NONLINEAR SYSTEM MODELLING with FUZZY NEURAL NETWORK USING GENETIC ALGORITHMS with SIMULATED ANNEALING

Kadir ERKAN, Mehmet YILDIRIM
Kocaeli University, Anıtpark,
41300, İzmit-Kocaeli, TURKEY
E-mail: myildirim@kou.edu.tr

Abstruct– **In this paper, a nonlinear dynamic system has been modelled by using fuzzy neural network (FNN) topology. In this technique, system is divided into several fuzzy operating regions and linear reduced order models are used in each region to achieve approximation. System inputs are divided into several fuzzy sets, and membership functions for operating regions are produced. The overall output of the model is implemented by a defuzzification method. Genetic algorithms with simulated annealing is used for obtaining FNN weights. Genetic algorithm is used automatically and efficiently searching for a set of model weights or parameters for better performance.**

I. INTRODUCTION

This paper presents a nonlinear dynamic system modelling with fuzzy neural network using genetic algorithms with simulated annealing. The fuzzy neural network approach to nonlinear system modelling provides a way of openning up the purely black box approach normally seen in neural network applications.

In practice, many nonlinear systems are approximated by reduced order models, possibly linear. However, these models may only be valid within certain specific operating ranges. When operating conditions change, a different model may be required to be used or the model parameters may need to be adapted. One approach to the modelling of nonlinear system is to divide the whole envelope of the system operation into several operating regions, and use local reduced order model to aproximate the system in each region [1]. The local models may be of ARMAX (Autoregressive moving average vith exogenous inputs) [2] form. The definition of operating conditions is often vague in nature. It is generally not possible to precisely define system operating regions and there can be overlappings between various operating regions. Fuzzy sets provide an appropriate means to define operating regions. The input space of nonlinear system is divided into several fuzzy operating regions and local linear model is used in each region. The overall output is obtained through the center of gravity defuzzification.

The basic principles of genetic algorithms for problem solving are inspired by the mechanism of natural selection. Natural selection is a biological process in which stronger individuals are likely to be winners in competing environment. Genetic algorithms use a direct analogy of natural evolution. Genetic algorithms are global search techniques for optimisation but they are poor at hill-climbing. Simulated annealing has the ability of probabilistic hill-climbing. Thus, the two techinques are combined here to get a fine-tuned algorithm that yields a faster convergence and a more accurate search by introducing a new mutation operator like simulated annealing or an adaptive cooling schedule [3].

II. FUZZY NEURAL NETWORK TOPOLOGY

FNN topology shown in fig.1 is used for modelling a nonlinear system with fuzzy logic and neural network [4]. FNN is constructed from four layers. These are fuzzification layer, rule layers, function layer and defuzzification layer.

A-Fuzzification Layer

The inputs of fuzzification layer are system variables used for identifying fuzzy operating regions. Fuzzy sets like "*low*", "*medium*", and "*high*" [5] are used for system variables. Each neuron in this layer corresponds to a fuzzy set, and the output of this neuron is membership function of the fuzzy set. In this layer, three types of neuron activation function [6] are used. These are;

$$y = \frac{1}{1+e^{-(w_1 x + w_0)}} \quad \text{,for sigmoid} \qquad (1)$$

$$y = 1 - \frac{1}{1+e^{-(w_1 x + w_0)}} \quad \text{,for complement sigmoid} \qquad (2)$$

$$y = e^{-(w_1 x + w_0)^2} \quad \text{,for gaussian} \qquad (3)$$

B-Rule Layers

Rule layers implement fuzzy inference. Neurons in these layers use sigmoid activation function. Each input of neurons in the first rule layer corresponds to a fuzzy set. Each output of neurons in the second rule layer corresponds to a membership function for operating regions. These membership functions can be described as logical combination of system variables which are transformed into fuzzy sets.

