# Identification of Nonlinear Systems Based on a Genetically Trained Fuzzy Neural Network

**Omar F. Al-Karkhy*, Intisar A.M. Al-Said* and Ahmed I. Al-Dulaimy***

Received on: 1/3/2004
Accepted on: 15/8/2004

**Abstract**

This paper presents an intelligent modeling technique that combines the merits of Fuzzy Logic (FL), Neural Networks (NNs) and Genetic Algorithms (GAs), where the GA is used to train a Fuzzy Neural Identifier (FNI) to identify ill-defined dynamical systems using the series-parallel identification model.
The parameters of the FNI (including the input and output scaling factors, the centers and widths of the membership functions (MFs) for the input variables, and the quantization levels of the output variable, that are subjected to constraints on their values by the expert) are modified by the real-coding GA with hybrid selection method and elitism strategy based on minimizing the Mean Square of Error (MSE) criterion. The simulation results for modeling three different nonlinear plants show the effectiveness of this FNI.

الخلاصة

يقدم هذا البحث تقنية نمذجة ذكية تجمع فوائد المنطق المضبب والشبكات العصبية والخوارزميات الجينية، إذ تستخدم الخوارزمية الجينية لتعليم معرّف ضبابي عصبي لتعريف أنظمة ديناميكية قليلة التعريف باستخدام نموذج التعريف المتسلسل— المتوازي.
معاملات المعرّف الضبابي العصبي (بضمنها عوامل التقييس للإدخال والإخراج، والمراكز والامتدادات لدوال العضوية لمتغيرات الإدخال، ومستويات الكميات لمتغير الإخراج والتي تخضع لقيود على قيمها من قبل الخبير) يتم تعديلها من قبل الخوارزمية الجينية ذات معاملات الترميز بالقيم الحقيقية وطريقة اختيار مهجنة وحكم النخبة بناءً على تقليل معيار (معدل مربع الخطأ).
وقد أظهرت نتائج التمثيل الناتجة عن تعريف ثلاثة أنظمة لا خطية مختلفة فعالية هذا المعرّف الضبابي العصبي.

*Control and Systems Engineering Department, University of Technology, Baghdad, Iraq.

## 1. Introduction

In the fields of physical science and engineering, the development of a mathematical model of some physical phenomenon is of a significant importance in order to make analytical predictions about the behavior of the system. In control applications, we are often interested in modeling a physical plant, so as to predict the effect of control efforts and disturbances on the plant [1]. Increasingly, control systems are required to have high dynamical performance and robust behaviors, yet are expected to cope with more complex, uncertain and highly nonlinear dynamic processes. Along with this increased process complexity is increased abstraction and uncertainty in the models and their mathematical representation. One significant approach in dealing with major changes and uncertainty in nonlinear dynamical processes is through intelligent modeling and control [2].

Currently, there are a number of techniques that can be used as a basis for the development of intelligent systems, namely, Expert Systems (ES), Fuzzy Logic (FL), Neural Networks (NNs), Genetic Algorithms (GAs) and Artificial Life (AL) [3]. However, every intelligent technique mentioned above has its particular drawback, which limits its use to particular applications and not for others. For example, while NNs are good at recognizing patterns, they are not good at explaining how they reach their decisions. On the other hand, FL systems, which can reason with imprecise information, are good at explaining their decisions, but they can't automatically acquire the rules they use to make those decisions. These limitations, and others, have been a central driving force behind the appearance of a rapidly emerging field of Fuzzy Neural Networks (FNNs), which attempt to obtain the advantages of both FL and NNs techniques while avoiding their individual drawbacks [4].

The choice of learning method for the NNs is of big importance. Supervised learning, for example, is the most common method for learning NNs, which depends on availability of sufficient data. However, there are some situations where such data can't be obtained. In addition, there is another problem with the supervised learning algorithms, which is the tendency to get stuck at local optima in weight space.

As an alternative learning algorithm for the NNs, the GA may be used to overcome the two problems just mentioned above. Because in GAs, the need for the availability of supervised data does not appear. In addition, the best regions, as defined by the fitness evaluation function, of the search space gather increasing numbers of vectors. Hence, all regions of the space continue to receive attention. Moreover, the crossover and mutation operators work to ensure that all regions of the search space continue to be explored. Therefore, the GA is more likely to finally converge toward finding the global solution of a given problem [5][6][7].

This emerging field of integrating the merits of FL, NNs, and GAs has been termed Soft Computing (SC), representing a flexible and more powerful approach that takes advantage of the three methodologies [8].

Utilizing this powerful approach, this paper presents a genetically trained Fuzzy Neural Identifier (FNI) to identify ill-defined dynamical systems.

## 2. Structure of the Fuzzy Neural Identifier (FNI)

The structure used by Ad'doory [9] as

a controller and by Ismaeel [10] as an emulator is used in the present work but with the genetic learning instead of the Back Propagation Algorithm (BPA) used in the mentioned references.

The structure of this FNI has a total of six layers as can be seen in Fig. (1). For convenience, the following notation will be used to describe the function of each layer

$I_i^L$ : the net input value to the i[th] node in layer L,

$O_i^L$ : the output value of the i[th] node in layer L.

The function of each layer can be summarized as follows:

*Layer One (Input Layer)*

There are three input (linguistic) nodes in this layer, which transmit their inputs to layer two after they have been scaled into a predetermined range.

