ISSN 2278-5612

Development of Multiple Neuro-Fuzzy System Using Back-propagation Algorithm

Dr. Imad S. Alshawi[1], Md. Haider Khalaf Jabbar[2], Dr. Rafiqul Zaman Khan[3]
[1]Assistant Professor
Department of Computer Science,
College of Science, Basra University, Basra, Iraq
emadalshawi@gmail.com
[2]Research Scholar,
University of Misan, Misan, Iraq and
Department of Computer Science
Aligarh Muslim University, Aligarh-202002 (U.P), INDIA
haider.allamy@gmail.com, he25_allamy@yahoo.com
[3]Associate Professor,
Department of Computer Science,
Aligarh Muslim University, Aligarh-202002 (U.P), INDIA
rzk32@yahoo.co.in, rzkhan.cs@mail.amu.ac.in

Abstract

When fuzzy systems are highly nonlinear or include a large number of input variables, the number of fuzzy rules constituting the underlying model is usually large. Dealing with a large-size fuzzy model may face many practical problems in terms of training time, ease of updating, generalizing ability and interpretability. Multiple Fuzzy System (MFS) is one of effective methods to reduce the number of rules, increase the speed to obtain good results. This paper is therefore proposes another approach call Multiple Neuro-Fuzzy System (MNFS) which can further enhance the performance of the MFS approach. The new approach is used Back-propagation algorithm in the learning process. The performance of the proposed approach evaluates and compares with MFS by three experiments on nonlinear functions. Simulation results demonstrate the effectiveness of the new approach than MFS with regards to enhancement of the accuracy of the results.

Indexing terms/Keywords

Multiple fuzzy system, neuro-fuzzy systems, fuzzy system, back-propagation algorithm.

Academic Discipline And Sub-Disciplines

Computer science, Expert System and Artificial Intelligence

SUBJECT CLASSIFICATION

Artificial Intelligence, neuro-fuzzy system

TYPE (METHOD/APPROACH

New Approach For Development

Council for Innovative Research

Peer Review Research Publishing System

Journal: International Journal of Management & Information Technology

Vol. 6, No. 2
editor@cirworld.com
www.cirworld.com, member.cirworld.com

1. INTRODUCTION

Fuzzy rule-based models have been shown to be powerful tools in the modeling and control of complex nonlinear systems. When a fuzzy system being considered is highly nonlinear or includes a number of input variables, the number of fuzzy rules constituting the underlying model is usually large. Dealing with such model with large size may pose many practical issues in terms of training time, ease of updating, generalizing ability, and interpretability [1] [2] [3] [4]. Therefore, researchers have spent lot of efforts to find new methods to decrease the number of rules.

Optimization of fuzzy systems can be done from different perspectives, such as tuning membership parameters (e.g. center and width), optimizing learning rates, and compressing the number of rules. J. A. Dickerson used a fast simulated annealing to reduce the number of fuzzy rules [5]. E. Kolman and M. Margaliot proposed a fuzzy rule base with a special structure, referred to the Fuzzy All-permutations Rule Base (FARB) [6]. M. Mizumoto and Y. Shi presented a learning algorithm for tuning fuzzy rules [7]. The main advantage of such method is that the fuzzy rules can be tuned without changing the form of fuzzy rule table, so that the case of weak-firing ease can be avoided. C. Yeh considered the least-squares multiple regressions with fuzzy data [8]. The regression coefficients are assumed to be real. Z. Chen *et al.* presented architecture and learning algorithm based on the use of rules, preliminarily the rules must be tailored to quantum processing. Cheu *et al*. suggested a neural-fuzzy system with rules generated from fuzzy grid partitions created by data space partitioning based on a decision for ANCFIS, the inductive-learning architecture that employs complex fuzzy logic and rule interference [9]. M. Panella *et al*. investigated the application of nonlinear quantum processing to Neuro-Fuzzy networks [10]. Since these networks are tree classification algorithm [11]. J. Yen *et al*. presented an alternative methodology for designing fuzzy systems called the Multiple Fuzzy System (MFS) [12]. The essential scheme for this method is to decompose the overall system into subsystems and then combine their individual results. The idea of decomposition is well developed in the area of modeling and control of large scale complex systems where the main difficulty arises from the high dimensionality of the problem, and one approach in tackling such problems is to reduce the dimensionality by a suitable decomposition.

Although previous works have achieved wonderful results in this area, there is still the need to improve upon the efficiency of the fuzzy system to obtain more accurate results with high speed. In this paper, we improve the performance of the MFS by utilizing the concept of neuro-fuzzy system, called Multiple Neuro-Fuzzy System (MNFS). The training process is done by using the back-propagation algorithm for all parameters in each subsystem (the parameters in the IF-part and the Then-part), in addition to the training for the integrating unit's weights. The proposed method may have some advantages from different perspectives: *(i)* High speed, each neuro-fuzzy subsystem computing simultaneously to produce the total output of the whole system. *(ii)* Results of the combined system are of high reliability. *(iii)* Simplicity of design, where each neuro-fuzzy subsystem is separate from others and can be seen as a small system that contains fewer membership functions.

To show the increased efficiency of MNFS, we test it by several experiments on nonlinear functions. Results prove the improvement of MNFS over MFS. The rest of the paper is organized as follows. Sec. 2 presents the architecture and identification algorithm of MNFS. Experimental results are given in Sec. 3. Conclusion and discussion are presented in Sec. 4.

1.1. Multiple Neuro-Fuzzy System (MNFS)

The basic scheme of MNFS is to decompose the overall system into subsystems. Each subsystem is considered as a neuro-fuzzy system, and their individual results are combined to generate the final output. This approach has been used to overcome the complexity problem encountered when the number of inputs of the system is significantly increased.

