ 0, & otherwise \end{cases} \tag{2}$$

Else if $j_i = N_{l,p,i}$,

$$\mu_{l,p,i}^{N_{l,p,i}} = \begin{cases} (x_{l,p,i} - e_{l,p,i}^{N_{l,p,i}-1})/(e_{l,p,i}^{N_{l,p,i}} - e_{l,p,i}^{N_{l,p,i}-1}) & e_{l,p,i}^{N_{l,p,i}-1} \le x_{l,p,i} < e_{l,p,i}^{N_{l,p,i}} \\ 0, & otherwise \end{cases} \tag{3}$$

In our algorithm, a commonly used defuzzifier, centre-average, is applied as follows.

$$\hat{y}_{l,p} = \frac{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} y_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}}{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{pl}} \mu_{l,p,i}^{j_i}} = \sum_{j_1 j_2 \cdots j_{n_{l,p}}} \left[\frac{\prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}}{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{pl}} \mu_{l,p,i}^{j_i}} \right] y_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}} \tag{4}$$

By using the triangular membership function shown in Fig. 3, the edge of one triangular membership function is the central point of its neighbouring membership functions, hence we can conclude that given an input, only 2^n rules fire each time, where n is the number of input variables. In Equation (4), we have $\sum_{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i} = 1$.

If $B_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}}$ is defined as:

$$B_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}} = \frac{\prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}}{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}} = \prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}$$

Then, Equation (4) can be written as:

$$\hat{y}_{l,p} = \sum_{j_1 j_2 \cdots j_{n_{l,p}}} \left[\frac{\prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}}{\sum_{j_1 j_2 \cdots j_{n_{l,p}}} \prod_{i=1}^{n_{l,p}} \mu_{l,p,i}^{j_i}} \right] y_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}}$$

$$= \sum_{j_1 j_2 \cdots j_{n_{l,p}}} B_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}} y_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}}$$

where for the pth sub-fuzzy system in the lth level, $j_i = 1 \ldots N_{l,p,i}$, $x_{l,p,i}$ is the ith input, and $y_{l,p}^{j_1 j_2 \cdots j_{n_{l,p}}}$ is the THEN part of the $j_1 j_2 \cdots j_{n_{l,p}}$ th fuzzy rule. The triangular membership function can be represented by $\Delta(x_{l,p,i}, e_{l,p,i}^{j-1}, e_{l,p,i}^{j}, e_{l,p,i}^{j+1})$. $\hat{y}_{l,p}$ is the model output of the pth sub-fuzzy in the lth level.

Then the final output is computed by:

$$\hat{y}_L = f_L(\hat{y}_{L-1,1}, \hat{y}_{L-1,2}, \ldots, \hat{y}_{L-1,P_{L-1}}, x_{L-1,1}, \ldots, x_{L-1,Q_{L-1,p}})$$

where L is the total number of levels, $\hat{y}_L$ is the final model output (the output for the Lth level), $\hat{y}_{L-1,j}$ is the jth output from the (L-1)th level (P_{L-1} is the total number of output from the (L-1)th level), $x_{L-1,j}$ is the jth original input variable from the (L-1)th level (Q_{L-1} is the total number of original inputs to the Lth level).

Similarly, the output for the pth sub-fuzzy system in the lth level is:

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

The output for the pth sub-fuzzy system in the 1st level is:

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

B. Learning based on gradient-descent algorithm

For a standard fuzzy system the Least Square Method (LSM) is usually used to gain an optimal result. If the expected intermediate variables (outputs of the lower level and inputs to the upper level) are known, LSM could be applied for training a hierarchical fuzzy system (as done in a standard fuzzy system). However, it is not easy to apply the LSM when developing a hierarchical fuzzy system because in most cases the intermediate variables have no actual meaning and are unexpected. A possible solution is for an expert to supply the required knowledge needed to develop such a hierarchical fuzzy system. Alternatively, machine learning or optimization techniques can be used to design such a system.

In our work, the gradient-descent algorithm is used to optimize the parameters in the hierarchical fuzzy system described in the previous section. The error of the final output is propagated back from the upper level to the lower level. The updating of the parameters of the lower levels is based on these errors propagated back from the upper level.