*Layer Two*

The three fuzzy sets in this layer are consisting of $n_1$, $n_2$, and $n_3$ linguistic terms, respectively. Hence, $n_1+n_2+n_3$ nodes are included in this layer (in this work $n_1=n_2=n_3=7$). The input of each node in this layer can be expressed as:

$$I_i^2 = \begin{cases} c_1 x_1 - b_{i1} & i = 1,..,7 \\ c_{11} x_2 - b_{i2} & i = 8,...,14 \\ c_{111} x_3 - b_{i3} & i = 15,..,21 \end{cases} \quad (1)$$

Where $x_1$, $x_2$ and $x_3$ represent the states of the system to be identified; $c_1$, $c_{11}$ and $c_{111}$ are the input scaling factors representing the weights of the links between the input layer and layer two. $b_{i1}$, $b_{i2}$ and $b_{i3}$ are the biases to the three fuzzy sets in layer two respectively, and they determine the positions of the "centers" of the input membership functions (MFs). The activation functions of all nodes in layer two are linear and thus their outputs $O_i^2$ are:

$$O_i^2 = I_i^2 \qquad i = 1,2,..,21 \quad (2)$$

*Layer Three*

It has the same number of nodes as layer two. The nodes in the two layers are linked one-to-one and the variable weights of the connections between them $t_{ij}$ define the "widths" of the input MFs for each fuzzy set. All nodes in this layer have a bell-shaped activation functions and their outputs $O_i^3$ are expressed as:

$$O_i^3 = \exp(-1/2.[\frac{O_i^2}{t_{ij}}]^2) \quad (3)$$

Where i=1,2…21, and j=1,2,3. Layer three together with layer two can be considered as the antecedent part of this FNI.

*Layer Four (Rule Layer)*

This layer, which has seven rule nodes, performs the multiplication operation for fuzzy inference. The links between layers three and four have unit weights. The input to each node in this layer is:

$$I_i^4 = O_i^3.O_{i+7}^3.O_{i+14}^3 \quad (4)$$

Where i=1,2…7. While the corresponding output is:

$$O_i^4 = I_i^4 \quad (5)$$

*Layer Five (Consequent Layer)*

It has only two nodes, which perform the center of gravity defuzzification method. The total inputs $I_1^5$ and $I_2^5$ to both nodes are given by:

$$I_1^5 or\ I_2^5 = \sum_{i=1}^{7} y_{ji} O_i^4 \quad (6)$$

where j=1,2. For the first node $y_{1i}$ defines the value of the i[th] output quantization level and is obtained through training. For the second node all $y_{2i}$s' are fixed at unity. The outputs of these units are:

$$O_1^5 = I_1^5 \quad (7)$$

$$O_2^5 = 1 / I_2^5 \quad (8)$$

*Layer Six (Action Layer)*

It has only one node, which receives the outputs of layer five ($O_1^5$ and $O_2^5$) via links with unit weights and multiplies them together to yield its net input $I_6$:

$$I_6 = O_1^5 O_2^5 \quad (9)$$

And finally, to obtain the output $y_m$ of the FNI, $I_6$ is multiplied with a factor $c_2$ representing the output scaling factor of the FNI:

$$y_m = O_6 = c_2 I_6 \quad (10)$$

## 3. Operators of the real-coded GA

The following operators were used in the real-coded GA used to train the FNI:

### 3.1 Hybrid selection

This selection method, which was first introduced by Al-Said [11], is a combination of Roulette Wheel and deterministic selection, forming a robust strategy inspired from the simplex selection method. In this method, it is to accept in the new population only those strings that have better fitness values than the worst individual in the old population. This method is expected to ensure good guidance in the complex and nonlinear search space, due to its ability to improve the strings in a given generation from those in the previous one.

### 3.2 Elitism

In this operation, the best n parents (in this work the best two) from the current generation are copied directly into the next generation as they are. This approach prevents the best fitness value in a given generation from becoming worse than that in the previous generation [12].

### 3.3 Crossover

In the real-coding crossover operator, which is similar to that of binary coding, a pair of mating chromosomes exchanges information by exchanging a subset of their components, where an integer position k is selected uniformly at random along the chromosome length. Then two new chromosomes are created by swapping all the genes between positions k+1 and L, where L is the chromosome length [12].

For example, the pair of chromosomes a and b as

a=[2.3,5,1.9,7,4,3.2,8.5]

b=[7.6,9,3,2.9,6,5,1]

are crossed over at the third digit position to yield:

a'=[2.3,5,3,2.9,6,5,1]

b'=[7.6,9,1.9,7,4,3.2,8.5]

### 3.4 Mutation

This operation causes random changes in the components of the chromosomes in the new population. In binary-coding GA, this operator randomly flips some of the bits in chromosomes. For example, the chromosome 00010 might be mutated in its second position to yield 01010. In real-coded GA this operator is adapted by simply replacing the mutated 'gene' with another random number chosen in the same range assigned for that 'gene'. As an example, the chromosome c=[5,8.1,1.6,4,2,9,3] is mutated at the fifth 'gene' to yield:

c'=[5,8.1,1.6,4,3.3,9,3].

