Adaptive Neuro-fuzzy Inference System for two outputs

* T. Benmiloud *A. Omari

* Fac. of Electrical Engineering, Dept. of Electrotechnic
University of Sciences and Technology Oran
Laboratory of Automation and Control of Systems
BP 1505 El-Mnaouar - Oran, Algeria.
E-mail : tarek_bm2004@yahoo.fr

Abstract

In this paper we propose an architecture of adaptive neuro-fuzzy inference system (ANFIS) with two outputs. So as to prove its reliability, the proposed neuro-fuzzy inference system is used to make the approximation of two different functions. Simulation results show that this neuro-fuzzy system can approximates, with the desired precision, two functions at the same time.

Keywords: adaptive neuro-fuzzy inference system, neural network, training, gradient descent.

1 Introduction

A neuro-fuzzy system is based on a inference system formed by a training algorithm derived from the neural theory. By the association of the fuzzy system to neural network, the aptitude of training will be an advantage for the fuzzy system, and the transformation of linguistic rules will be an advantage for neural network.
There exists several approaches to integrate artificial neuron systems and the fuzzy logic, and very often the choice depends on the application. Jang and Sun **[1]** introduced the adaptive network-based fuzzy inference system ANFIS. This system makes use of a hybrid-learning rule to optimize the fuzzy system parameters of a first order Sugeno system.
Adaptive neuro-fuzzy system developed by Jang has an alone output; our goal is to propose an adaptive neuro-fuzzy system with two different outputs, which can be used in applications of systems with two outputs.

2 Adaptive Neuro-fuzzy Inference System for two outputs

The proposed adaptive neuro-fuzzy system for two outputs possesses a similar architecture to a classic adaptive neuro-fuzzy system, except a difference in the fourth layer. Architecture of the adaptive neuro-fuzzy system for two outputs and one-input first-order Sugeno fuzzy model with three rules is shown by Fig 1. **[2]**, **[3]**

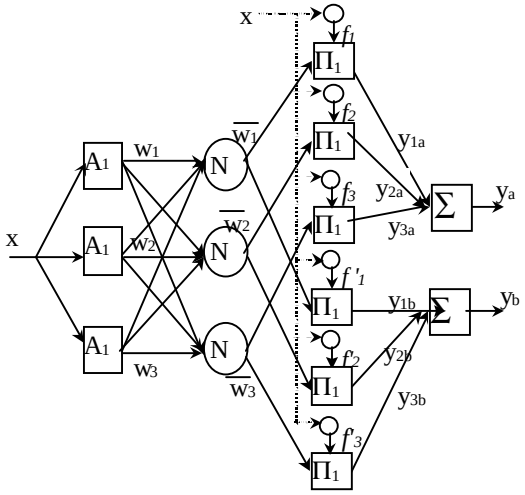


Fig 1: ANFIS for a one-input first-order Sugeno model with three rules architecture with two outputs

Output of the nodes in each respective layer is represented by *Oi*, where i is the i th node of layer l. The following is a layer-by-layer description of a one input one rule first-order Sugeno system.**[1],[3].**

Layer 1. Generates the membership grades:

$$O_i^1 = g(x)$$

g: function of membership of adaptivve neuro-fuzzy system. For our case, the chosen membership function is the trapezoidal function.

Layer 2. Generates the firing strengths.

$$O_i^2 = w_i = \Pi_{j=1}^{m} g(x)$$

Layer 3. Normalizes the firing strengths.

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2}$$

Layer 4. Calculates rule outputs based on the consequent parameters.

$$O_i^4 = y_i = \overline{w}_i \cdot f_i = \overline{w}_i \cdot (p_i \cdot x + q_i \cdot x + r_i)$$

$$O_i'^4 = y_i' = \overline{w}_i \cdot f_i' = \overline{w}_i \cdot (p_i' \cdot x + q_i' \cdot x + r_i')$$

Layer 5. Sum all the inputs from layer 4.

$$O_1^5 = y_a = \sum_i y_i = \sum_i \overline{w}_i \cdot f_i = \sum_i \overline{w}_i \cdot (p_i \cdot x + q_i \cdot x + r_i)$$

$$O_2^5 = y_b = \sum_i y_i' = \sum_i \overline{w}_i \cdot f_i' = \overline{w}_i \cdot (p_i' \cdot x + q_i' \cdot x + r_i')$$

