NEURO-FUZZY CONTROL OF A PH PLANT

M. J. Fuente [*] **G. I. Sainz** [**] **M. Alonso** [*] **A. Aguado** [***]

[*] *Department of Systems Engineering and Control. Faculty of Science, University of Valladolid. Prado de la Magdalena s/n, 47011 Valladolid, Spain. E-mail: maria@autom.uva.es*
[**] *ETSII. University of Valladolid. Paseo del Cauce s/n, 47011 Valladolid, Spain. E-mail:gresai@eis.uva.es.*
[***] *Department of Automatic Control, ICIMAF, Cuba.*

Abstract: This paper studies the control of a pH process by using a neuro fuzzy controller with gain scheduling. As the process to be controlled is highly non-linear the PI-type fuzzy controller that will be used generally is not able to control the system adequately. For this, a very simple feedforward neural network trained on-line, is put at the output of the PI-type fuzzy controller in order to calculate the gain of the controller. This neuro-fuzzy regulator has been tested in real-time on a bench plant. On-line results show that the designed control system allows the plant to operate in a range of pH values, despite perturbations and variations of the plant parameters, obtaining good performance at the desired workings points. *Copyright ©2005 IFAC.*

Keywords: Fuzzy controller, neural networks, Nonlinear process control, real pH plant, gain scheduling.

1. INTRODUCTION

Control of pH process plays an important role in chemical plants, such as biological, wastewater treatment, precipitation plants etc. However, it is difficult to control a pH process with adequate performance due to its nonlinearities, time-varying properties and sensibility to small disturbances when working near the equivalence point. Moreover, chemical plants where pH processes are important usually work with more than one type of product, with frequent product and grade changeovers, giving as a result transitions between different regimes. The control system must regulate the plant not only at these varying points, but also in transition regimes. Many different approaches to pH control has been implemented previously, such as Linear Adaptive, Model Based Predictive, Nonlinear Adaptive, Neural Network and Robust Controllers, (Loh *et al.*, 1995), (Palancar *et al.*, 1996),(Gomm *et al.*, 1996), (Klatt and Engell, 1996) and (Tadeo *et al.*, 1998).

Unfortunately, as noted by these authors, there are some weaknesses in these solutions: the control structures are quite complex, so they could be difficult to implement in existing distributed control systems. And a key difficulty seems to be the wide range of operation conditions over which good control is required, and the extent to which the dynamics vary as a consequence. In other words, the main difficulty remains the necessity of developing a model that represents adequately the pH process in any operating condition. Obtaining this precise non-linear model that accurately matches the plant at all working points is a difficult problem, so most of the advanced controllers are usually designed using a linear model of the process based on fix information of the plant that is imperfect and incomplete. Thus, control quality may deteriorate when working conditions change.

To solve this problem, this paper discusses a technique based on fuzzy control to design a pH controller. Fuzzy logic control (FLC) has been widely applied to industries in recent years (Biasizzo *et al.*, 1997), (Edgar and Postlethwaite, 2000), etc. A typical fuzzy controller is composed of three basic parts: a fuzzification block that transforms the continuous input signals into linguistic fuzzy variables, a fuzzy engine that carries out the rule inference where human experience can easily be injected through linguistic rules. And a defuzzification block that converts the inferred control action back to continuous signal that interpolates between simultaneous fired rules. So, two distinct features of fuzzy logic are that human experience can easily be integrated, so it does not necessitate a mathematical model of the system and that fuzzy logic provides nonlinear relationship induced by membership functions, rules and defuzzification. These features make fuzzy logic promising for process control where conventional control technologies do a poor job and human operator experience exists.

Most fuzzy controllers used are PI-type fuzzy controllers, which are not able to differentiate in which region the process operates, an important information necessary to control non-linear processes. In order to solve this problem, different approaches could be carried out, as a multiregional fuzzy controller (Qin and Borders, 1994), (Fuente *et al.*, 2002), but in this case a big number of fuzzy rules is necessary to reach an accuracy controller. Another possibility is presented in this paper, where a neuro-fuzzy controller is designed as a gain scheduling controller, and the gain is calculated through a feedforward neural network trained on-line put at the output of a classical fuzzy controller. This neural network is trained with the control error (e) and the change in the control error (Δe), and its output is the gain of the controller, that is changing in real time based on the process output. Such neuro-fuzzy controller can compensate the process non-linearity, so the control performance is more uniform.