## 4. The proposed genetic learning for the FNI

The following genetic procedure is used for training the FNI:

**Step1:** Initialize the genetic operators: the crossover probability Pc, the mutation probability Pm, the population size, and the maximum number of generations.
**Step2:** Generate randomly the initial population within certain bounds, in which each individual represents the entire modifiable weight connections of a single FNI.
**Step3:** Evaluate the objective function for each individual in the population using the Mean Square of Error (MSE) criterion having the form:

$$MSE = \sum_{k=1}^{Np} \frac{[y_p(k) - y_m(k)]^2}{N_p} \qquad (11)$$

where $y_p(k)$ is the plant's output at sample k, $y_m(k)$ is the FNI's output at sample k and $N_p$ is the number of the training patterns. Then, for each individual, calculate the fitness function using the Darwinian fitness equation of the form [11]:

$$fitness = \frac{1}{\varepsilon + objective \quad function} \qquad (12)$$

where $\varepsilon$ is a small constant chosen to avoid division by zero.
**Step4:** Put in descending order all the chromosomes in the current population (i.e. the first one is the fittest). Then apply "Elitism" strategy described in section 3.2.
**Step5:** Select individuals by using the hybrid selection method [11], and then apply the real-coded genetic operators of crossover and mutation described previously.
**Step6:** Stop if the maximum number of generations is reached, otherwise increment the generations counter by one and go to step3.

## 5. Simulation results

When the series-parallel identification model and the parallel identification model are employed for identification, the system is assumed to be bounded-input bounded-output (BIBO) stable in the presence of an input. However, if the parallel structure is employed, it cannot be guaranteed that the learning process of the weights will converge or the error between the output of the system and that of the network will tend to zero [10,13].
Therefore, the series-parallel model is used in this work (see Fig.2).
The Universe Of Discourse (U.O.D) for both the input MFs and the output quantization levels are selected to be from –6 to 6. It should be noted that another range could also be selected since the input and output scaling factors can be modified genetically.
The following training input signal is used to train the FNI:

$$u_{train}(k) = \begin{cases} 0 & 0 \le k < 13 \\ -0.5 & 13 \le k < 63 \\ 0.5 & 63 \le k < 126 \\ 0.5\sin(\frac{k\pi}{9}) & 126 \le k \le 250 \end{cases}$$

While the best individual (FNI) in the GA is tested using the following test signal [14]:

$$u_{test}(k) = \begin{cases} 0.5\sin(\frac{k\pi}{5}) & 0 \le k < 63 \\ 0.5 & 63 \le k < 125 \\ -0.5 & 125 \le k < 188 \\ 0.5(0.3\sin(\frac{k\pi}{12}) + 0.1\sin(\frac{k\pi}{16}) + 0.6\sin(\frac{k\pi}{5})) & 188 \le k \le 250 \end{cases}$$

The real-coded GA is set to the following parameters:
Population size: 30

Maximum number of generations: 1000
Pc (crossover probability): 0.8
Pm (mutation probability): 0.05
The following nonlinear plants are to be identified by the FNI:

Plant 1: [13]

$$y(k)=\frac{y(k-1)}{1.5+y^2(k-1)}-0.3y(k-2)+0.5u(k-1)$$

Plant 2: [15]

$$y(k+1)=\frac{y(k)}{1+y^2(k)}+(u(k)+1).u(k).(u(k)-1)$$

Plant 3: [10]

$$y(k+1)=\frac{y(k)y(k-1)(y(k)+0.5)}{1+y^2(k)+y^2(k-1)}+u(k)$$

For plant3, we need only two input (linguistic) nodes in layer1 and hence, two fuzzy sets in layer2. Therefore, we need only (14) nodes in layers 2 and 3 of Fig. (1).
Figures (3), (4) and (5) show the output of the plant ($y_p$) and that of the FNI ($y_m$) along with the best MSE value against generations for each plant respectively.

## 6. Conclusions

In this paper, a genetically trained FNI, which combines the merits of FL, NNs and GAs, is used as a series-parallel identifier for nonlinear systems. A real-coding operators GA has been used to adjust the parameters of this FNI based on minimizing the MSE criterion. The difficult problem of finding the optimal values of the input and output scaling factors for the FNI has been overcome by making the GA find these optimal values. The simulation results showed the effectiveness of the proposed identifier in terms of the convergence accuracy to the real system.

## References

[1] Jacquot R.C.; "Modern Digital Control Systems", Marcel Dekker, INC. 1981.

[2] Brown M. and Harris C.; "Neurofuzzy Adaptive Modeling and Control"; Prentice Hall international (UK) Ltd.; 1994.

[3] Omatu S., Khalid M. and Yusof R.; "Neuro-Control and its Applications"; Springer-Verlag London Ltd, Second Printing, 1996.

[4] Fullér R.; "Introduction to Neuro-Fuzzy Systems"; Physica-Verlag; Heidelberg; 2000.

[5] Linkens D.A. and Nyongesa H.O.; "Learning Systems in Intelligent Control: an Appraisal of Fuzzy, Neural and Genetic Algorithm Control Applications"; IEE Proceeding on Control Theory Applications; Vol. 143; No. 4; PP. 367-386; July; 1996.

[6] Rzempoluck E.J.; "Neural Network Data Analysis Using Simulnet", Springer-Verlag New York, Inc.; 1998.

[7] Morles E.C. and Mort N.; "Neural Network-Based Adaptive Control Design"; Journal of Systems Engineering; Department of Automatic Control and Systems Engineering; University of Sheffield; P.O. Box 600; Mappin Street; Sheffield Sl 4DV; UK; 1993.

