Neural-Network-Based Fuzzy Identifier: Design and Evaluation

Dr. Kasim M. Al-Aubidy
Computer & Software Eng. Dept.
Philadelphia University, Jordan
e-mail: kasimal@yahoo.com

Mr. Salam A. Al-Ani
Computer & Control Eng. Dept.
University of Technology,
Iraq

Abstract:
This paper is concerned with the design and development of a simple fuzzy system for modeling of ill-defined dynamical systems. This system, which is represented as a feed forward neural network, is able to incorporate qualitative and quantitative information. Supervised linear back propagation learning algorithm has been applied to model a system through identifying the fuzzy parameters. This adaptive fuzzy system is implemented as an identifier of dynamical systems. The system performance has been evaluated for different simulated systems to demonstrate the application of the proposed system to identify the dynamics of linear and nonlinear time-invariant and time-variant systems.

1. Introduction:

Along with increased process complexity the abstraction and uncertainty are increased in the models, therefore their mathematical representation becomes very difficult. One significant approach in dealing with the major changes and uncertainty in dynamical processes is through artificial intelligence (AI), where one of the aims of AI is to replace human beings carrying out precise tasks by machine and hence the link between AI and control theory is strong. Intelligence in a system reefers to its ability to learn or adapt, and to modify its functional dependencies in response to new experiences or due to changes in the functional relationship [1,2].
Learning is an integral part of any intelligent system and exists at many levels of abstraction. At the guidance and servo levels, learning algorithms have been studied in the adaptive control field for many years, and these have been complemented by research into adaptive fuzzy and neural networks[3].
The proposed approach handles an architecture which is somehow similar to that employed by Takagi and Sugeno[4], and Kang[5] respectively, where they tried to introduce fuzzy rules from observation and adaptation. Unlike the methods used by them, the approach is based on neural network theory for training the fuzzy system in parameter tuning phase.
The rest of this paper is organized in four sections. The architecture of the adaptive fuzzy system and learning algorithm is presented in section 2. Section 3 outlines the resulting adaptive fuzzy system used as an identifier for dynamical systems. In section 4, the fuzzy identifier ability has been evaluated using several simulations. Finally, the conclusions are presented in section 5.

2. Adaptive fuzzy system:

The fundamental configuration of a fuzzy logic system is given in figure 1. It consists of four basic blocks; the fuzzy rule base, the fuzzy inference engine, the fuzzifier, and the defuzzifier.

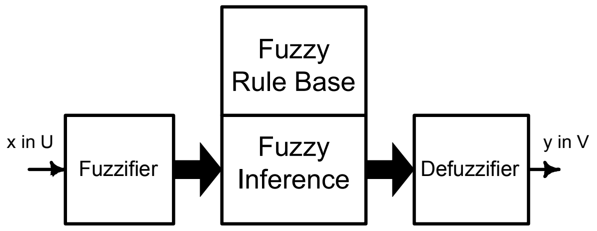

Figure 1. Basic Configuration of a Fuzzy System

Different interpretations for the fuzzy IF-THEN rules result in different mappings of the fuzzy inference engine, also there are different types of fuzzifier and defuzzifier. Several combinations of fuzzy inference engine, fuzzifier, and defuzzifier may constitute useful logic system. If the fuzzy logic system can be represented as a feed forward network, then the idea of back propagation training algorithm can be used to train them.

2.1. Adaptive fuzzy system structure:

The most useful class of fuzzifier is the center average [2,6], of the form;

$$f(x) = \frac{\sum_{j=1}^{M} y_j \left(\mu_{F_j}(y_j)\right)}{\sum_{j=1}^{M} \left(\mu_{F_j}(y_j)\right)} \tag{1}$$

where **M** is the number of fuzzy rules, $\mathbf{y_j}$ is the center of fuzzy set $\mathbf{F_j}$, which is a point in the universe of discourse **V** when $\mu_{Fj}(y)$ achieves its maximum value, and $\mu_{Fj}(y)$ is given by a product inference engine. Hence using product operator, equation 1 becomes;

$$f(x) = \frac{\sum_{j=1}^{M} y_j \left(\prod_{i=1}^{n} \mu_{F_i}(x_i)\right)}{\sum_{j=1}^{M} \left(\prod_{i=1}^{n} \mu_{F_i}(x_i)\right)} \tag{2}$$

where **n** is the number of input variables.