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

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

and the errors propagated back is defined as:

$$e_p(k) = e_q(k) \times \frac{\partial y_q(k)}{\partial y_p(k)}$$

$$= e_q(k) \times \frac{\partial\left(\sum_{j_1 j_2 \cdots j_{n_q}} B_q^{j_1 j_2 \cdots j_{n_q}}(k) y_q^{j_1 j_2 \cdots j_{n_q}}(k)\right)}{\partial y_p(k)}$$

$$= e_q(k) \times \sum_{j_1 j_2 \cdots j_{n_q}} y_q^{j_1 j_2 \cdots j_{n_q}} \frac{\partial (B_q^{j_1 j_2 \cdots j_{n_q}}(k))}{\partial y_p(k)}$$

$$= e_q(k) \times \sum_{j_1 j_2 \cdots j_{n_q}} \left(\frac{y_q^{j_1 j_2 \cdots j_{n_q}}(k) B_q^{j_1 j_2 \cdots j_{n_q}}(k)}{\mu_{q,p}^{j_p}(k)} \times \frac{\partial (\mu_{q,p}^{j_p}(k))}{\partial y_p(k)} \right)$$

where $e_p(k)$ is the error of sub-fuzzy system p, which is propagated form its neighbouring upper level sub-fuzzy system q. $e_q(k)$ is the propagated error of sub-fuzzy system q.

Because from Equation (1)-(3) we have,

$$\frac{\partial(\mu_{q,p}^{j_p}(k))}{\partial y_p(k)} = \begin{cases} 1, & e_p^{j_{p-1}}(k) \le y_p(k) < e_p^{j_p}(k) \\ -1, & e_p^{j_p}(k) \le y_p(k) < e_p^{j_{p+1}}(k) \\ 0, & otherwise \end{cases}$$

Hence,

$$e_p(k) = \begin{cases} e_q(k) \times \sum_{j_1 j_2 \cdots j_{n_q}} \left(\frac{y_q^{j_1 j_2 \cdots j_{n_q}}(k) B_q^{j_1 j_2 \cdots j_{n_q}}(k)}{\mu_{q,p}^{j_p}(k)} \right), \\ \quad if \quad e_p^{j_{p-1}}(k) \le y_p(k) < e_p^{j_p}(k) \\ -e_q(k) \times \sum_{j_1 j_2 \cdots j_{n_q}} \left(\frac{y_q^{j_1 j_2 \cdots j_{n_q}}(k) B_q^{j_1 j_2 \cdots j_{n_q}}(k)}{\mu_{q,p}^{j_p}(k)} \right), \\ \quad if \quad e_p^{j_p}(k) \le y_p(k) < e_p^{j_{p+1}}(k) \\ 0, \quad otherwise \end{cases}$$

The formula to update the parameters $y_p^{j_1 j_2 \cdots j_{n_p}}(k)$ is

$$y_p^{j_1 j_2 \cdots j_{n_p}}(k+1) = y_p^{j_1 j_2 \cdots j_{n_p}}(k) - \eta \times \frac{\partial y_p(k)}{\partial (y_p^{j_1 j_2 \cdots j_{n_p}}(k))} \times e_p \quad (5)$$

where

$$\frac{\partial y_p(k)}{\partial (y_p^{j_1 j_2 \cdots j_{n_p}}(k))} = B_p^{j_1 j_2 \cdots j_{n_p}}(k)$$

Then Equation (5) can be represented by:

$$y_p^{j_1 j_2 \cdots j_{n_p}}(k+1) = y_p^{j_1 j_2 \cdots j_{n_p}}(k) - \eta \times B_p^{j_1 j_2 \cdots j_{n_p}}(k) \times e_p \quad (6)$$

where η is the learning rate.

C. Learning Algorithms

In this section the learning algorithm is given as follows:

Step 1) Choose the membership functions for each input variables: both the original input variables and the intermediate variables (the output of the lower level sub-fuzzy systems which are inputted to the upper level sub-fuzzy systems). To do this, we assign an even partition for each input variable on their corresponding definition domain. The definition domain for the original input variables can be directly gained from the training set. The definition domain for the intermediate variables might have no actual meaning and no explicit definition domain; we could just simply assign the definition domain for the intermediate variables as [0, 1].