[8] Alcalā R., Casillas J., Cordōn O., Herrera F. and Zwir S. "Techniques for Learning and Tuning Fuzzy Rule-Based Systems for Linguistic Modeling and their Application"; Knowledge Engineering Systems Techniques and Applications; Vol. 3; Leondes (Ed.); Academic Press 2000.

[9] Ad'doory Q.F.; "Controlling a Rigid Body Satellite Using Fuzzy Neural Network"; M.Sc. thesis; Electrical Engineering Dept.; College of Engineering; University of Baghdad; 2000.

[10] Ismaeel S.A.; "Fuzzy-Neural Modeling of Dynamical Systems"; M.Sc. thesis; Control and Computers Engineering Dept.; University of Technology; 1997.

[11] Al-Said I.A.M.; Genetic Algorithms Based Intelligent Control; Ph.D. thesis, University of Technology, January 2000.

[12] Mitchell M.; "An Introduction to Genetic Algorithms" First MIT Press Paperback Edition, Cambridge 1998.

[13] Pham D.T. and Xing L.; "Neural Networks for Identification, Prediction and Control"; Springer-Verlag, London Ltd.; third printing, 1997.

[14] Al-Sibakhi R.K.; "Fuzzy Neural Control of Dynamical Systems"; M.Sc. thesis; Electrical Engineering Dept. College of Engineering; University of Mostansiriyah; 1996.

[15] Mohammed F.A.; "Neural Network Controller for Non-linear Systems"; M.Sc. thesis; Control and Computers Engineering Dept.; University of Technology; 1995.