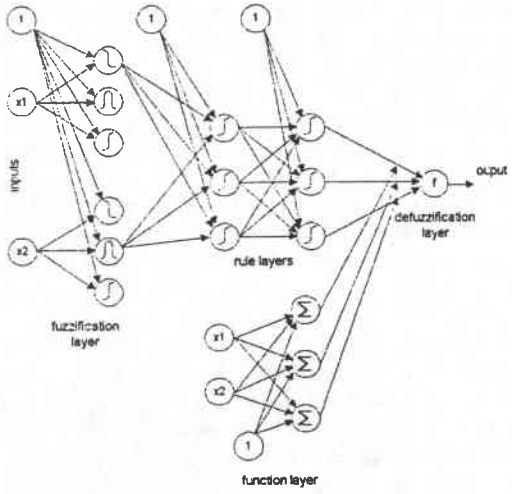

Fig.1. FNN topology.

C-Function Layers

Neurons in this layer implement the ARMAX models for operating regions. Each neuron corresponds to an operating region and it is linear. The outputs of neurons are sums of weighted system variables. The weights in function layer are parameters of linear models in operating regions. Any of operating regions can be described as (4). Biases used in neurons represent the constant terms in local models.

$$R_i:\ \hat{y}_i(t) = \sum_{j=1}^{no} a_{ij}\, y(t-j) + \sum_{j=1}^{ni} b_{ij}\, u(t-j) + c_i \qquad (4)$$

where; y is system output, u is system input, $\hat{y}_i$ are outputs of operating regions, nr is number of fuzzy operating regions, ni and no are time delays for system inputs and system outputs respectively, a_{ij} and b_{ij} are parameters of reduced order model, and t is time index.

D-Defuzzification Layer

The inputs of neuron in this layer are the output of operating regions and membership functions for these operating regions. Defuzzification layer implements the defuzzification with center of gravity method and forms the output of FNN model. No weights are used in this layer. Center of gravity method is given by (5).

$$\hat{y}(t) = \frac{\sum_{i=1}^{nr} \mu_i \hat{y}_i(t)}{\sum_{i=1}^{nr} \mu_i} \qquad (5)$$

where; μ_i is membership function for operating region, $\hat{y}_i$ is output of linear model, nr is the number of operating region, $\hat{y}$ is output of model.

III. GENETIC ALGORITHMS

Calculus-based search problems such as point to point method used in many optimisation methods from a singular point to next in decision space, using some transition rule to find next point, are risky because it is a perfect prescription for locating false peaks in many peaked search spaces. Some problems may be expressed by a parameter set. These parameters are regarded as chromosome genes and can be structured with a string of values in binary form. GAs work from a highly valued database of points simultaneously climbing many peaks in parallel; so, the probability of finding a false peak is reduced over methods that move point to point. Many search techniques need some auxiliary information to work acceptably. For example, gradient techniques need derivatives calculated analytically or numerically to be able to climb the peak. In contrast, GAs don't need this kind of secondary information, they only need objective function values. This characteristic makes GAs more canonical. In fact GAs are not considered as a mathematically guided algorithm. GAs are stochastic and non-linear processes and a final product containing the best or strongest elements of the previous generetions tends to be carried forward into the following generation. In other words, the rule is survival of the fittest. GAs use random choice as a mechanism to guide a search toward regions of the search spaces and probabilistic transition rules to get a trajectory for search.

GAs have some different aspects from the other optimisation methods:

1. GAs work with a coding of the parameter set, not the parameters themselves.
2. GAs search from a population of points, not a single point.
3. GAs use objective function information, not derivatives or other auxiliary knowledge.
4. GAs use probabilistic transition rules, not deterministic rules [7].

Initial population of size N is generated randomly. The user can define several strings if the user guesses

the model parameters for the different operating region of the system. User defined strings may be kept constant between generations so that actual parameter neighbourhood of these is found faster by GAs. In this paper no user-defined string is used. GAs work between sampling instants. Several generations can be made for each sample. The proposed algorithm can be summarised as follows.

1. Determine the parents according to fitness value.
2. Reproduce the children from parents.
3. Apply simulated annealing to new population.
4. Calculate fitness value of each string.
5. Select fittest string as model parameters.