In this last layer the consequent parameters can be solved by using the algorithm of least squares. Let us rearrange this last equation into a more usable form:

$$y_a = \begin{pmatrix} w_1 x_1 & w_1 x_2 & w_1 & w_2 x_1 & w_2 x_2 & w_2 \end{pmatrix} \begin{bmatrix} p_1 \\ q_1 \\ r_3 \\ p_2 \\ q_2 \\ r_2 \end{bmatrix}$$

$$y_b = \begin{pmatrix} w_1 x_1 & w_1 x_2 & w_1 & w_2 x_1 & w_2 x_2 & w_2 \end{pmatrix} \begin{bmatrix} p_1' \\ q_1' \\ r_3' \\ p_2' \\ q_2' \\ r_2' \end{bmatrix}$$

The neuro-fuzzy system for two outputs comprises an alone input, so, there are no rules of inference for this system, but there exist operations of fuzzification and defuzzification similar to that of neuro-fuzzy system for one output **[2]**.

2.2 Operation of training [5]

The ANFIS training paradigm uses a gradient descent algorithm to optimize the antecedent parameters, and a least squares algorithm to solve for the consequent parameters. The consequent parameters are updated first using a least squares algorithm, and the antecedent parameters are then updated by back-propagating the errors that still exist.

. The back-propagation of the gradient [2]

In the stage of back-propagation, the signal of error is back propagated and local parameters are updated by the method of gradient descent. For the neuro-fuzzy system to an alone output *y*, we have:

$$a_{ij}(t+1) = a_{ij}(t) - \frac{h}{p} \cdot \frac{\partial E}{\partial a_{ij}}$$

h: the training rate for a_{ij} ,

p : number of data of x (or y_d),

The following rule is used to calculate partial derivatives, employed to update the parameters of membership function *g*. **[1-6]**

$$\frac{\partial E}{\partial a_{ij}} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial y_i} \cdot \frac{\partial y_i}{\partial w_i} \cdot \frac{\partial w_i}{\partial g_{ij}} \cdot \frac{\partial g_{ij}}{\partial a_{ij}}$$

$$E = \frac{1}{2}(y - y_d)^2$$

E: the quadratic cost function,

Adaptive neuro-fuzzy system for two outputs as shown by Fig 1, possesses similar entry weights to these of neural system for an alone output (therefore similar local parameters a_i, b_i, c_i, d_i). The difference resides in consequent parameters. For neuro-fuzzy system for two outputs, each output possesses these clean consequent parameters (p_i , q_i, r_i for y_a , and p_i' , q_i' , r_i for y_b).

To make the local parameter correction, neuro-fuzzy system for two outputs uses the sum of the gradient of the two errors of the two outputs:

$e_1 = y_a - y_{d1}$

$e_2 = y_b - y_{d2}$.

Such that:

$$a_{ij}(t+1) = a_{ij}(t) - \frac{h}{p} \cdot \left(\frac{\partial E_1}{\partial a_{ij}} + \frac{\partial E_2}{\partial a_{ij}} \right)$$

where: $\frac{\partial E_1}{\partial a_{ij}} = f(e_1)$, $\frac{\partial E_2}{\partial a_{ij}} = f(e_2)$.

3 Application of ANFIS system to two outputs

To show the efficiency of the proposed neuro-fuzzy system for two outputs proposed, we consider the approximation of the two following functions:

$$y_{d1} = 0.005 \cdot x^3 - 0.002 \cdot x^2 - 0.03 \cdot x + 2$$

$$y_{d2} = -0.007 \cdot x^3 + 0.003 \cdot x^2 - 0.04 \cdot x + 5$$

with : $x = [-10, 10]$

The precision of neural system increases with the number of weight of entry. Neuro-fuzzy system for two outputs, concerns two errors of estimation (y_{d1} and y_{d2}).

To make the approximation of the two functions y_{d1} and y_{d2}, we have used a neuro-fuzzy system to 3 weights to the entry. Then, and to have best results of approximation, we have used a system to 4 weight of entry. Then we have made the comparison of the results of the approximation for the two neuro-fuzzy systems.

Local parameters are set to small values that we have chosen to accelerate the convergence. The membership function of neuro-fuzzy system that we have used is the trapezoidal function.