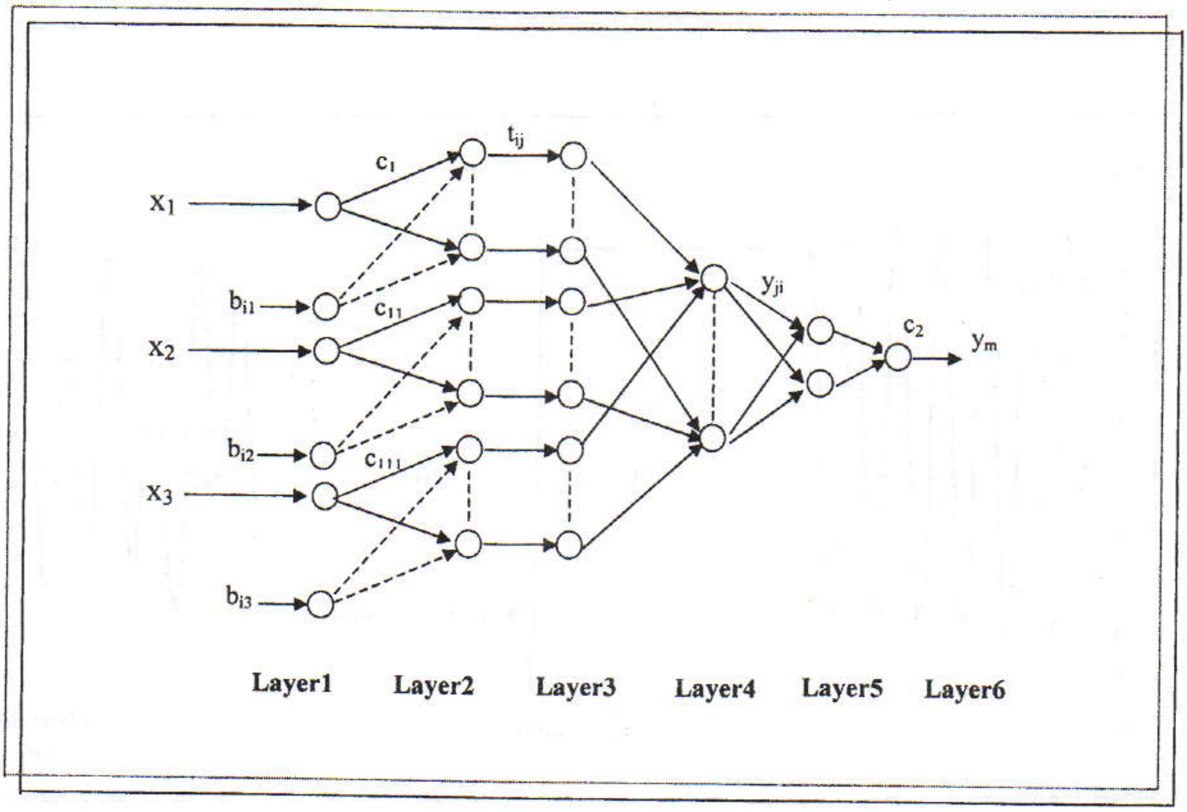



**Fig. (1) Structure of the FNI.**

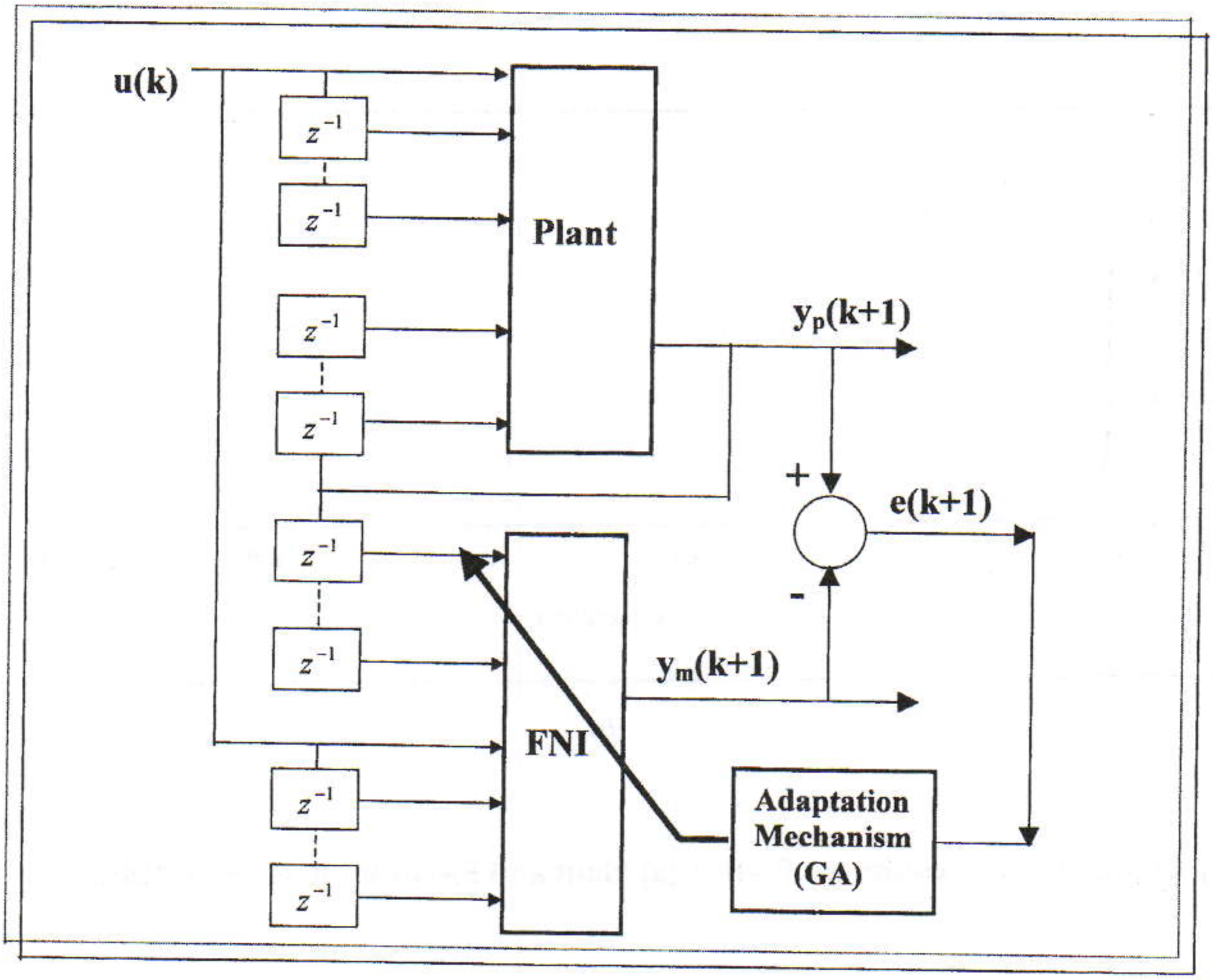