1.2. Architecture of MNFS

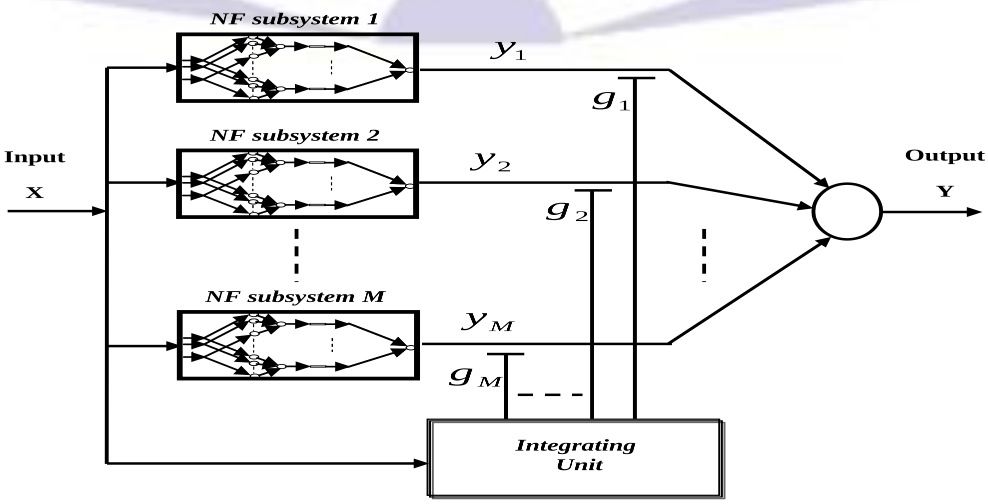


Figure 1. Architecture of MNFS (y_i : output of the subsystem *i*, and g_i: output of the integrating unit; i=1:m)

Figure 1 shows the configuration of the MNFS, which consists of *M* neuro-fuzzy subsystems and an integrating unit. In this method, each subsystem is a neuro-fuzzy system obeying a Mamdani model [13, 14]. This structure can be viewed as a special case of *modular network*. Each subsystem amounts to "local expert" in the modular network and the integrating unit is used to coordinate the separate outputs of subsystems and acts like a decision switch. Therefore the MNFS structure can be regarded as a combined neuro-fuzzy system.

The structure of the S^{th} subsystem is determined by the functions used to represent the fuzzy sets. A general layout of the S^{th} subsystem with multi-inputs and one output is shown in Figure 2. The architecture of this network is analogous to that of artificial neural network with four-layers [2] [13] [14] [20].

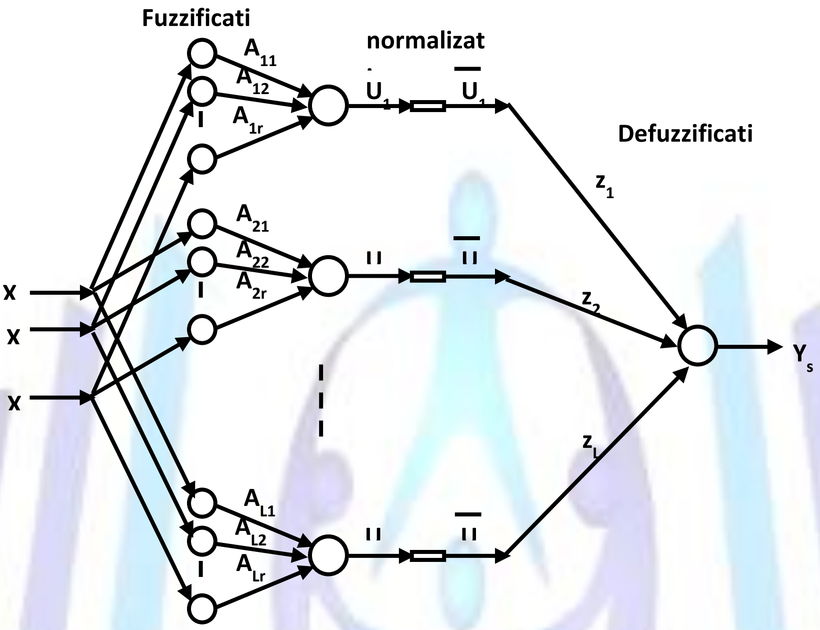


Figure 2. The S^{th} subsystem of MNFS in Figure 1.

In the first layer, the fuzzification operator for each linguistic variable is performed using Gaussian membership functions, which are given in the following:

$$A_{ij}\left(x_j\right) = e^{-\frac{1}{2}\left(\frac{x_j - a_{ij}}{b_{ij}}\right)^2}. \tag{1}$$

where x_j is the j^{th} input variable; a_{ij}, and b_{ij} are the center and the width for the Gaussian membership function, respectively.

The outputs of the first layer are fed to the next layer that performs a T-norm operation (product operation) [15] [16]. The output of this layer represents the firing strength of the antecedent part for each rule, which could be calculated according to the following:

$$U_i = \prod_{j=1}^{r} A_{ij}. \tag{2}$$

Where *r* is the number of input variables.

The firing strength is normalized in the third layer through dividing its value by the summation of all the firing strengths of all rules as:

$$\overline{U}_i = \frac{U_i}{\sum_{k=1}^{L} U_k}. \tag{3}$$

where L is the number of rules.

Finally, in the fourth layer, the summation of all the normalized values of U_i are multiplied by the corresponding weight c_i that represents the center of the membership function in the consequent-part of the rules [2] [4] [17] [18] to produce the center-of-gravity defuzzification operation. The output of the fourth layer represents the crisp output value for the given inputs, which can be obtained by the following formula:

$$y_s = \sum_{i=1}^{L} \overline{U_i} * c_i. \tag{4}$$

The integrating unit [12] is assumed to be a single layer neural network restricted to having as many output units as there are in neuro-fuzzy systems. Figure 3 depicts the integrating unit's architecture.