Step 2) Choose initial parameters $y_{l,p}^{j_1 j_2 \cdots j_{n_q}}$ for the *p*th sub-fuzzy system of the *l*th level randomly. These parameters will be adjusted in the following steps.

Step 3) In one learning iteration *k*, for a given input-output pair ($\boldsymbol{x}^r$, y^r), where *r* is the index of training data items, update the parameters $y_{l,p}^{j_1 j_2 \cdots j_{n_q}}$ using Equation (6).

Step 4) If $r<N$, go to Step 3) with $r=r+1$, where *N* is the total number of data items in the training set.

Step 5) Compute the accumulated error $E = \frac{1}{2} \times \sum_{r=1}^{N} (\hat{y}^r - y^r)^2$. If *E* is less than a prespecified small value ε, or the iteration *k* is larger than the prespecified maximal iterations *K*, then end the training process. Else, go to Step 3 with $k = k + 1$.

Remark: it should be noted that, triangle membership functions are not differential at their apex and edge. However, in the gradient descent learning process, the optimisation process is based on gradient descent computing, so we can still get the parameters adjusted to their optimal values.

V. SIMULATION

In this section we apply our algorithm to a function approximation problem. Here we consider a function with three input variables (x_1, x_2, x_3): $y = f(x_1, x_2, x_3) = (1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2$ on $U = [1,6]^3$. We compare our result with that from standard (flat) fuzzy system to show the advantages of our algorithm for hierarchical fuzzy systems.

For simulation, 341 samples are uniformly created on the definition domain $U = [1,6]^3$, 216 of which are for training and the remaining 125 are for testing. These three input variables are grouped as ((x_1, x_2) x_3)), then $y = f_2(f_1(x_1, x_2), x_3)$. The hierarchical structure is shown in Fig. 4.

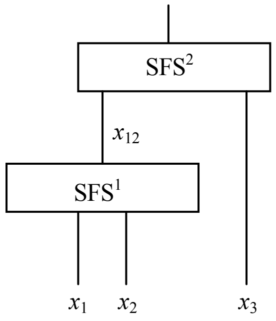

Fig. 4. The hierarchical structure for the function approximation of $y = f(x_1, x_2, x_3) = (1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2$

We train the hierarchical fuzzy system by setting different values of partitions and learning rate. The result comparison is shown in Table 1. The error is reported and also referred to in the following discussion in terms of APE%.

Firstly, we partition each variable into 5 subspaces (for both the original input variables: x_1, x_2 and x_3, and the intermediate variable: x_{12}), and then there are 6 membership functions for each variable defined by Equation (1)-(3). The learning rate is set as: $\eta = 0.0004$. The training error is $0.8655173\,\%$, and the testing error is 1.4829928% from using 6×6+6×6=72 rules. If the learning rate is set as: $\eta = 0.0005$, the training error is 0.86328%, and the testing error is 1.4932056% by using 72 rules. If we use a standard (flat) fuzzy system, by partitioning each variable into 5 subspaces, the number of rules is 6×6×6=216, and the training error is quite low: $0.076629606\,\%$. Theoretically speaking, the training error should be zero (for 216 samples are used to train the model with 216 parameters), but it is infeasible to always find the optimal solution, and it is almost the optimal one with error 0.076629606%. The testing error is 6.5011896% from using a standard fuzzy system, which is much larger than the testing result from the hierarchical fuzzy system.

When we partition each variable into 4 subspaces (for both the original input variables: x_1, x_2 and x_3, and the intermediate variable: x_{12}), and then there are 5 membership functions for each variable defined by Equation (1)-(3). The learning rate is set as: $\eta = 0.0004$. The training error is $1.1753109\,\%$, and the testing error is 1.7112675% from using 5×5+5×5=50 rules. When the learning rate is set as: $\eta = 0.0005$. The training error is $1.1867441\,\%$, and the testing error is 1.7046347%. Similarly, if we use a standard (flat) fuzzy system, by using partition each variable into 4 subspaces, the number of rules is 5×5×5=125, and the training error is: 0.5859885 % and the testing error is 6.379889%, which is much larger than the testing result from the hierarchical fuzzy system.