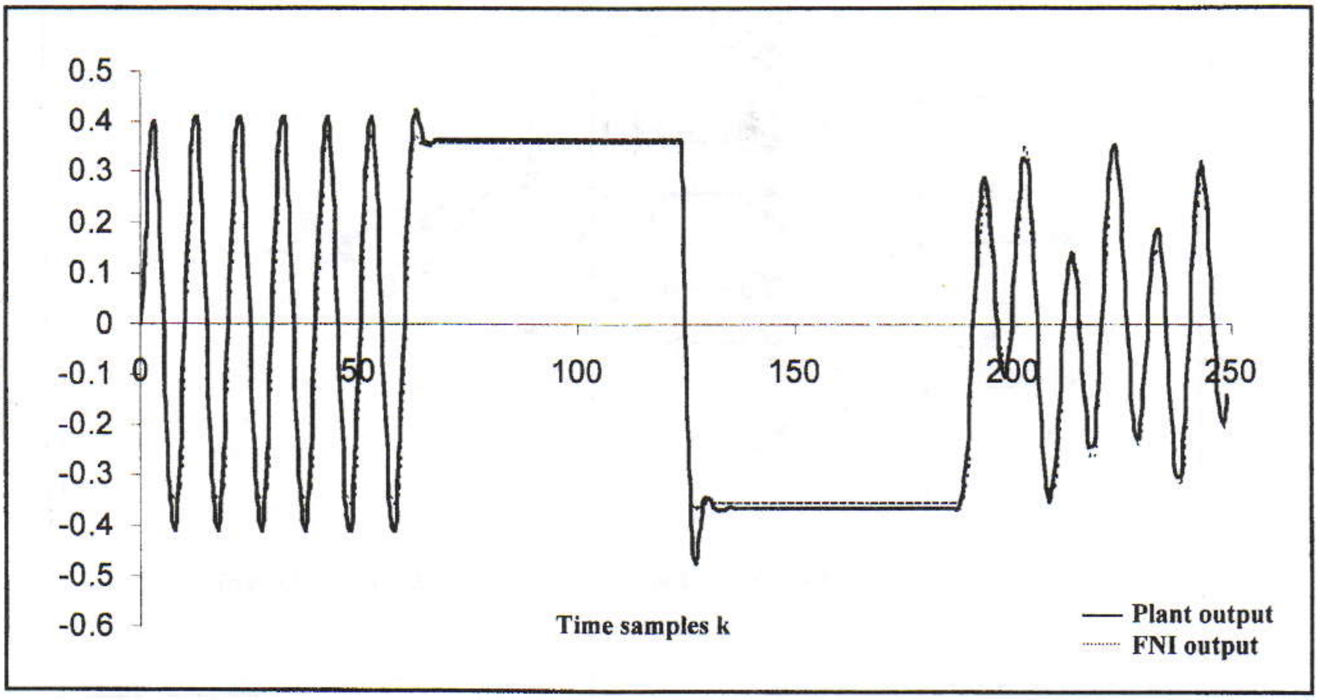
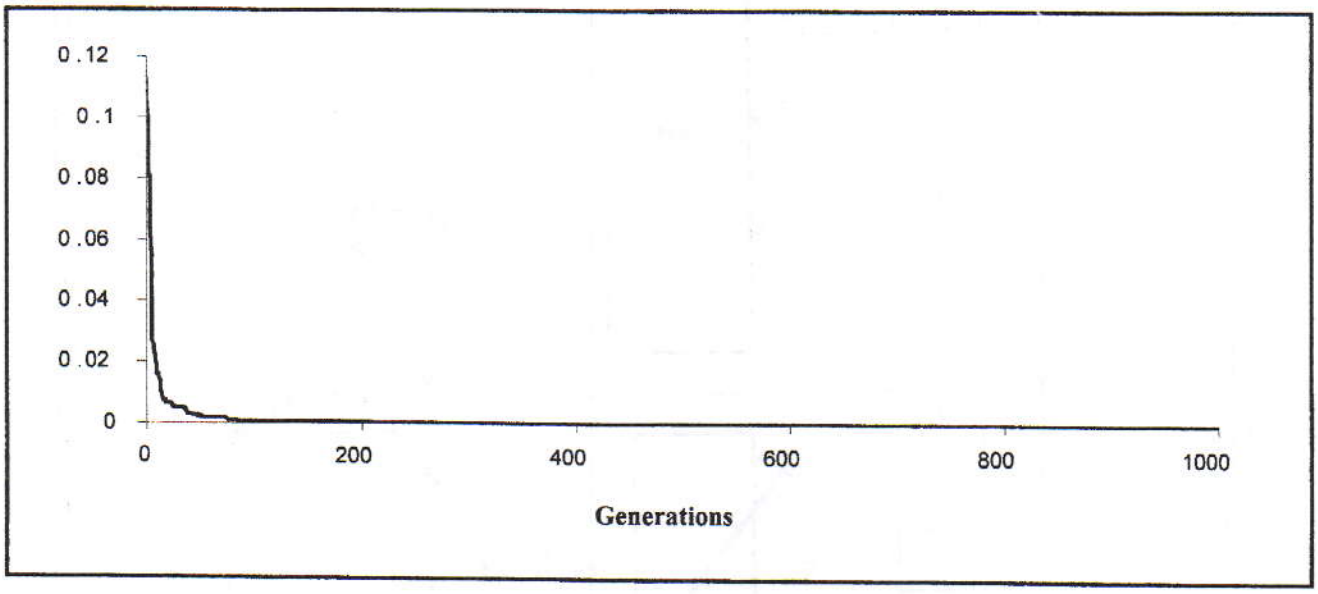