In order to develop training algorithms for this fuzzy logic system, the functional form of μ_{Fi} (xi) must be specified. The bell-shaped membership function, based on the normal distribution of the grades of the membership [7,8] is proposed, i.e. the membership function will be given by:

$$\mu_{F_i}(x_i) = \exp\left[-\left(\frac{x_i - m_i}{\sigma_i}\right)^2\right] \tag{3}$$

where $\mathbf{m_i}$ and $\boldsymbol{\sigma_i}$ are the center and width of the bell-shaped function of the **ith** input variable. Combining equations 2 and 3, the overall function of the fuzzy logic system is;

$$f(x) = \frac{\sum_{j=1}^{M} y_j \left(\prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - m_{ij}}{\sigma_{ij}}\right)^2\right]\right)}{\sum_{j=1}^{M} \left(\prod_{i=1}^{n} \exp\left[-\left(\frac{x_i - m_{ij}}{\sigma_{ij}}\right)^2\right]\right)} \tag{4}$$

This equation represents a fuzzy logic system with center average defuzzifier, product inference rule, non-singleton fuzzifier, and bell-shaped membership function. Equation 4 can be implemented on a Forward Neural Network (FNN). This connectionist model machines the approximate reasoning of fuzzy logic into a five layer neural network structure [9], as illustrated in figure 2.

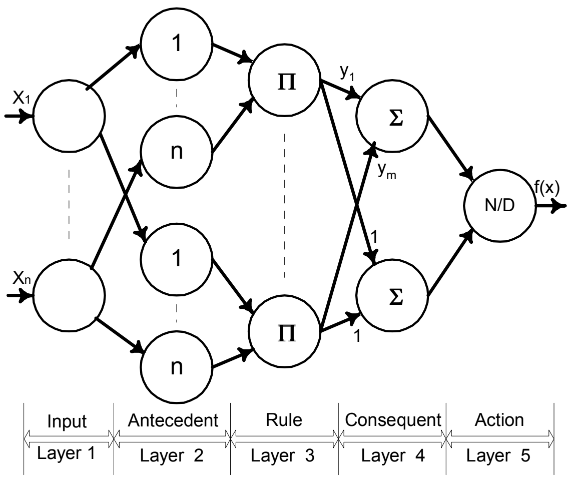

Figure 2. Adaptive Fuzzy System as an FNN.

Associated with each node in a typical neural network is an integration function, which serves to combine information or activation from the other nodes. This function X_i^L provides the net input of the **ith** node in layer **L**. A second action taken by each node is to output an activation value as a function of its net input;

$$O_i^L(k) = g\left(X_i^L(k)\right) \tag{5}$$

where **g(.)** denotes the activation function.

The basic function of the nodes in each layer would be defined as follows;

(1). Input layer:
The nodes in this layer transmit their inputs to layer 2;

$$X_1^1 = x_1, X_2^1 = x_2, \ldots, X_n^1 = x_n \tag{6}$$

$$O_i^1 = X_i^1 \tag{7}$$

where i=1,2,….,**n** and **n** is the number of the input linguistic variables.

(2) Antecedent layer:
The output from this layer is;

$$O_i^2 = \mu_{F_i}\left(X_i^2\right) \tag{8}$$

where X_i^2 is the input to node *i* in layer 2, and $\mathbf{F}_i$ is the linguistic label assigned to fuzzy set (small, large,..etc). Using equation 3, equation 8 can be rewritten to have:

$$O_i^2 = \exp\left[-\left(\frac{x_i^2 - m_{ij}}{\sigma_{ij}}\right)^2\right] \tag{9}$$

where $\mathbf{m}_{ij}$ and $\boldsymbol{\sigma}_{ij}$ are the center and width of the bell-shape function of the **i**th input of the **j**th rule.