The mechanics of genetic algorithms consist of copying and swapping partial strings. Simple GAs are composed of three operators: repruduction, cross-over and mutation.

IV. SIMULATED ANNEALING

Simulated annealing (SA) is a stochastic computational technique derived from statistical mechanics to find near global minimum cost solutions to large optimisation problems. It is a very important and powerful algorithm in optimisation and high order problems although it is very slow. Behaviour of very large systems of interacting systems in a thermal equilibrium at a finite temperature is studied in statistical mechanics. SA searches minimal energy states by using random processes. To improve simulated annealing, GAs can be merged with SA. This improved algorithm uses SA crossover and SA mutation operators instead of standard ones.

The set of system component spatial positions represents system configuration. If a system is in thermal equilibrium at temperature T, then the probability $\pi_T(s)$ that the system is in a given configuration s depends on the energy $E(s)$, and follows the Boltzman distribution:

$$\pi_T(s) = \frac{\exp\left[\frac{-E(s)}{kT}\right]}{\sum_{w \in S} \exp\left[\frac{-E(w)}{kT}\right]} \qquad (6)$$

where k si Boltzman's constant ans S is the set of all possible configurations.

Let at time i the system be in configuration q for the simulation of the behaviour of a system in thermal equilibrium. A candidate r for the configuration at time $i+1$ is generated randomly and accepted according to:

$$p = \frac{\pi_T(r)}{\pi_T(q)} = \exp\left[\frac{-(E(r) - E(q))}{kT}\right] \qquad (7)$$

If p is greater than 1, that means energy of r is less than energy of q, then configuration r is accepted as the new configuration for time $i+1$. If p is less than or equal to 1, then configuration r is accepted as the new configuration, with the probability p. Thus configurations of the higher energy states may be attained [8].

V. SIMULATION RESULTS

The system given by (8) has been studied as a nonlinear dynamic system [9] modelling,

$$\begin{aligned} y(t) = &\left[0.8 - 0.5\exp(-y^2(t-1))\right]y(t-1) \\ &-\left[0.3 + 0.9\exp(-y^2(t-1))\right]y(t-2) \\ &+u(t-1) + 0.2u(t-2) \\ &+0.1u(t-1)u(t-2) \end{aligned} \qquad (8)$$

Firstly, FNN topology was determined. Four system variables were divided into three fuzzy sets each. Therefore, twelve neurons were used in fuzzification layer. Three operating regions and according to this three local linear models were used in function layer. Two layers, three neurons in each, were used in rule layers. Center of gravity method was used in defuzzification layer. Genetic algorithms with simulated annealing was used for training fuzzy neural network. At the training stage, random population was used initially. Reproductions were made by using fitnesses which determined by the parameters and the system input-output data. After training FNN, best parameters shown in fig.2. were obtained.

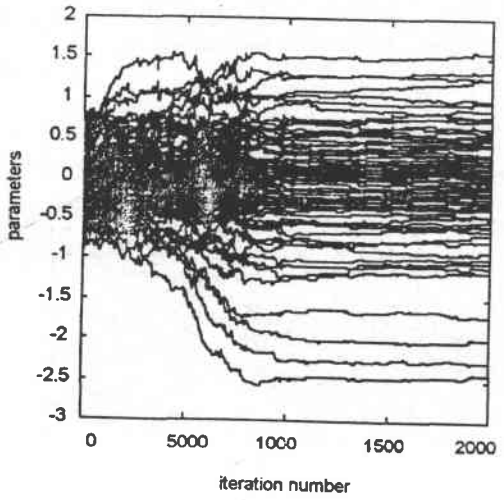

Fig.2. Model parameters.

In fig.3-6, nonlinear and linear output surfaces of the system, and model are shown respectively. The system output, the model output and the errors in the range of 0-100th samples are shown in fig.7. Furthermore, sum-squared error is shown in fig.8. It is seen that the sum-squared error very close to zero. In fig.9 and fig.10, maximum fitness and the average fitness are shown respectively.