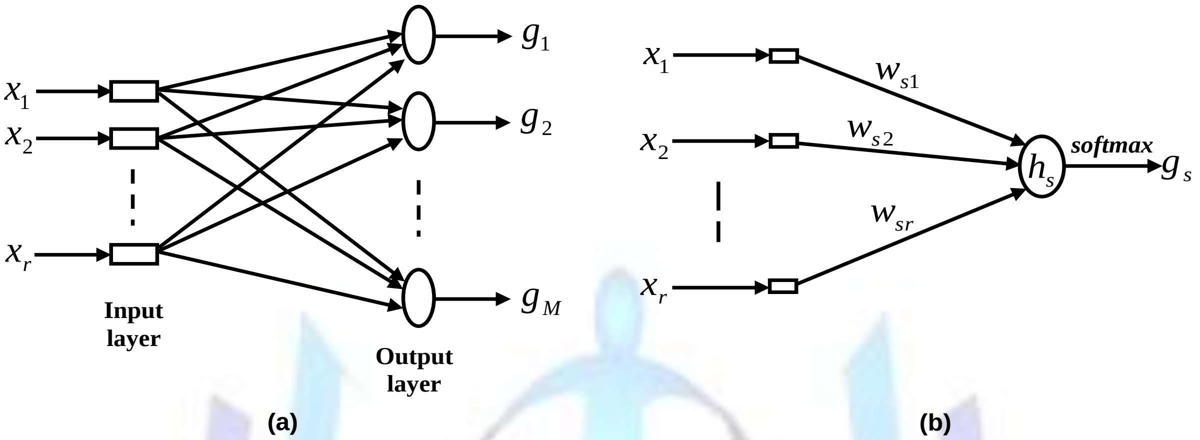


Figure 3. (a) A single layer of neurons constituting the integrating unit; (b) The S^{th} component of the integrating unit

In Figure 3(b), h_s is the weighted sum of input variables applied to the S^{th} output unit and is computed by:

$$h_s = \sum_{k=1}^{r} w_{sk}\ x_k. \tag{5}$$

where w_{sk} is the weight for the S^{th} component of the integrating unit and the k^{th} input variable, and the activation function g_s of the S^{th} output unit is related to h_s via a ***softmax*** transformation, which is described by:

$$g_s = \frac{\exp(h_s)}{\sum_{L=1}^{M} \exp(h_L)} \tag{6}$$

This transformation generalizes the maximum picking, or "***winner-takes-all***" operation in the sense that the output changes smoothly with variations in inputs. Finally, the total output for the whole system can be obtained by :

$$Y = \sum_{s=1}^{M} g_s\ y_s \tag{7}$$

1.3. Identification Algorithm of MNFS

The objective of the MNFS is to design a combined fuzzy system such that its output error is minimized in the Least-Mean-Square (*LMS*). Figure 4 shows the structure of the identification problem with the MNFS model [3] [13] [15] .

In MNFS, a different type of objective function is used with the formulas as:

$$E^p = \tfrac{1}{2}[\sum_{s=1}^{M} g_s^p (d^p - y_s^p)^2]. \tag{8}$$

and

$$E = \sum_{p=1}^{P} E^p. \tag{9}$$

where E is the total error for the whole system; E^p is the error in the pattern p; d^p is the desired output in the pattern p; y_s^p is the output for subsystem S in the pattern p; g_s^p is the output for component S, in the integrating unit, in the pattern p; P is the pattern number; M is the subsystem number.

ISSN 2278-5612

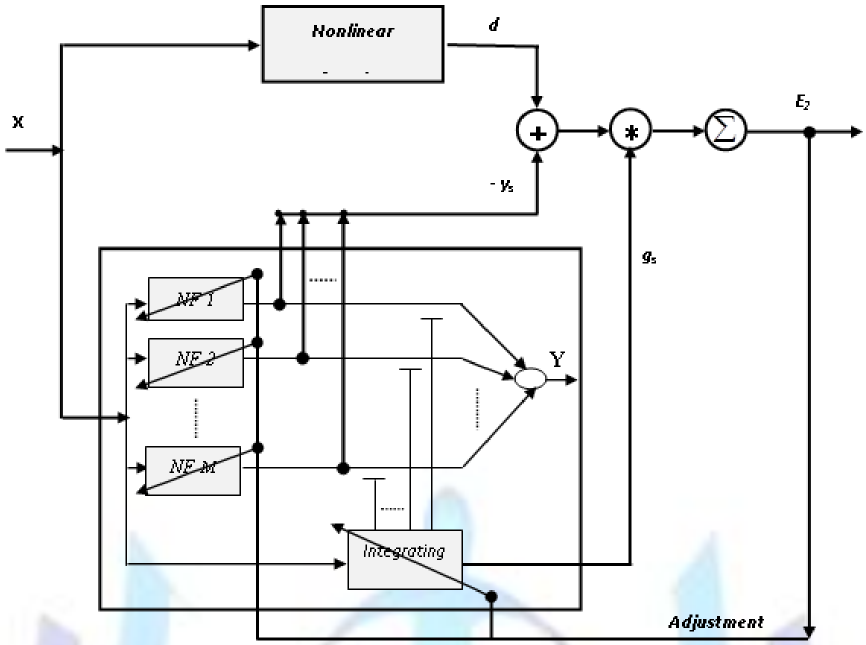


Figure 4. Identification problem with MNFS model

This function was first proposed by Jacobs *et al*. under the framework of competitive learning and modular networks [19]. The objective function E is defined with respect to all parameters in each subsystem, such as the centers a_{ij} and the widths b_{ij} for input membership functions, the centers c_i for output membership functions and the weights w_{sj} in the integrating unit.

$$a_{ij}(t+1)=a_{ij}(t)+k_a\left[g_s(t)*\left(d(t)-y_s(t)\right)*\left(c_i(t)-y_s(t)\right)*\frac{u_i}{\sum_{i=1}^{L}u_i}*\frac{\left(x_i-a_{ij}(t)\right)}{\left(b_{ij}(t)\right)^2}\right] \qquad (10)$$

$$b_{ij}(t+1)=b_{ij}(t)+k_b\left[g_s(t)*\left(d(t)-y_s(t)\right)*\left(c_i(t)-y_s(t)\right)*\frac{u_i}{\sum_{i=1}^{L}u_i}*\frac{\left(x_i-a_{ij}(t)\right)^2}{\left(b_{ij}(t)\right)^3}\right] \qquad (11)$$

$$c_i(t+1)=c_i(t)+k_c[g_s(t)*\left(d(t)-y_s(t)\right)*\frac{u_i}{\sum_{i=1}^{L}u_i}]. \qquad (12)$$

$$w_{sj}(t+1)=w_{sj}(t)-\tfrac{1}{2}k_w[\left(d(t)-y_s(t)\right)^2*g_s(t)*\left(1-g_s(t)\right)*x_j]. \qquad (13)$$

where i =1,…, L,{L is no. of rules in subsystem s}; j =1,…, r, {r is no. of input variables}; k_a, k_b, k_c, and k_w are the learning rates, and t means the learning iteration.