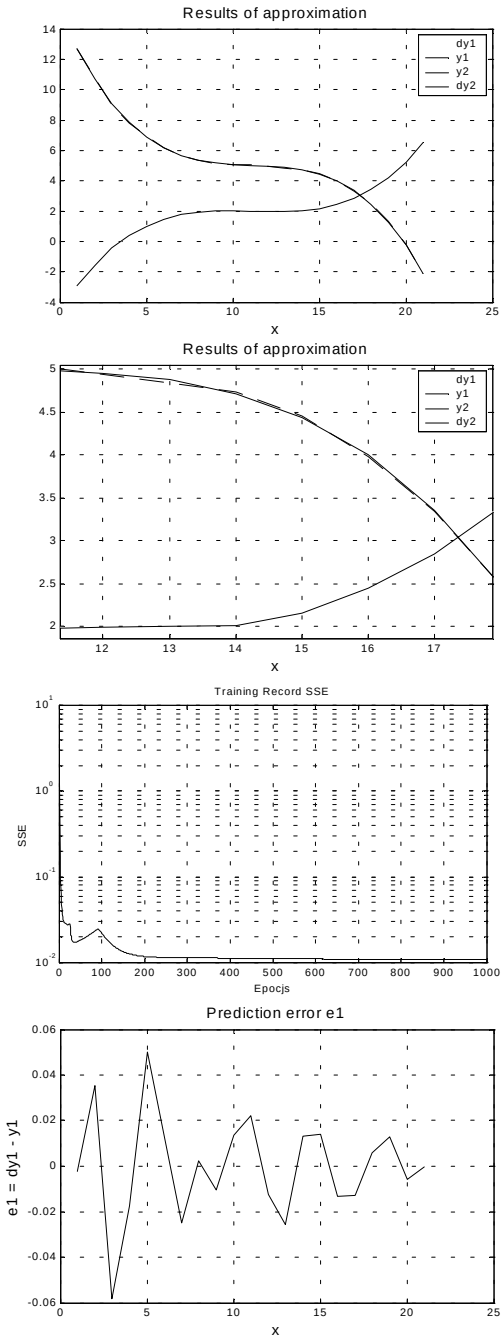
4 Simulation Results

To make the simulation of the neuro-fuzzy system, we have used the software Matlab 6.5. Next figures show results of the approximation of the two func-

tions y_{d1} , y_{d2} by the neuro-fuzzy system to 3 entries, and by neuro-fuzzy system to 4 entries. The evolution of square error SSE is also represented, as well as the two errors of approximation e_1 and e_2 for the two neuro-fuzzy systems.

SSE = $e^2 = (e_1 + e_2)^2$

For each of the two neuro-fuzzy systems, we have used N=1000 epochs to make the training.



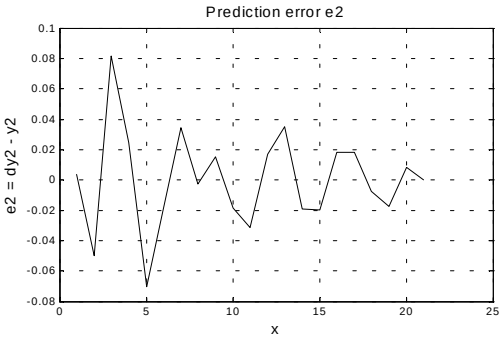
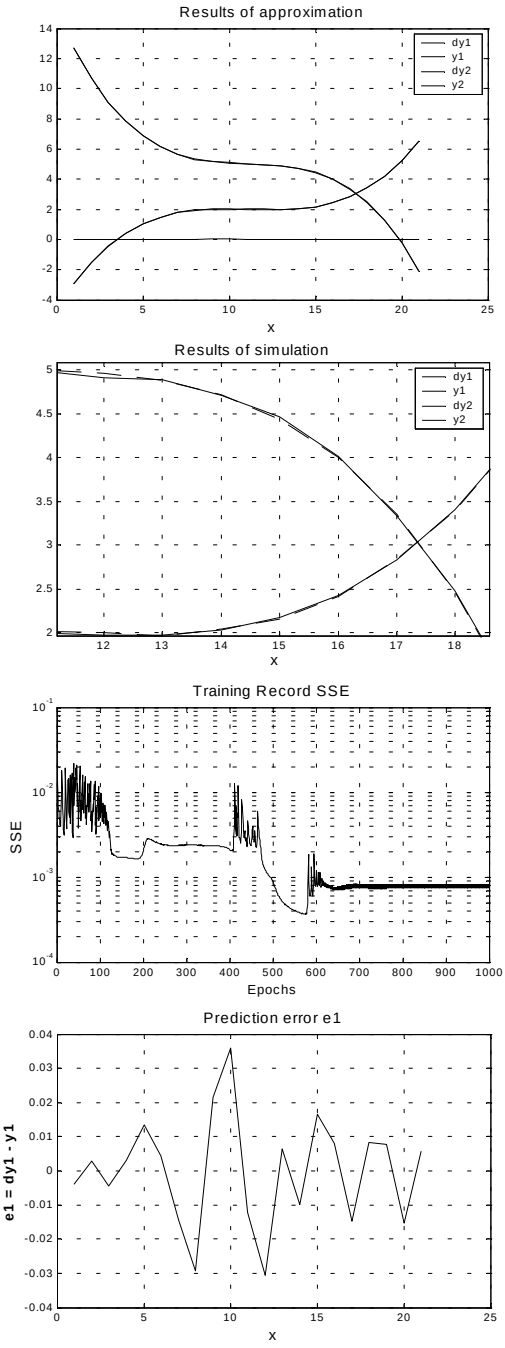