We can conclude from the above that hierarchical fuzzy systems have a better generalization capability than standard fuzzy systems. The reason for this is analyzed in detail in the next section.

Hierarchical fuzzy systems also obtain higher accuracy by using a similar number of fuzzy rules to standard fuzzy systems. By using 72 rules (the 4th and 5th row), a hierarchical fuzzy system gains a training accuracy of 0.86, which is a litter worse than a standard fuzzy system (0.58) with about double (125) the number of rules. In addition, from the comparison of the 6th and 7th row and the 3th row, we find that the hierarchical fuzzy system obtains better accuracy (1.17) by using fewer fuzzy rules (50) than a standard fuzzy system (the 3th row, 1.4 accuracy for training and 2.98 accuracy for testing by using 64 rules). So we can say that hierarchical fuzzy systems have higher accuracy and fewer rules than standard fuzzy systems.

Fig. 5 and Fig. 6 show the comparison between the model output and the actual output for the training data and the testing data respectively by using the partition of (5, 5, 5) and the learning rate is set as 0.0004. We can see that the hierarchical fuzzy systems constructed by our algorithms approximate the objective function quite well.

Fig. 7 and Fig. 8 show the error for each sample in the training set and the testing set respectively.

TABLE I
RESULT COMPARISON BETWEEN HIERARCHICAL FUZZY SYSTEM AND STANDARD FUZZY SYSTEM

Method	partitions	Rules	Parameters	Learning rate	APE % training	APE% testing
standard	(5,5,5)	216	216	LSM	0.076629606	6.5011896
standard	(4,4,4)	125	125	LSM	0.5859885	6.379889
standard	(3,3,3)	64	64	LSM	1.4044434	2.9811306
hierarchical	(5,5,5)	72	72	0.0004	0.8655173	1.4829928
hierarchical	(5,5,5)	72	72	0.0005	0.86328	1.4932056
hierarchical	(4,4,4)	50	50	0.0004	1.1753109	1.7112675
hierarchical	(4,4,4)	50	50	0.0005	1.1867441	1.7046347
hierarchical	(3,3,3)	32	32	0.0005	1.9707694	2.6047569
hierarchical	(3,3,3)	32	32	0.0004	1.9702787	2.5857594

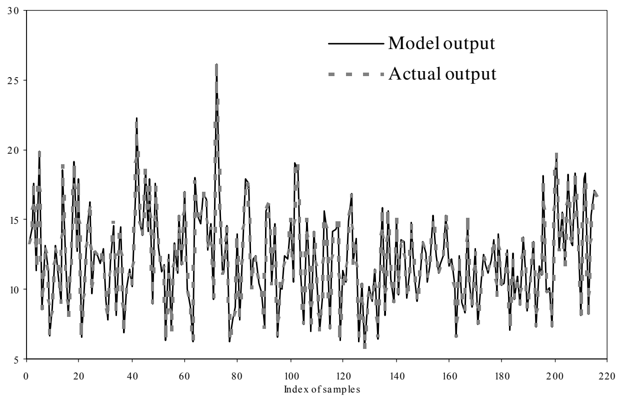

Fig. 5. Comparison between the model output and the actual output for the training set

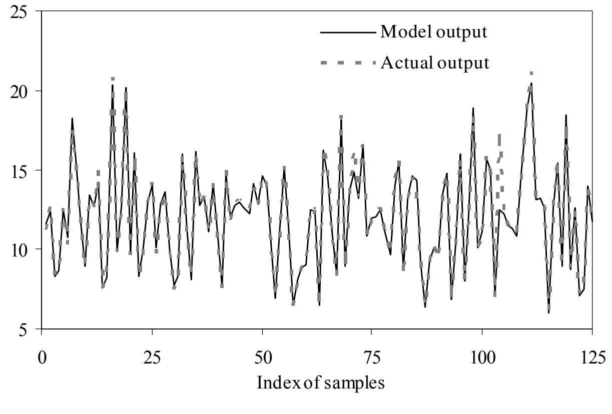

Fig. 6. Comparison between the model output and the actual output for the testing set