2. EXPERIMENTAL RESULTS

To test the performance of MNFS, we compare with the MFS apporach on several nonlinear functions. Two subsystems are used in both MFS and MNFS and each subsystem consists of three membership functions. The numbers of input-output data (training patterns) are chosen to be constant for these functions and equal to 100.

Experiment 1

$$D=\left\{\begin{array}{ll} 10-e^{0.5x} & x\leq 0 \\ 10-e^{-0.5x} & x>0 \end{array}\right\}$$

where $x \in$ [-5,5] is an input variable, and D is an output variable [12].

Table 1 shows the results of MNFS and MFS approaches for the D function. It can be seen that the MNFS gives a much better performance.

Table 1. Results of MNFS and MFS for D function

Approach	Error	Epochs
MNFS	0.0000419	300
MFS	0.0001375	500

Figure 5 illustrates the approximated outputs of MNFS and MFS approaches together with the desired *D* function.

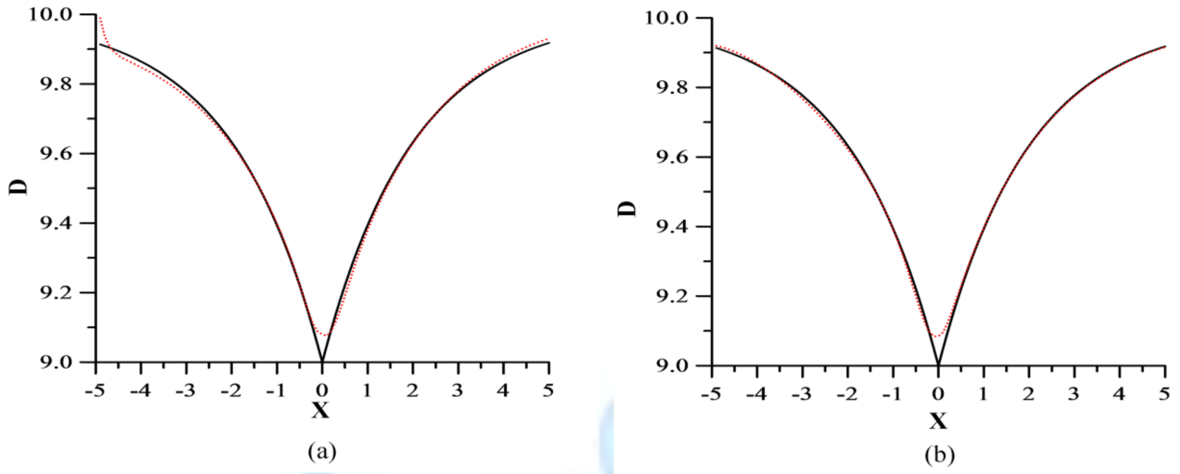


Figure 5. Outputs of desire *D* function (solid-line in blue) and the both models MFS & MNFS (dotted-line in red) for the *D* function. (a) Output of desired D function with MFS. (b) Output of desired D function with MNFS.

The Log (LMS) of error curves using the MNFS and MFS approaches against the learning-epochs as show in Figure 6. From Figures 5 and 6, it can be seen that MNFS has a better result compared with MFS for the *D* function.

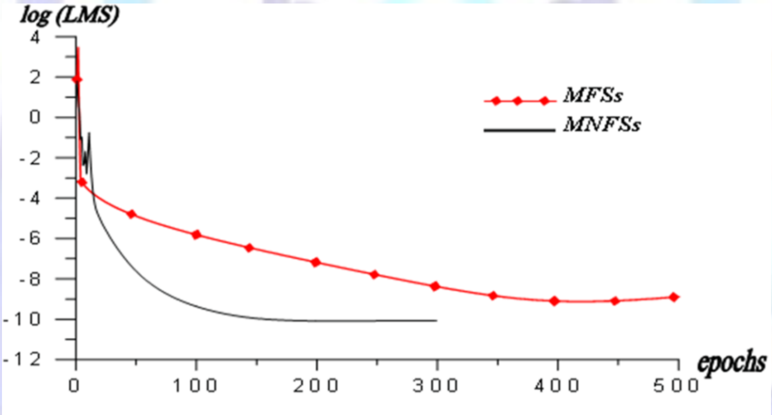


Figure 6. Log (LMS) curves for MNFS and MFS pproaches for *D* function.

In order to illustrate the individual behavior of two subsystems in the combined neuro-fuzzy system, Figure 7 depicts their separate outputs.

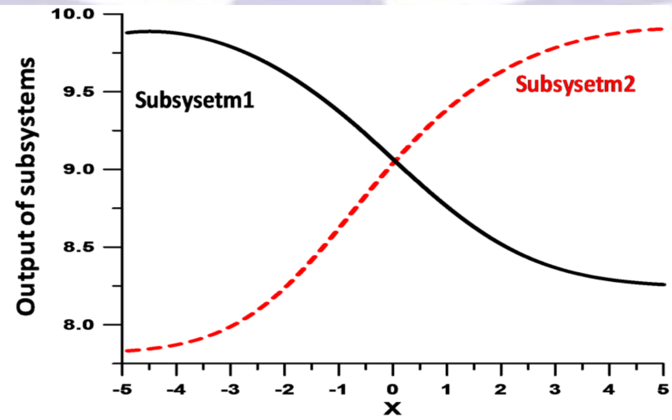


Fig. 7 Separate outputs of neuro-fuzzy subsystem.

These two subsystems compete with each other for the right to produce the desired approximation. As a result, ***subsystem 1*** is the winner in approximating the left piece of the function but the loser in approximating the right piece of the function, while ***subsystem 2*** just goes to the contrary. The competition has been coordinated by the integrating unit whose function is to provide relative weight values for the subsystems. Figure 8 shows the outputs of the integrating unit.