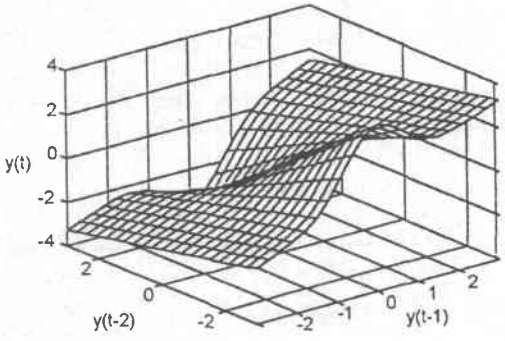

Fig.3. Nonlinear output of the system.

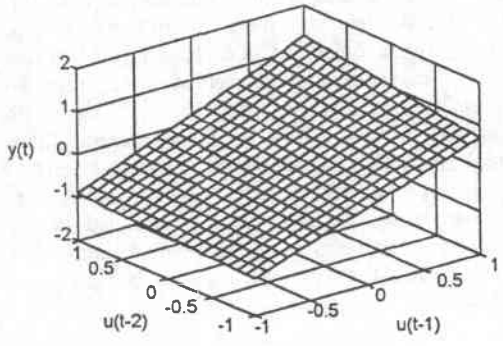

Fig.4. Linear output of the system.

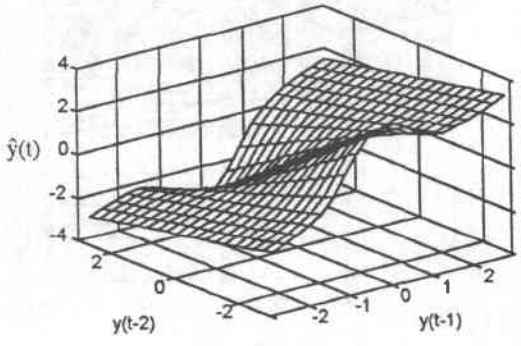

Fig.5. Nonlinear output of the model.

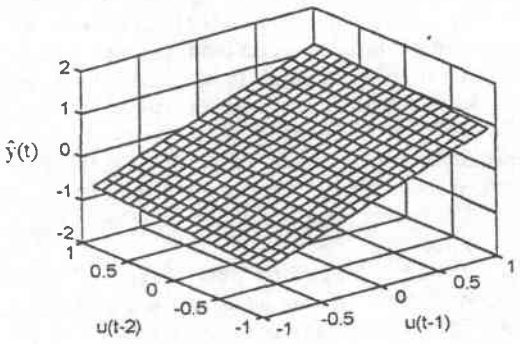

Fig.6. Linear output of the model.

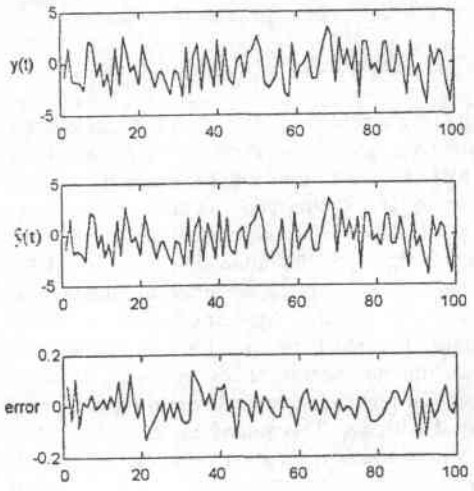

Fig.7. System output, model output and error.

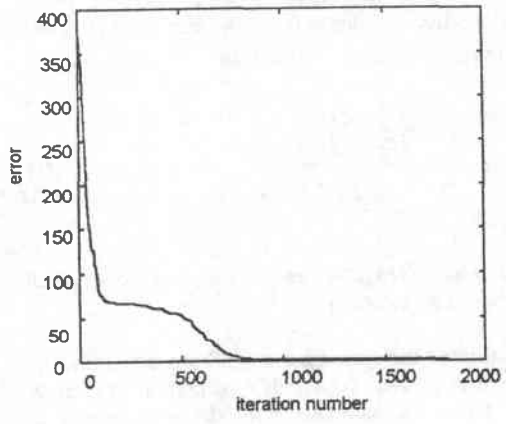

Fig.8. Sum-squared error.