**Fig. (2) Series-parallel identification structure.**

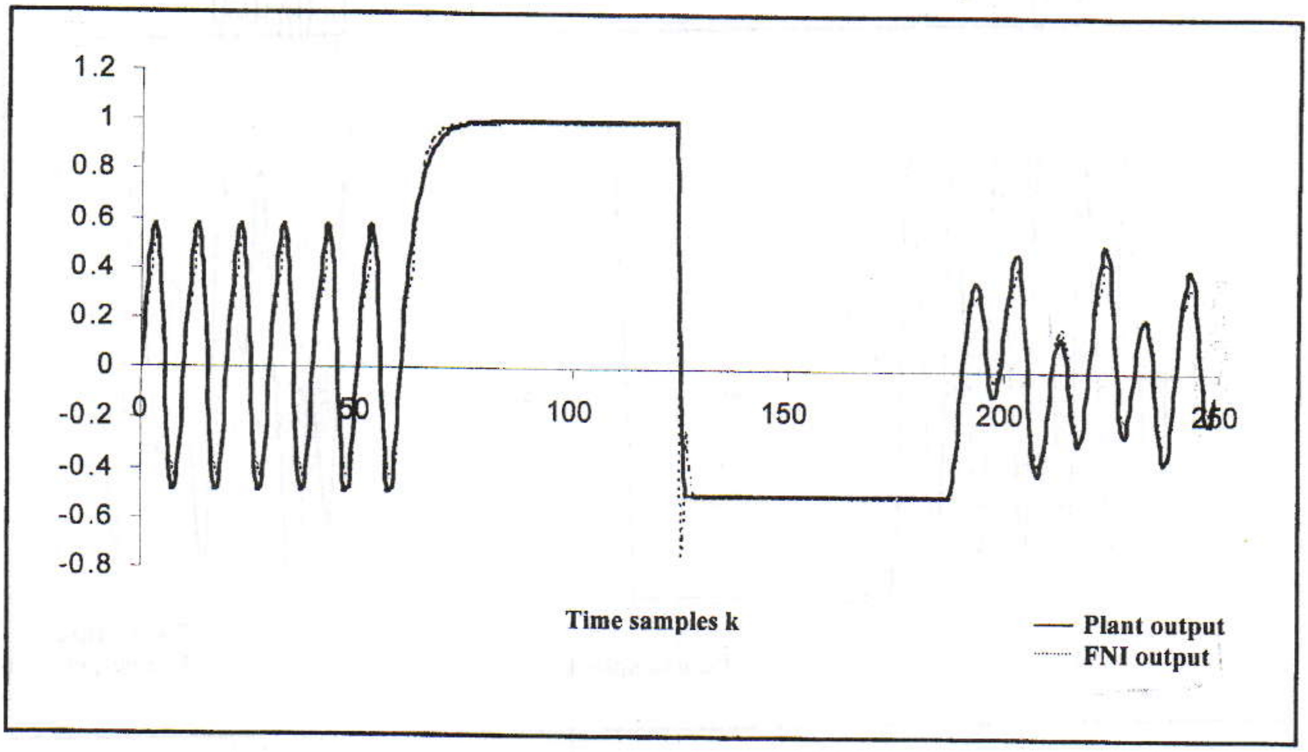



(a)

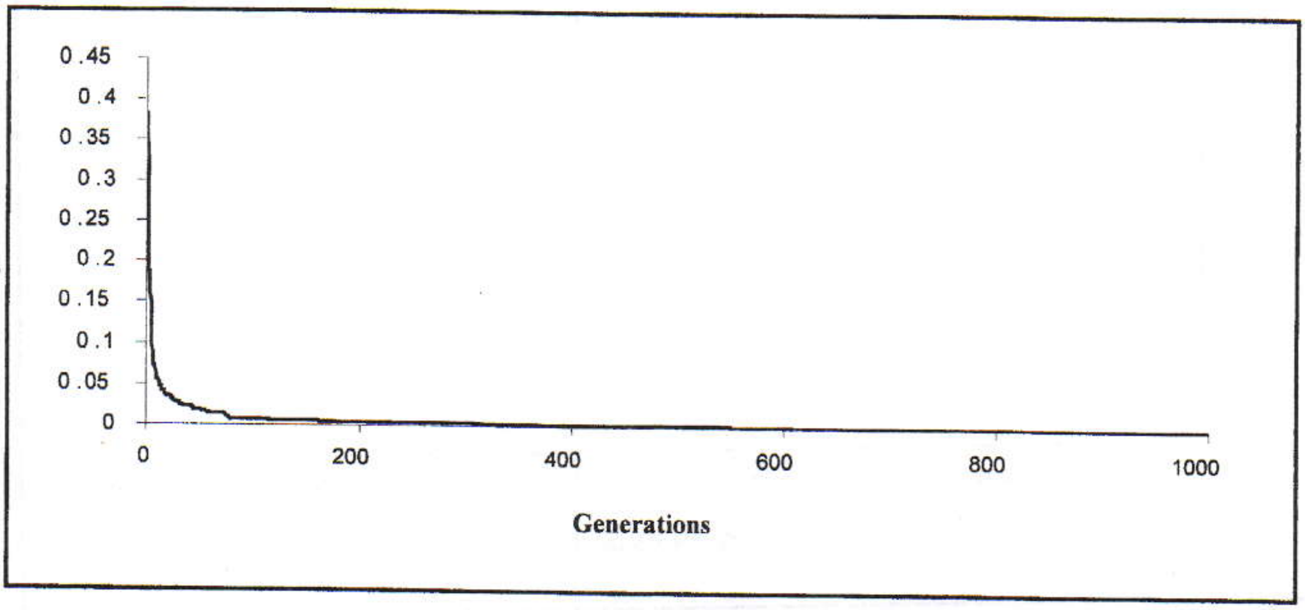



(b)