(3). Rule layer:
The output from each node in this layer is dictated by the firing strength of the corresponding rule. With the proposed scheme (equation 4), the rule nodes perform the fuzzy product operation, therefore;

$$z_j = O_j^3 = \prod_{i=1}^{n} X_{ij}^3 \tag{10}$$

where X_{ij}^3 denotes the **i**th input to node **j** in layer 3.

(4). Consequent layer:
The upper node of this layer sums all outputs from the rule layer with action strength (y_j) and the lower node those with unity strength;

$$N = O_1^4 = \sum_{j=1}^{M} y_j X_j^4 \tag{11}$$

$$D = O_2^4 = \sum_{j=1}^{M} X_j^4 \tag{12}$$

where **N** and **D** are the numerator and denominator of equation 4.

(5). Action layer:
The network output would be pumped out the single node layer;

$$f(x) = O^5 = \frac{X_1^5}{X_2^5} = \frac{N}{D} \tag{13}$$

2.2. Adaptive fuzzy training algorithm:

Based on the error back propagation algorithm, the goal is to determine a fuzzy logic system f(x), in the form of equation 4, which minimizes the error function;

$$E(k) = 0.5\sum_{j=1}^{P}\left[f_j(x(k)) - d_j(k)\right]^2 \tag{14}$$

where **P** is the number of outputs and **dj(k)** is the **j**th desired output at time **k**.
Without loss of generality, multi-input single-output (MISO) fuzzy logic system is considered in this paper. A multi-output system can be always decomposed into a group of single-output systems [10], therefore for P=1, equation 14 is reduced to:

$$E(k) = 0.5(f(x(k)) - d(k))^2 \tag{15}$$

According to equation 4, if the number of rules is **M**, then the problem becomes training the parameters y_j, m_{ij}, and σ_{ij} such that E(k) is minimized.
Based on the back propagation training algorithm the iterative equations for training the parameters y_j, m_{ij}, and σ_{ij} are;

$$y_j(k+1) = y_j(k) - \eta(f(x(k)) - d(k))\frac{z_j}{D} \tag{16}$$

$$m_{ij}(k+1) = m_{ij}(k) - 2\eta\frac{z_j}{D}(f(x(k)) - d(k))\cdot (y_j(k) - f(x(k)))\cdot\left(\frac{X_i^2(k) - m_{ij}}{(\sigma_{ij})^2}\right) \tag{17}$$

$$\sigma_{ij}(k+1) = \sigma_{ij}(k) - 2\eta\frac{z_j}{D}(f(x(k)) - d(k))\cdot (y_j(k) - f(x(k)))\cdot\left(\frac{(X_i^2(k) - m_{ij})^2}{(\sigma_{ij})^3}\right) \tag{18}$$

where η is the learning rate. Equations 16, 17 and 18 perform an error back propagation procedure.

3. Fuzzy identifier:

A fuzzy logic system, given in equation 4, can be considered as a universal approximator with a back propagation training algorithm. This system can be used as a system identifier.

3.1. Fuzzy identifier structure:

The series-parallel model enjoys several advantages over the parallel model in the case that the plant is stable in the bounded-input-bounded-output sense, therefore, the series-parallel model is used in the simulation of fuzzy identifier. Sometimes the unknown function is a combination of a known linear part and an unknown linear part. In this case, the unknown nonlinear part is to be modeled by the fuzzy logic system.

3.2. Parameters selection:

The fuzzy identifier has good parameter choosing method, some parameters of the fuzzy identifier which are; the learning rate η, and the number of rules **M,** would be selected according to the application. So, what is good for one situation is not necessary suitable for another. This is due to the adaptive fuzzy system, described in section 2, that has no rules to govern the relationships between these two parameters. Therefore, experimentation may be required to achieve good results.