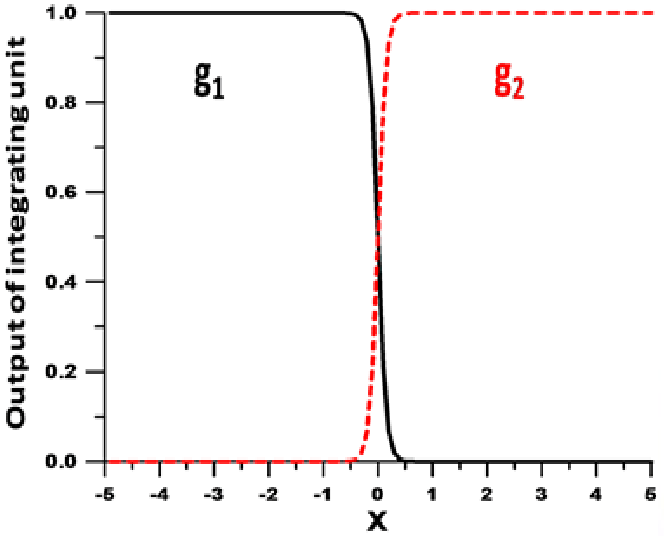


Fig. 8 Outputs of the integrating unit.

Clearly, in approximating the left piece of the function, the integrating unit adds a greater weight (nearly 1) to ***subsystem 1*** and a smaller weight (nearly 0) to ***subsystem 2***; while in approximating the right piece of the function, the integrating unit adds a greater weight (nearly 1) to ***subsystem 2*** and a smaller weight (nearly 0) to subsystem 1.

Experiment 2

$$Z_1 = \sin(x) * \cos(y)$$

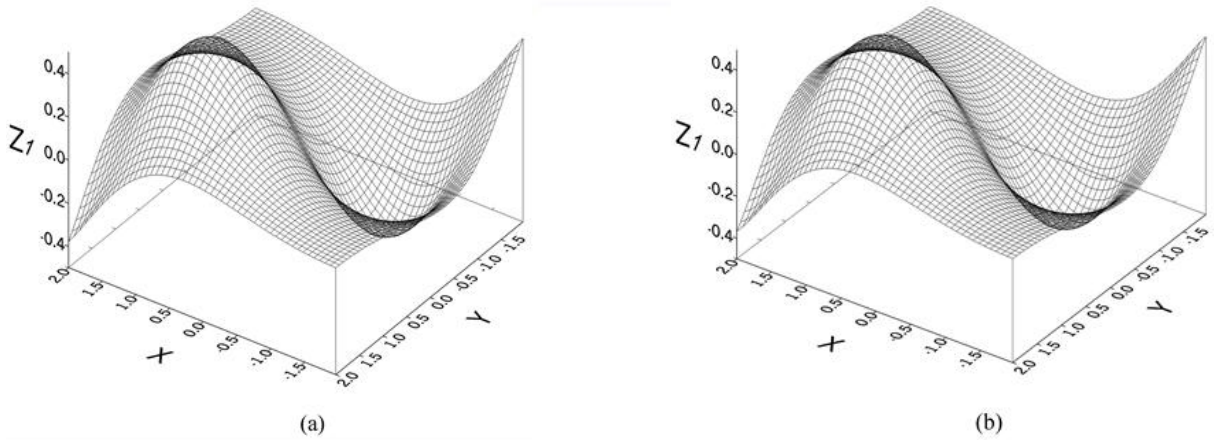
where $x,y \in [-2,2]$ is an input variables, and Z_1 is an output variable [5].

Table 2 shows results of MNFS and MFS approaches for Z_1 function. Again MNFS exhibits a much better performance than MFS.

Table 2. Results of MNFS and MFS for Z1 function

Approach	Error	Epochs
MNFS	0.00039	500
MFS	0.00519	500

Figure 9 presents a 3-D presentation of the desired and the approximated values for MNFS and MFS outputs for the Z_1 function.



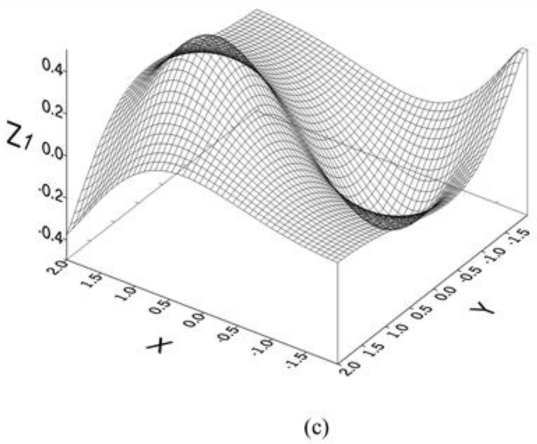


Figure 9. Desired and approximated (MNFS and MFS) outputs for the Z_1 function. (a) Desired Output. (b)MNFS output. (c) MFS output.

Similarly, Figures 10(a) and 10(b) show a graphic presentation of the error in the outputs for MNFS and MFS, respectively, for the Z_1 function. Figure 11 further shows the Log (LMS) of error curves using MNFS and MFS approaches against the learning-epochs.