**Fig. (3) Simulation results for Plant 1 (a) plant and FNI outputs (b) best MSE.**

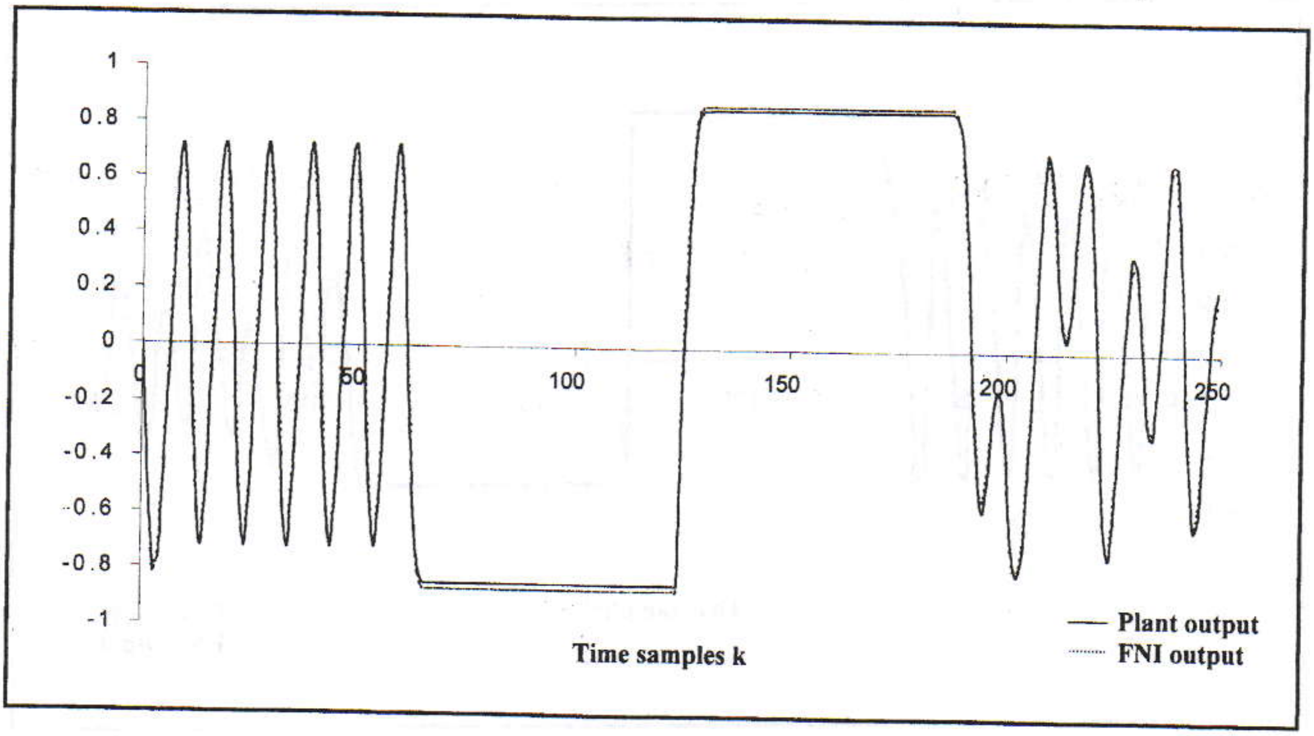



(a)

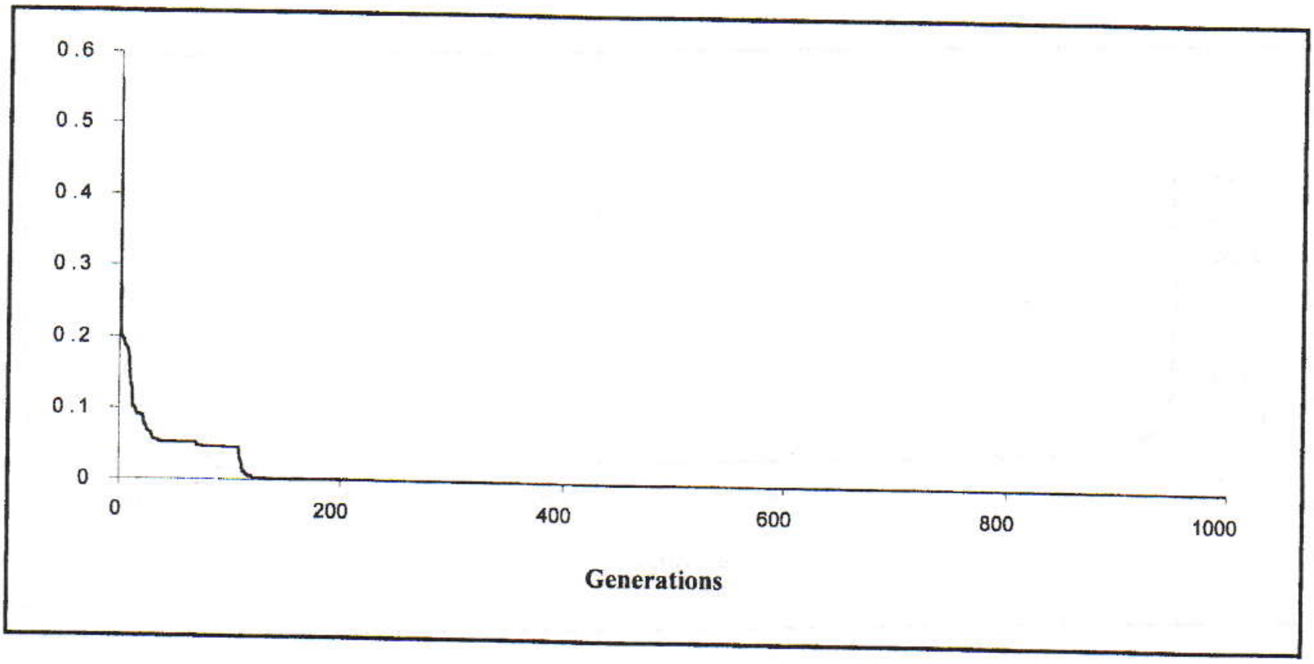



(b)

**Fig. (4) Simulation results for Plant 2 (a) plant and FNI outputs (b) best MSE.**




(a)




(b)

**Fig. (5) Simulation results for Plant 3 (a) plant and FNI outputs (b) best MSE.**