3.3. Fuzzy identifier adaptation:

On-line adaptation is performed by supervised learning algorithm given in equations 16,17, and 18 to obtain the optimum values of the parameters y_j, m_{ij}, and σ_{ij} respectively. This ensures that, for every input, the fuzzy system output is sufficiently closed to the desired output [7].

4. Simulation results:

In simulation of all time-invariant plants the following were considered;

- A series-parallel identifier was used, where the order of the plant must be predetermined.
- The unknown function represented in general form considered in equation 4 with **M** chosen by trial and error. All the nonlinear functions were presented in difference equations.
- Separated functions are used to represent systems with combined functions.
- To evaluate the identifier performance with a time-invariant system, a test phase is imposed. The output of the network is compared with that of the plant on a test signal of 250 samples. The test signal consists of mixture of sinusoids and constant inputs;

$$u_{test}[k] = \begin{cases} \sin(\pi k/5) & k \le 62 \\ 1 & 63 \le k \le 124 \\ -1 & 125 \le k \le 187 \\ 0.3\sin(\pi k/12) + 0.1\sin(\pi k/16) + 0.6\sin(\pi k/5) & 188 \le k \le 250 \end{cases} \quad \ldots(19)$$

In most plants, the parameters $y_j(0)$, $m_{ij}(0)$, and $\delta_{ij}(0)$ are initialized randomly over [-2,2], [-1,1], and [0.01,3] respectively. The test signal was applied after 5000 iterations of random training signal.

4.1 Nonlinear systems:

Plant 1:

$$y_p[k+1] = \frac{y_p[k]}{1 + y_p^2[k-1]} + u[k] \qquad (20)$$

To identify this plant [11], a series-parallel model described by:

$$y_m[k+1] = f(y_p[k], y_p[k-1]) + u[k] \qquad (21)$$

The resultant mean squared error was 0.55×10^{-3}. Figure 3 shows the output of the plant and that of the network model for the given test signal.

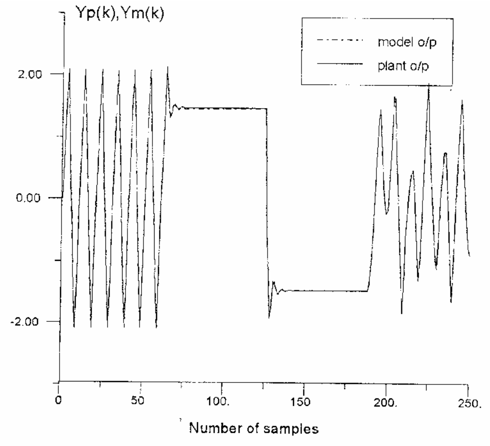

Figure 3. Output of plant 1 and its model.

Plant 2:

Consider this plant [10];

$$y_p[k+1] = 0.6\sin(\pi u[k]) + 0.3\sin(3\pi u[k]) + 0.1\sin(5\pi u[k]) + 0.3y_p[k] + 0.6y_p[k-1] \quad \ldots (22)$$

The model will be:

$$y_m[k+1] = f(u[k]) + 0.3y_p[k] + 0.6y_p[k-1] \qquad (23)$$

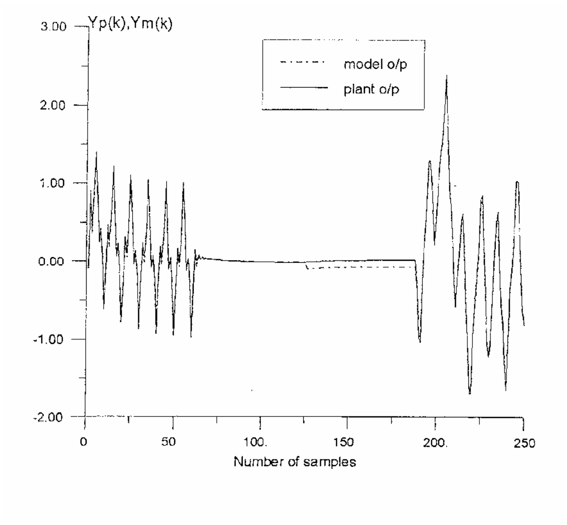

Figure 4. Output of plant 2 and its model.