Fig 2: approximation of functions y_{d1} ,y_{d2} with neuro-fuzzy system to 3 weights to the entry



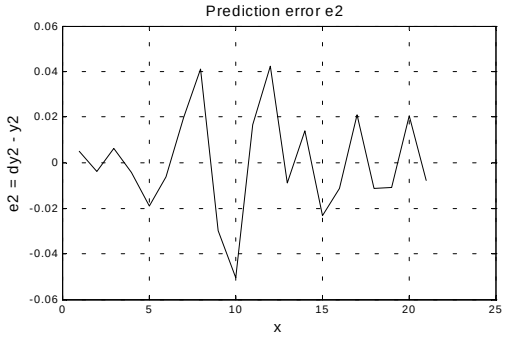


Fig 3: approximation of functions y_{d1} ,y_{d2} with neuro-fuzzy system to 4 weights to the entry

In these results of function approximation (of y_{d1} and y_{d2}), oscillations appear in the graph of the evolution of the square error (SSE) for the adaptive neuro-fuzzy system to 4 weights to the entry.
To avoid these oscillations, we use the gradient that has the greatest value between $\frac{\partial E_1}{\partial a_{ij}}$ and $\frac{\partial E_2}{\partial a_{ij}}$ (for the calculation of weight modifications), instead of the sum of these two gradients. Such that:

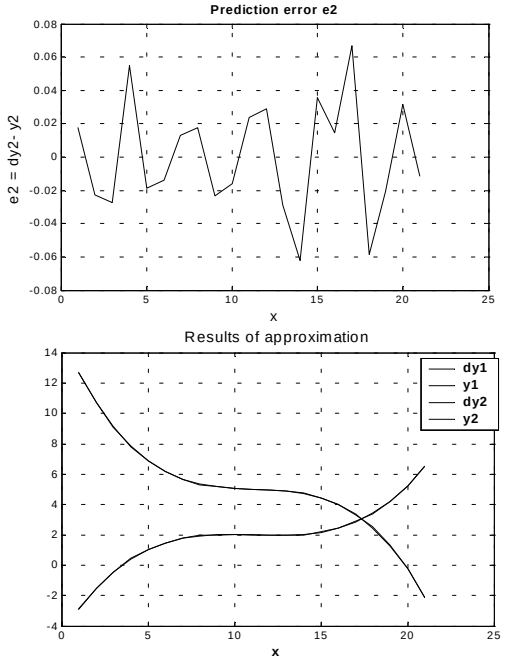
$$a_{ij}(t+1) = a_{ij}(t) - \frac{h}{p} \cdot \frac{\partial E}{\partial a_{ij}}$$

if $\frac{\partial E_1}{\partial a_{ij}} > \frac{\partial E_2}{\partial a_{ij}}$ **,** $\frac{\partial E}{\partial a_{ij}} = \frac{\partial E_1}{\partial a_{ij}}$

else if $\frac{\partial E}{\partial a_{ij}} = \frac{\partial E_2}{\partial a_{ij}}$

end

Results of the approximation of functions y_{d1}, y_{d2} by the neuro-fuzzy system, according to this approach is given by next figures:



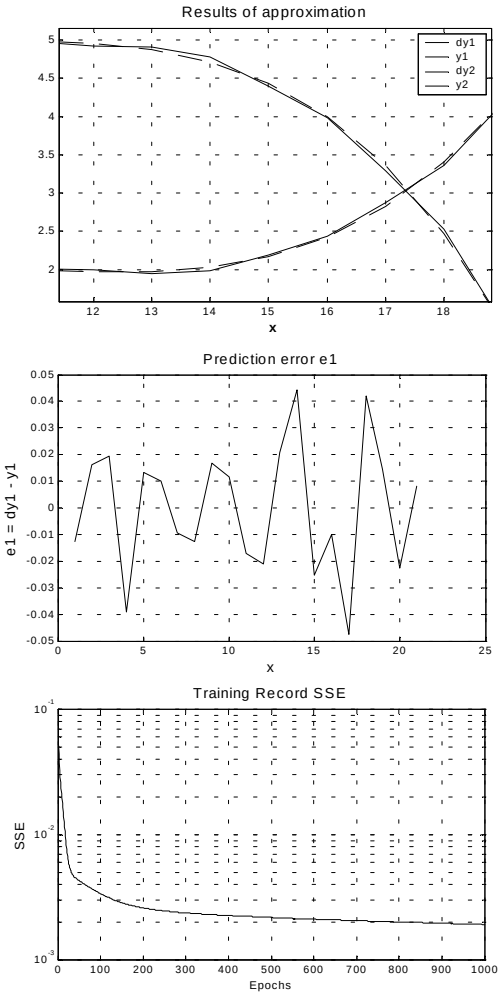


Fig 4: approximation of y_{d1} ,y_{d2} with neuro-fuzzy system (4 weight to the entry) with new approach

5 Interpretation of results

Figure. 2, Fig. 3 and Fig. 4 show that the proposed adaptive neuro-fuzzy system for two outputs, can approximate the two function y_{d1} and y_{d2} at the same time and with a great precision for the two functions.
The increase of the number of weight to the entry of neuro-fuzzy system (3 weight to 4 weight), allows to improve precision of the approximation of the two desired functions simultaneously:
- Errors of approximation (e_1 and e_2) are smaller for neuro-fuzzy system to 4 weight to the entry that these of neuro-fuzzy system to 3 weight to the entry.
- The square error (SSE) passes of meadows of 10^{-2} for neuro-fuzzy system to 3 weights to the entry to almost 10^{-3} for neuro-fuzzy system to 4 weights to the entry.

The neuro-fuzzy system to 3 weights to the entry converges more rapidly than neuro-fuzzy system to 4 weights to the entry.

6 Conclusions

Adaptive neuro-fuzzy system for 2 outputs has given good results of approximation of the two different functions y_{d1} and y_{d2}.
The increase of the number of weight of the neuro-fuzzy system allows improving the precision of approximation:

1. The Square of the sum of the two errors of approximation of the two functions is decreased of 10 times,
2. The values of the two errors of approximation of the two functions y_{d1} and y_{d2} have also decreased.

From that, we can say that the proposed adaptive neuro-fuzzy system for two outputs can approximates functions correctly and with good precision.

Bibliographie:

[1] J.S.R Jang, "Neuro-Fuzzy and Soft Computing", Prentice Hall, Upper Saddle River, NJ: 07458. [Jang, 1992, Jang and Gulley, 1995]

[2] J. Wesley, «Fuzzy and Neural Approaches in Engineering «, Hines New York 1997.

[3] Hongxing Li, C.L. PhiliD Ghen, Han-Pang Huang, *"* Fuzzy Neral Intelligent System: Mathematical foundation and the applications in engineering *"*, by CRC Press LLC 2001.

[4] Yu.Hen.Hu, Jenq-Neng Hwang, *«Introduction* to Neural Networks for Signal Processing *"*, by CRC Press LLC 2002.

[5] Lakhmi C. Jain and Berend Jan van der Zwaag and Ajith Abraham. *"Innovations in Intelligent Systems Design, Management and Applications* ", Springer, 2004.

[6] Xuan F. Zha, *"*Artificial Intelligence And Integrated Intelligent Information Systems-Emerging Tech-nologies And applications ", Idea Group Inc (IGI), 2006.