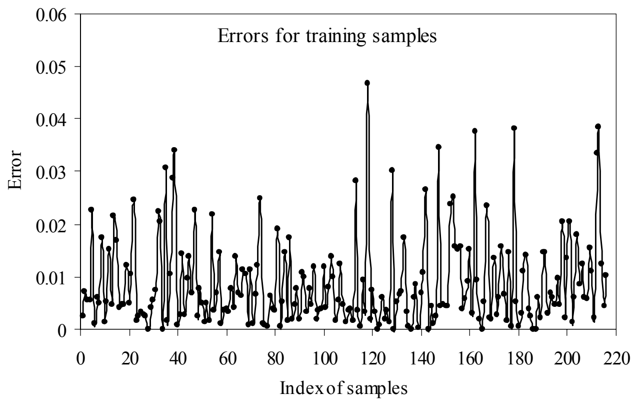

Fig. 7. Error for the training set

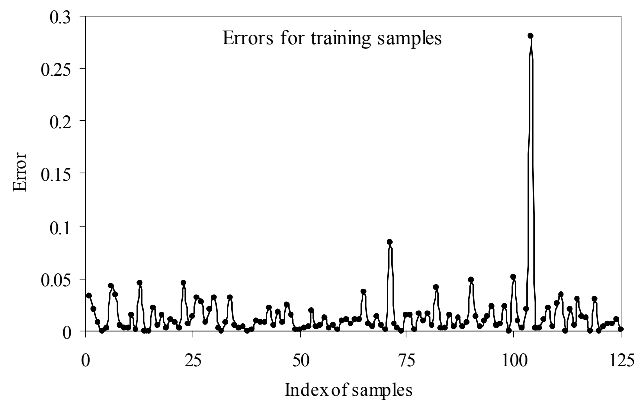

Fig. 8. Error for the testing set

VI. DISCUSSION ON HIERARCHICAL FUZZY SYSTEMS

A. Intermediate variables

Hierarchical structures introduce intermediate variables, which are the outputs of the lower level sub-fuzzy systems, and are propagated to the upper level sub-fuzzy systems. If a task can be decomposed into sub-tasks, the application itself has a natural hierarchical structure. The resultant hierarchical fuzzy system should have the same hierarchical structure as the problem structure. In this case, the intermediate variables possess some actual meaning and it is easy to design such a hierarchical fuzzy system. Unfortunately, in most applications, it is difficult to decompose a main task into sub-tasks. In these applications, the intermediate variables usually do not possess any actual meaning and consequently make hierarchical fuzzy systems hard to design.

In addition, these intermediate variables without 'actual meaning' also reduce the transparency of fuzzy systems. Hierarchical fuzzy systems give a set of multi-stage rules as:

IF X_1 is A_1 and …and X_n is A_n, THEN Y_1 is B_1 and …and Y_m is B_m

IF Y_1 is B_1 and …and Y_m is B_m, THEN Z is C

If the intermediate variables: Y_1 … Y_m have no actual meaning, the multi-stage reasoning becomes less understandable. Furthermore, fewer partitions of the intermediate variables will cause information loss.

The defuzzification process in a standard fuzzy system transforms the fuzzy conclusion into a crisp (usually numerical) output value. Some information is lost during this dufuzzification process [19]; in the design of a hierarchical fuzzy system, defuzzification of lower sub-fuzzy systems causes some information loss. This information loss will be propagated to the upper sub-fuzzy systems. This information loss introduced by the intermediate variables is termed the spread of fuzziness or fuzzy explosion [19]. The spread of fuzziness in multi-stage (hierarchical) fuzzy systems has been extensively studied by Maeda and colleagues [19] and Babuska [20]. This propagation of the fuzzy explosion in hierarchical fuzzy systems makes the final solution less understandable. The more modules are considered, the more uncertainty it is likely.

Unfortunately, to date little work has been done investigating the intermediate variables in hierarchical fuzzy structures. Such developments should enable hierarchical fuzzy systems to be used in wide areas.

B. Generalization capability

Hierarchical fuzzy systems have better generalization performance than standard fuzzy systems for which, as shown in Section 2, the data needed increases exponentially corresponding to the number of input variables. The reason is that the number of parameters increases exponentially corresponding to the number of input variables and meanwhile the data needed is proportional to number of parameters.