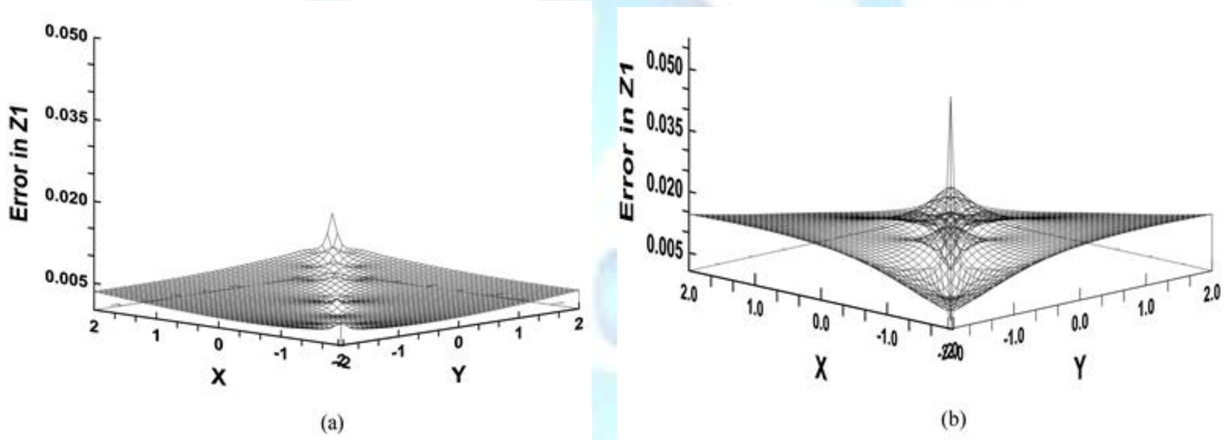


Figure 10. Error of MNFS and MFS approaches for the Z_1 function. (a) Error in MNFS. (b) Error in MFS.

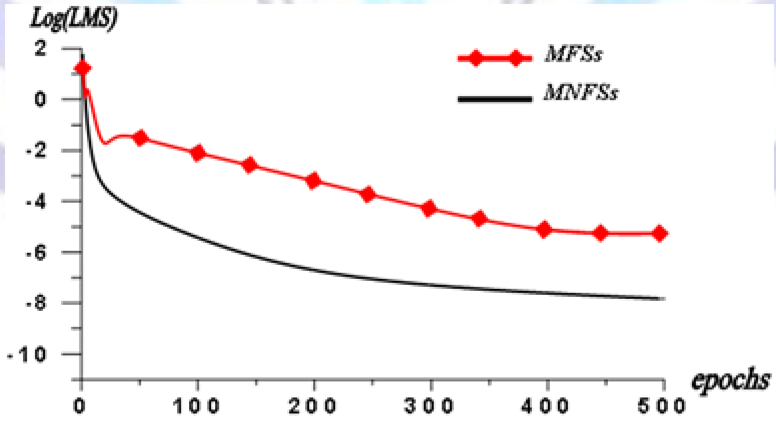


Figure 11. Log (LMS) curves for MNFS and MFS approaches for Z_1 function.

From Figures 9-11, it can be seen that the MNFS has a better result compared with the original MFS for the Z_1 function.

Note that this function has been investigated in [5] with an error equal to *0.0138* with *5000* training epochs.

Experiment 3

$$Z_2 = [4 \cdot \sin(\pi x) + 2 \cdot \cos(\pi y)] / 12 + 0.5$$

where $x,y \in [-1,1]$ is an input variables and Z_2 is an output variable [7].

Table 3 shows results of MNFS and MFS approaches for the Z_2 function.

Table 3. Results of MNFS and MFS for Z2 function

Approach	Error	Epochs
MNFS	0.00166	135
MFS	0.006	135

Figure 12 presents the desired and the approximated values for MNFS and MFS outputs for the Z_2 function.

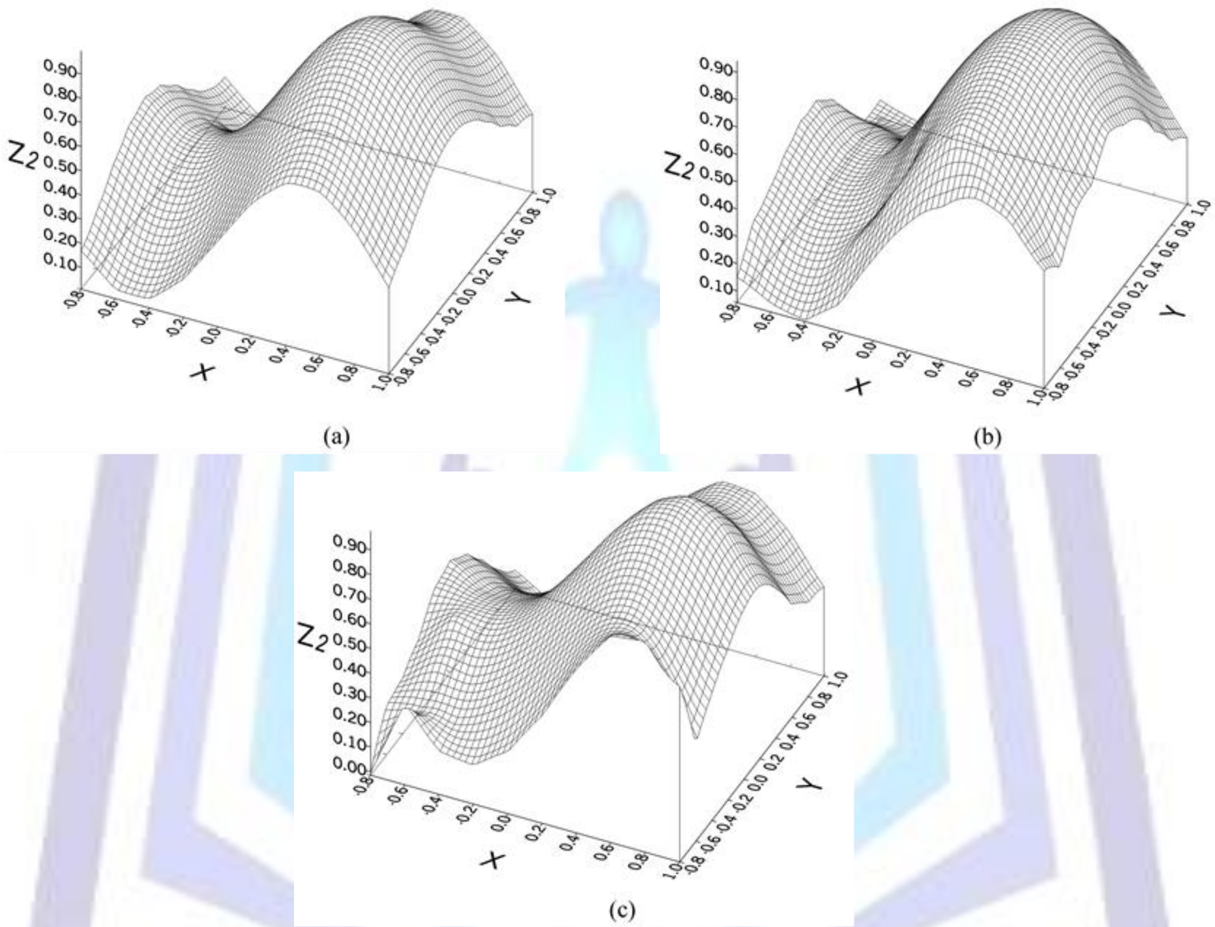


Figure 12. The desired and approximated outputs for z_2 function. (a) Desired Output. (b) MNFS output. (c) MFS output.