Figure 4 shows the outputs of the plant and the model for the given test signal. The parameters $y_j(0)$, $m_{ij}(0)$, and $\delta_{ij}(0)$ are initialized randomly over [-5,5], [-1,1], and [0.001,1] respectively. After 5000 iterations the resultant mean squared absolute error was 0.3×10^{-3}.

Plant 3: [8]

$$y_p[k+1] = \frac{1.5y_p[k]\cdot y_p[k-1]}{1+y_p^2[k]+y_p^2[k-1]} + 0.1\sin(y_p[k]+y_p[k-1])+1.2u[k] \quad ..(24)$$

The model will be:

$$y_m[k+1] = f_1(y_p[k], y_p[k-1]) + f_2(y_p[k], y_p[k-1]) + 1.2u[k] \quad (25)$$

Figure 5 illustrates the output of the plant and the trained model when applying the test signal. The resultant mean squared error was 0.1×10^{-3}.

Plant 4: [12]

$$y_p[k+1] = \frac{y_p[k]}{1+y_p^2[k]} + (u[k]+1).(u[k]-1) \quad (26)$$

The model will be:

$$y_m[k+1] = f_1(y_p[k]) + f_2(u[k]) \quad (27)$$

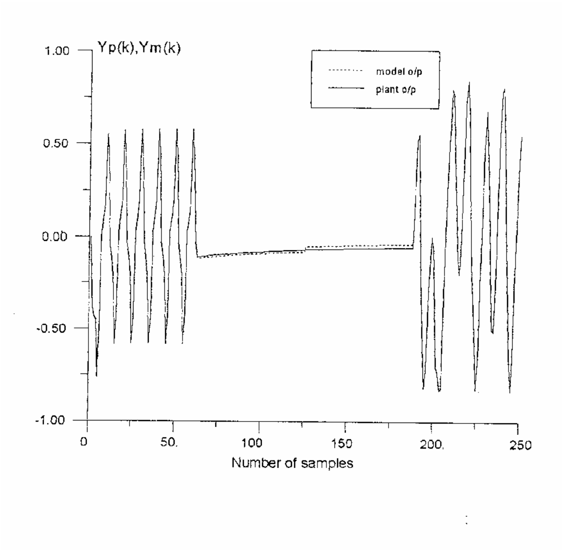

Figure 5. Output of plant 3 and its model.

In this test, the resultant mean squared error was 0.1×10^{-3}. Figure 6 shows the output of the plant and that of the network model for the given test signal.

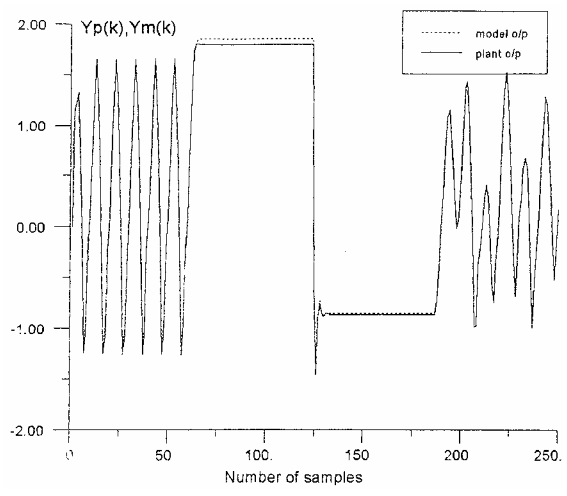

Figure 6. Output of plant 4 and its model.