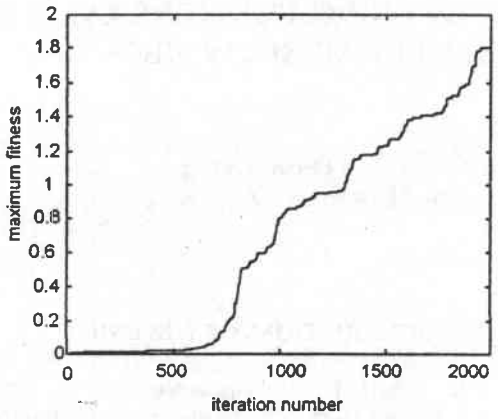

Fig.9. Maximum fitness.

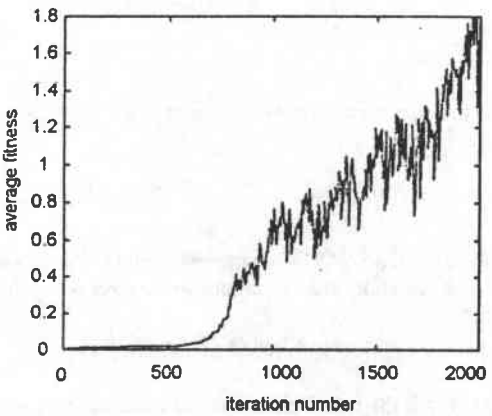

Fig.10. Average fitness.

VI. RESULTS

A nonlinear system modelling with fuzzy neural network using genetic algorithms with simulated annealing has been derived in this paper. The unknown parameters or weights of the model were estimated by using genetic algorithms with simulated annealing. Simulation results indicate that the model approaches system with the error close to zero. Even though the gradient based algorithms are succesful for training fuzzy neural network, genetic algorithms with simulated annealing is more appropriate since it is not risky in many peaked search spaces.

REFERENCES

1. TAKAGI, T., and SUGENO, M., 1985. Fuzzy Identification of Systems and It's Application to Modelling and Control. IEEE Transactions, SMC-15, (1), pp.116-132.

2. JOHANSEN, T.A., and FOSS, B.A., 1993. Constructing NARMAX Model Using ARMAX Models. Int. J. Control, 58, (5), pp.1125-1153.

3. ERKAN, K., BÜTÜN, E., 1998. Reactor Controller Design Using Genetic Algorithms With Simulated Annealing. Fuzzy Logic and Intelligent Technologies for Nuclear Science and Industry, Proceeding of 3rd International FLINS Workshop, Antwerp, Belgium, September 14-16.

4. ZHANG, J., MORRIS, A.J., 1995. Fuzzy Neural Network for Nonlinear System Modelling. IEE Proceedings, Control Theory Applications, vol.42, no.6, pp.551-561.

5. JAMSHIDI, M., VADIEE, N., ROSS, T.J., 1993. Fuzzy Logic and Control: Software and Hardware Applications. Prentice-Hall Inc. New Jersey.

6. NARENDRA, K.S., and PARTHASARATHY, K., 1991. Gradient Methods for the Optimisation of Dynamical Systems Containing Neural Networks. IEEE Transections on Neural Networks, vol.2, no.2, pp.252-262.

7. HOLLAND, J.H., 1992. Adaptation in Neural and Arificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. Cambridge: The MIT Press.

8. JEONG, I.K., and LEE, J.J., 1996. Adaptive Simulated Annealing Genetic Algorithm for System Identification. Eng. Appl. Artif. Intell.,vol.9, no.5, pp.523.

9. BOSSLEY, K.M., 1997. Neuro-Fuzzy Approaches in System Identification. University of Southampton, Faculty of Engineering and Applied Science, Electronics and Computer Science Department, Doctor of Philosophy Thesis, England.