Similarly, Figures13(a) and 13(b) show the error in the outputs for MNFS and MFS, respectively, for the Z_2 function. Figure14 shows the Log (LMS) of error curves using the MNFS and MFS approaches against the learning-epochs.

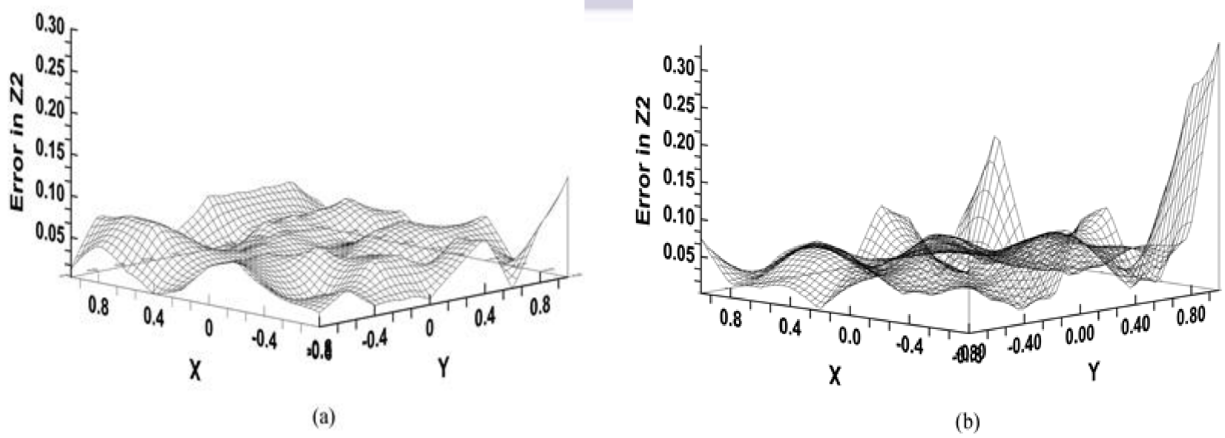


Figure 13. Error MNFS and MFS approaches for the z_2 function. (a) Error in the MNFS. (b) Error in MFS.

ISSN 2278-5612

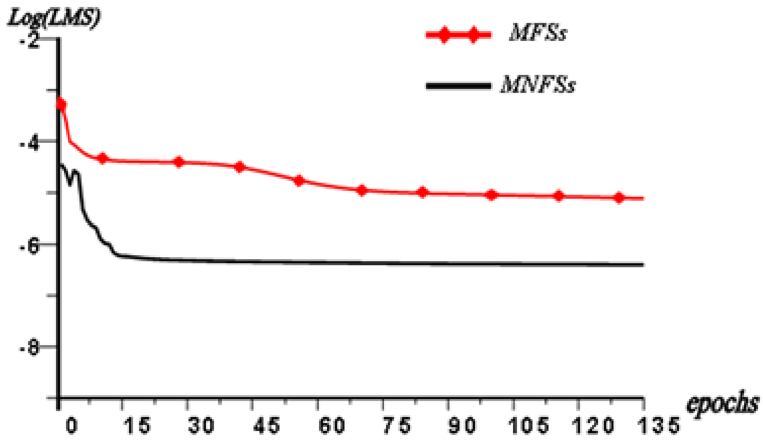


Figure 14. Log (LMS) curves for the MNFS and MFS approaches for the Z_2 function.

From Figures 12-14, it can be seen that the MNFS has a better result compared with the original MFS for the Z_2 function.

Note that this function has been considered in [7], with an error equal to *0.0613* with *135* training epochs.

3. CONCLUSIONS

The Multiple Fuzzy System (MFS) is one of the most successful approaches to reduce the number of fuzzy rules that constitute the underlying model. Such system has a high efficiency, providing good results in a quick manner with yet simple design. A new approach, Multiple Neuro-fuzzy System (MNFS), has been proposed to further improve the MFS efficiency, proved by several experiments on nonlinear sytems. From results obtained in this work, one can conclude that including all parameters in the learning process could increase the efficiency of the system, as well as the speed. Future investigations include the determination of the optimal number of subsytems, fuzzy rules in each subsystem, and the adaptive adjustment of membership functions.

REFERENCES

[1] Kamalasadan, S. and Ghandakly A. A.: Multiple Fuzzy Reference Model Adaptive Controller Design for Pitch-Rate Tracking. *IEEE Trans. Instrum. Meas.* 56(5), 1797-1808 , (Oct. 2007).

[2] Meher, S. K.; Ghosh, A.; Shankar, B. U.; and Bruzzone, L.: Neuro-Fuzzy Fusion: A New Approach to Multiple Classifier System. *IEEE ICIT* 209-212, (Dec. 2006).

[3] Babuška, R. and Verbruggen, H.: Neuro-Fuzzy methods for nonlinear system identification. *Annu. Rev. Control* 27(1), 73-85, (Jan. 2003).

[4] Otadi, M. and Mosleh, M.: Simulation and evaluation of dual fully fuzzy linear systems by fuzzy neural network. *Applied Mathematical Modelling* 35(10), 5026-5039, (Oct. 2011).

[5] Dickerson, J. A.: Learning Optimal Fuzzy Rules Using Simulated Annealing. *NAFIPS '97,* 102-105, (1997).