4.2. Time-varying systems:

In this section, the case of the time-varying system is considered. The fuzzy identifier can be trained to overcome variation in system parameters such that the error between the plant and the model outputs is minimized. Let the plant described by equation 28 with time varying element **a**[k] is;

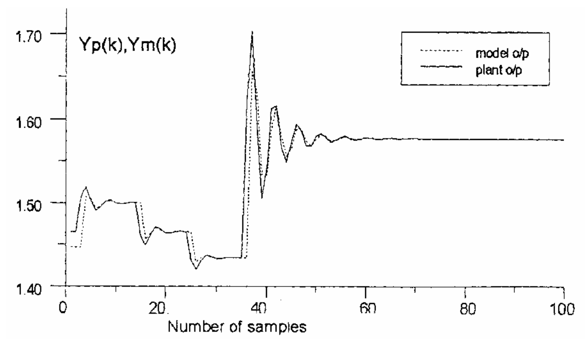

Figure 7. Output of the time-variant system.

$$y_p[k+1] = \frac{y_p[k]}{a[k] + y_p^2[k-1]} + \mu[k] \tag{28}$$

where the parameter **a**[k] is given by:

$$a[k] = \begin{bmatrix} 0.75 & 1 \le k \le 14 \\ 1 & 15 \le k \le 24 \\ 1.25 & 25 \le k \le 35 \\ 0.25 & k \ge 36 \end{bmatrix} \tag{29}$$

The model is;

$$y_m[k+1] = f(y_p[k], y_p[k-1]) + u[k] \tag{30}$$

From figure 7 it can be pointed that the fuzzy identifier adapt the variation in many parameters.

4.3. MIMO systems:

Consider the following system [10,13];

$$\begin{bmatrix} y_{p1}[k+1] \\ y_{p2}[k+1] \end{bmatrix} = \begin{bmatrix} \dfrac{y_{p1}[k]}{1+y_{p2}^2[k-1]} \\ \dfrac{y_{p1}[k] \cdot y_{p2}[k]}{1+y_{p2}^2[k-1]} \end{bmatrix} + \begin{bmatrix} \mu_1[k] \\ \mu_2[k] \end{bmatrix} \tag{31}$$

and the model will be;

$$\begin{bmatrix} y_{m1}[k+1] \\ y_{m2}[k+1] \end{bmatrix} = \begin{bmatrix} f_1(y_{p1}[k], y_{p2}[k]) \\ f_2(y_{p1}[k], y_{p2}[k]) \end{bmatrix} + \begin{bmatrix} \mu_1[k] \\ \mu_2[k] \end{bmatrix} \tag{32}$$

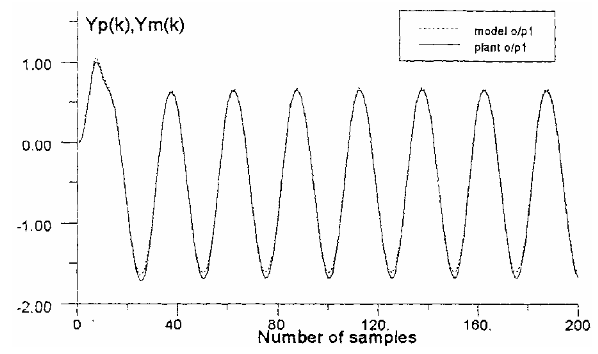

Figure 8. Output of the MIMO plant and its model.

Both f_1 and f_2 in the form of equation 4 with M=40 and η=0.5. The identification procedure was carried out for 5000 time steps using random inputs $\mu_1[k]$ and $\mu_2[k]$, whose magnitudes were uniformly distributed over [-1,1]. Figure 8 shows the resultant outputs of the plant and the identification model, where the two inputs ($\mu_1[k]$ and $\mu_2[k]$) are sinusoidal.