In real world applications, it is hard to obtain a sufficient data set with many input variables that cover the whole input space. In most cases there are limited samples to train the fuzzy systems. Given a fixed number of training samples and a fixed number of membership functions for each input variable, the number of parameters needed by a hierarchical fuzzy system (polynomial growth corresponding to the number of input variables) is much smaller than for a standard fuzzy system (exponential growth corresponding to the number of input variables). A standard fuzzy system (supported by more parameters) is more liable to over-fit than a hierarchical fuzzy system (supported by fewer parameters). In other words, hierarchical fuzzy systems reduce the number of fuzzy rules and parameters and this reduction prevents the whole model from over-fitting. Therefore, the obtained fuzzy systems have better generalization capability.

The better generalization capability of hierarchical fuzzy systems can be explained in another way. The accuracy of the approximation for the testing data depends on the density of the observed data samples in the input space. Suppose there are *n* input variables, each of which has *m* membership functions. Then the whole input space will be partitioned into $(m-1)^n$ sub-spaces for a complete fuzzy set. There should be at least one input sample (or more to gain better performance) located in each sub-space. That is, the number of training samples should be proportional to m^n. Otherwise, if the number of training samples is less than m^n, there will be some sub-input spaces with no training samples . An unexpected result will be obtained if the testing sample is located in these sub-input spaces with no training data. On the contrary, in hierarchical fuzzy systems, suppose each intermediate variable also has *m* membership functions, the whole input space will be partitioned into $(m-1)^{n_1}$ sub-spaces for a complete fuzzy set in the first (lowest) level, and will be partitioned into $(m-1)^{n_2}$ sub-spaces for a complete fuzzy set in the second level, and so on, where n_1 and n_2 are the number of inputs to the first and second level. The whole input space will be partitioned into $(m-1)^{n_i}$ sub-spaces for a complete fuzzy set in the *i*th level. Altogether, for a hierarchical fuzzy system with *L* level, the whole input space is partitioned into $\max_{i=1}^{L}(m-1)^{n_i}$ sub-spaces for a complete fuzzy system? for all levels, which is much smaller than $(m-1)^n$. The probability of a sub-space having no training data in a hierarchical fuzzy system is much smaller than the associated probability in a standard fuzzy system given the training set and partitions for each input variable. Therefore, when using hierarchical fuzzy systems, less data is needed for description of the input-output relationships in the subspace.

Hierarchical fuzzy systems reduce the number of fuzzy rules and parameters used in standard fuzzy systems and this reduction prevents the whole model from over-fitting. The obtained fuzzy systems therefore have better generalization capability.

VII. Summary

In this paper, we first reviewed the "curse of dimensionality" problem in standard fuzzy systems. That is, in complex applications with a large number of input variables, the number of rules, parameters and data need to increase exponentially according to the number of input variables; hence it is simply often infeasible to define a standard fuzzy system for such a complex problem. During recent years, hierarchical fuzzy systems have been investigated and validated both theoretically and practically as an effective alternative in overcoming the "curse of dimensionality" in many areas.

Despite this progress, research on learning methods for hierarchical fuzzy systems and their applications is rare. In this paper, we propose a scheme to construct general hierarchical fuzzy systems based on the gradient-descent technique.

From the function approximation application in Section 5, we conclude that hierarchical fuzzy systems are an effective alternative to solve this curse of dimensionality problem. Hierarchical fuzzy systems gain better performance than standard fuzzy systems in terms of accuracy by using the same number of, or fewer fuzzy rules and parameters to gain the same accuracy.

Hierarchical fuzzy systems have better performance than standard fuzzy systems in terms of their generalization capability. The reason is discussed theoretically in Section 6.2. In addition, from the result in Section 5, we also observe that hierarchical fuzzy systems have better performance in this aspect.

In our research, to retain the transparency advantage of fuzzy logic, we apply triangular membership functions. Triangular membership functions have comparatively better performance to overcome the overlaps between different membership functions if carefully designed. Specifically, if the edge of one triangular membership function does not go across to the middle point of its neighbouring triangular membership functions, then only 2^n rules fire each time, where n is the number of input variables. The fewer rules fire, the more understandable a fuzzy system is. In this case, the fuzzy rules are quite easy to understand.