[6] Kolman, E. and Margaliot, M.: A New Approach to Knowledge-Based Design of Recurrent Neural Networks. *IEEE Trans. Neural Netw.* 19(8), 1389-1401, (Aug. 2008).

[7] Shi, Y. and Mizumoto, M.: A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules. *Fuzzy Sets and Systems* 112(1), 99-116, (May 2000).

[8] Yeh, C.-T.: Reduction to Least-Squares Estimates in Multiple Fuzzy Regression Analysis. *IEEE Trans. Fuzzy Syst.* 17(4) (Aug. 2009) 935-948.

[9] Chen, Z.; Aghakhani, S.; Man, J. and Dick, S.: ANCFIS: A Neurofuzzy Architecture Employing Complex Fuzzy Sets. *IEEE Trans. Fuzzy Syst.* 19(2), 305-322, (Apr. 2011).

[10] Massimo, P. and Giuseppe M.: Neurofuzzy Networks with Nonlinear Quantum Learning, *IEEE Trans. Fuzzy Syst.* 17(3), 698-710, (Jun. 2009).

[11] Cheu, E. Y.; Quek, H. C. and Ng, S. K.: TNFIS: Tree-Based Neural Fuzzy Inference System. *IEEE IJCNN* 398-405, (Jun. 2008).

[12] Yen, J.; Wang, L. and Langari, R.: Multiple Fuzzy Systems For Function Approximation. *NAFIPS'97,* 154-159, (Sep. 1997).

[13] Ciftcioglu, Ö.; Bittermann, M. S. and Sariyildiz, I. S.: A Neural Fuzzy System for Soft Computing. *NAFIPS'07*, 489-495, (Jun. 2007).

ISSN 2278-5612

[14] Chaturvedi, D. K.: Soft Computing Techniques and its Applications in Electrical Engineering. In: Studies in Computational Intelligence , 103, pp. 479-499, Springer, Heidelberg (2008).

[15] Lin, C.-J.; Chen, C. H. and Lin, C. T.: Efficient Self-Evolving Evolutionary Learning for Neurofuzzy Inference Systems. *IEEE Trans. Fuzzy Syst.* 16(6), 1476-1490, (Dec. 2008).

[16] Wu, C. F.; Lin, C. J. and Lee, C. Y.: A functional neural fuzzy network for classification applications. *Expert Systems with Applications* 38(5), 6202-6208, (May 2011).

[17] Aik, L. E. and Jayakumar, Y. S.: A Study of Neuro-fuzzy System in Approximation-based Problems. *MATEMATIKA* 24(2), 113-130, (2008).

[18] Chen, C. H.; Lin, C. J. and Lin, C. T.: A Recurrent Functional-Link-Based Neural Fuzzy System and Its Applications. *IEEE CIISP* 415-420, (Apr. 2007).

[19] Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J. and Hinton, G. E.: Adaptive Mixtures of Local Experts. *Neural Computation* 3(1), 79-87, (Feb. 1991).

[20] Khan, Rafiqul Zaman, and Haider Allamy. "Training Algorithms for Supervised Machine Learning: Comparative Study." International Journal of Management & Information Technology 4, no. 3 (2013): 354-360.

Biography Authors:

Dr. Imad S. Alshawi (M'12) was born in Basra, Iraq, in 1976. He received the B.Sc. and M.Sc. degrees in computer science from the College of Science, Basra University, Basra, Iraq, in 2001 and 2003, respectively. He received the Ph.D. degree in information and communication system from Southwest Jiaotong University, Chengdu, China. He is currently an Assistant Professor with Computer Science Dept., Basra University. He is a frequent referee for 5 journals. He is the author or co-author of more than 20 papers published in prestigious journals and conference proceedings. His current research interests include wireless sensor network and artificial intelligent. Dr. Alshawi is a member of the IEEE, the IEEE Cloud Computing Community and the IEEE Computer Society Technical Committee on Computer Communications.

Md. Haider Khalaf Jabbar Allamy: Haider allamy, is a research scholar in the Department of Computer Science; Aligarh Muslim University, Aligarh, India. He joined to his Ph.D course in 28-09-2012. His research interest includes Artificial Intelligence and Expert system. He did B.Sc Degree in computer science from University of Basrah, College of Science, Iraq, M.Sc in computer science from Jamia Hamdard University, Delhi, India. He is envoy by the university of Misan/ Iraq for complete his Ph.D course in Aligarh Muslim University, Aligarh, India.

Dr. Rafiqul Zaman Khan: Dr. Rafiqul Zaman Khan, is presently working as a Associate Professor in the Department of Computer Science at Aligarh Muslim University, Aligarh, India. He received his B.Sc Degree from M.J.P Rohilkhand University, Bareilly, M.Sc and M.C.A from A.M.U. and Ph.D (Computer Science) from Jamia Hamdard University. He has 18 years of Teaching Experience of various reputed International and National Universities viz King Fahad University of Petroleum & Minerals (KFUPM), K.S.A, Ittihad University, U.A.E, Pune University, Jamia Hamdard University and AMU, Aligarh. He worked as a Head of the Department of Computer Science at Poona College, University of Pune. He also worked as a Chairman of the Department of Computer Science, AMU, Aligarh. His Research Interest includes Parallel & Distributed Computing, Gesture Recognition, Expert Systems and Artificial Intelligence. Presently 04 students are doing PhD under his supervision. He has published about 35 research papers in International Journals/Conferences. Names of some Journals of repute in which recently his articles have been published are International Journal of Computer Applications (ISSN: 0975-8887), U.S.A, Journal of Computer and Information Science (ISSN: 1913-8989), Canada, International Journal of Human Computer Interaction (ISSN: 2180-1347), Malaysia, and Malaysian Journal of Computer Science(ISSN: 0127-9084), Malaysia. He is the Member of Advisory Board of International Journal of Emerging Technology and Advanced Engineering (IJETAE), Editorial Board of International Journal of Advances in Engineering & Technology (IJAET), International Journal of Computer Science Engineering and Technology (IJCSET), International Journal in Foundations of Computer Science & technology (IJFCST) and Journal of Information Technology, and Organizations (JITO).