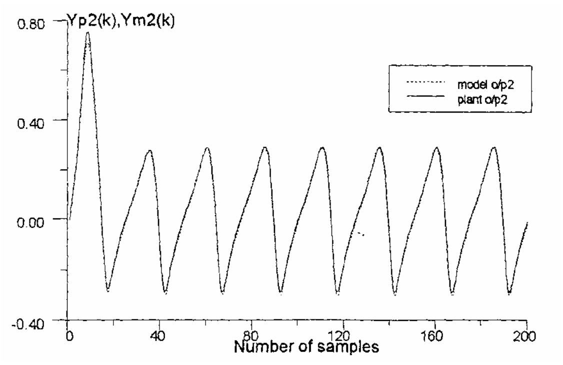

Figure 9. Output of the MIMO plant and its model.

5. Conclusions:

By simple comparison between a multi-layer perceptron (with preceding back propagation training algorithm) and the adaptive fuzzy identifier, the following points can be pointed out:

(a). They are similar in;

- their basic operation; forward computation and backward training,
- using iterative gradient algorithms to minimize the mean square error between the actual and desired outputs,
- both are universal approximators and qualified to solve nonlinear problems.

(b). They are different in;

- the adaptive fuzzy system parameters have clear physical meaning, therefore, a suitable initial parameters choosing method can be developed,
- linguistic information can be incorporated into adaptive fuzzy system.

The results obtained from the simulation demonstrate that;

- convergence of back propagation training algorithm for the fuzzy system is much faster than that of back propagation training algorithm for the neural network.
- the fuzzy identifier can achieve similar or better performance than that of the neural identifier.
- after incorporating some linguistic information, the fuzzy identifier converges faster to the real system.

References

[1] J. Yen and R. Langari, "Fuzzy Logic: Intelligence, Control, and Information, Prentice-Hall, USA 1999.

[2] M. Brown and C. Harris, "Neurofuzzy Adaptive Modelling and Control", Prentice-Hall, UK, 1994.

[3] T. J. Ross, "Fuzzy Logic with Engineering Applications", McGraw-Hill, USA, 1995.

[4] T. J. Takagi and M. Sugeno, "Fuzzy Identification of Systems and It's Applications to Modelling and Control", IEEE Trans on Systems, Man and Cybernetics, Vol.SMC-15, No.1, January/February 1985, pp.116-132.

[5] H. Kang and G. Vachtsevanos, "Adaptive Fuzzy Logic Control: Explicit Adaptive Control with Lyapunov Stability", Automatic Control Conference, USA, 1992, pp.2279-2283.

[6] S. G. Naoam, "Design of Fuzzy Logic Controller of a Distillation Column", M.Sc. Thesis, Control and Computer Eng. Dept, University of Technology, Iraq, September 1995.

[7] C. T. Lin and C. S. G. Lee, "Neural-Network-Based Fuzzy Logic Control and Decision System", IEEE Trans on Computers, Vol.40, No.12, December 1991, pp.1320-1336.

[8] R. K. Al-Sibakhi, "Fuzzy Neural Control of Dynamical Systems", M.Sc. Thesis, Electrical Eng. Dept, University of Al-Mustonsiriya, Iraq, October 1996.

[9] Y. M. Chen and K. F. Gill, "Application of Fuzzy Neural Network to Control of an Unstable System", IMACS Intr. Symposium on Signal Processing, Robotics and Neural Networks, France, April 1994, pp.454-457.

[10] L. X. Wang, "Adaptive Fuzzy Systems and Control: Design and Stability Analysis", Prentice-Hall, USA, 1994.

[11] P. S. Sastry, G. Santharam, and K. P. Unnikrishnan, "Memory Neural Networks for Identification and Control of Dynamical Systems", IEEE Trans on Neural Networks, Vol.5, No.2, March 1994, pp.306-319.

[12] K. S. Narendra and K. Prathasarathy, "Gradient Methods for the Optimization of Dynamical Systems Containing Neural Network", IEEE Trans on Neural Networks, Vol.2, No.2, March 1991, pp.224-231.

[13] K. S. Narendra and K. Prathasarathy, "Identification and Control of Dynamical Systems Using Neural Networks", IEEE Trans on Neural Networks, Vol.1, No.1, January 1990, pp.4-27.