However, there are many open problems which require further investigation for hierarchical fuzzy systems, such as how to determine the hierarchical structure for fuzzy systems; how to handle intermediate variables; and how to train hierarchical fuzzy systems efficiently. Developments in each of these areas should result in wider application of hierarchical fuzzy systems and, in turn, help to extend fuzzy systems to address more complicated and high dimensional problems.

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 Transaction on Fuzzy Systems*, Vol.2, No.2, pp.162-176, 1994.

[3] X. J. Zeng and M. G. Singh, "Approximation accuracy analysis of fuzzy system as function approximators", *IEEE Transactions on Fuzzy Systems*, Vol.4, No.1, pp.44-63, 1996.

[4] J. J. Buckley, "Universal fuzzy controller", *Automatica*, Vol.28, pp.1245-1248, 1992.

[5] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modelling and control", *IEEE Transactions on Systems Man and Cybernetics*, Vol.15, pp.116-132, 1985.

[6] G. Tsekourasa, H. Sarimveisb and E. K. George, " Ahierarchical fuzzy-clustering approach to fuzzy modelling", *Fuzzy Sets and Systems*, Vol.150, pp.245–266, 2005.

[7] Y. EI-Sonbaty and M. A.,Ismail, "Fuzzy clustering for symbolic data", *IEEE Transactions on Fuzzy Systems*, Vol.6, No.2, pp.195-204, 1998.

[8] L. X. Wang, "Fuzzy systems are universal approximptors", *in Proceedings of Conference on Fuzzy System*s, pp.1163-1170, San Diego,1992.

[9] G. V. S. Raju and J. Zhou, "Adaptive hierarchical fuzzy controller", *IEEE Transactions on System Man and Cybernetics*,Vol.23, No.4, pp. 973-980, 1993.

[10] X. J. Zeng and M. G. Singh, "Decomposition property of fuzzy systems and its applications", *IEEE Transactions on Fuzzy Systems*, Vol.4, No.2, pp.149-165, 1996.

[11] S. Nakayama, T. Furuhashi and Y. Uchikawa, "A proposal of hierarchical midelling", *Journal of Japan Society Fuzzy Theory Systems*, Vol.1, No.5, pp.1155-1168, 1993.

[12] W. Rattasiri and S. K. Halgamuge, "Computationally advantageous and stable hierarchical fuzzy systems for active suspension", *IEEE Transactions on Industrial Electronics*, Vol.50, No.1, pp. 48-61, 2003.

[13] L. X. Wang, "Analysis and design of hierarchical fuzzy systems", *IEEE Transactions on Fuzzy systems*, Vol.7, No.5, pp.617-624, 1999.

[14] F. L Chung and J. C. Duan, "On multistage fuzzy neural network modeling", *IEEE Transactions on fuzzy systems*, Vol.8, No.2, pp.125-142, 2000.

[15] R. J. G. B. Campello and W. C. Amaral, "Optimization of hierarchical neural fuzzy models", in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, Vol.5, pp.8–13, Como, Italy, July 2000.

[16] M. G. Joo and J. S. Lee, "A class of hierarchical fuzzy systems with constrains on fuzzy rules", *IEEE Transactions on Fuzzy Systems*, Vol.13, No. 2, pp.194- 203, 2005.

[17] X. J. Zeng and J. A. Keane, "Approximation capabilities of hierarchical fuzzy systems", *IEEE Transactions on Fuzzy Systems*, Vol.13, No.5, pp. 659-672, 2005.

[18] E. Mamdani "Advances in the linguistic synthesis of fuzzy controller," *Int. J. Man-Machine Studies,* vol.*8,* no. 6, pp. 669-678, 1976.

[19] H. Maeda, "An investigation on the spread of fuzziness in multi-fold multi-stage approximation reasoning by pictorial representation — under sup-min composition and triangular type membership function", *Fuzzy Sets and Systems*, Vol.80, pp.133-148, 1996.

[20] R. Babuska, "Construction of fuzzy system interplay between precision and transparency", in *ESIT 2000*, 12-15 Sep. Aachen, Genmany.