The organization of the paper is divided as follows, Section 2 presents the neuro-fuzzy controller, the fuzzy characteristics and the neural network training procedure. Section 3 describes the real plant where the fuzzy controller is applied, a pH neutralization process, with its mathematical model. Section 4 gives the results for this application and finally in Section 5 some conclusions are given.

2. NEURO-FUZZY CONTROLLER

Most industrial processes present considerable nonlinearity with respect to different regions of operation, as occurred in the pH process. To design a fuzzy controller that gives satisfactory performance for different regions of gain nonlinearity, a neuro-fuzzy controller is used (Figure 1). In addition to use a classical PI-type fuzzy controller with the control error (e) and change in the control error (Δe) as inputs, and the change in the signal control (Δu_f) as output, a neural network is used at the output of the fuzzy controller. This network

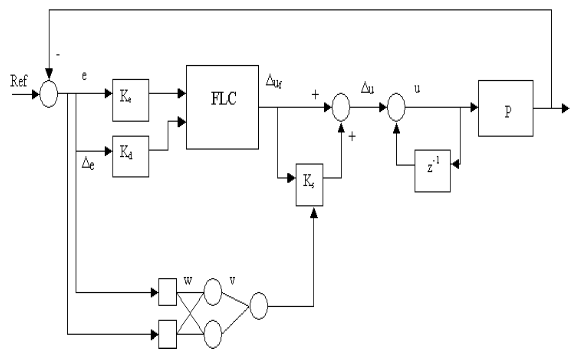

Figure 1. The fuzzy controller with neural gain scheduling

calculates the gain of the controller (K_s), which can change with the time, so the output of the neuro-fuzzy controller (Δu) is now:

$$\Delta u(t) = \Delta u_f(t)(1 + K_s) \tag{1}$$

Rule definition: a general fuzzy inference rule for this controller that have two inputs and one output is:

$$If\ e\ is\ A_i\ and\ \Delta e\ is\ B_i\ \ then\ \ make\ \Delta u_f\ C_i$$

where A_i, B_i and C_i are adjectives for the error (e), change in the error (Δe) and change in the control action (Δu_f) respectively. These adjectives could be descriptors such as *Negative Small*, *Positive Large*, *Zero* and so on. A fundamental requirement for these rules is that they have to perform negative feedback control for the sake of stability and they have to be adjusted by experience of human operators.

Neural network definition: The neural network used is a very simple multilayer perceptron with two neurons in the hidden layer and just one neuron in the output layer. And it is trained on-line with a simplified version of the backpropagation algorithm. To describe its basic principle, the following notation is used: $E(t)$ represents the square error function, which is the function to be optimized:

$$E(t) = \frac{1}{2}\sum_{k=1}^{t} e(k)^2 \tag{2}$$

where $e(k)$ is the control error, i.e, the difference between the set point and the system output $y(k)$ at time k:

$$e(k) = Ref(k) - y(k) \tag{3}$$

So, the error to be minimized here, eq. 2, differs of the traditional one used in the backpropagation algorithm, where it is a function of the network output error.

The network has two inputs: $x = [e(k) \quad \frac{de(k)}{dt}]$, two neurons in the hidden layer with the hyperbolic logarithm function as activation function, and a neuron in the output layer with linear activation function, to calculate the controller gain, K_s, i.e., the network topology is

$$
\begin{aligned}
S_i &= \Sigma_{j=1}^{2} w_{ij} x_j \qquad i = 1,2 \\
h_i &= \frac{1}{1+e^{-S_i}} \qquad i = 1,2 \\
K_s &= v_1 h_1 + v_2 h_2
\end{aligned}
\tag{4}
$$

where S_i is the input to the activation function in the hidden layer, w_{ij} represents the weights for the connections between the input layer and the hidden layer, h_i is the activation function in the hidden layer and v_i are the connection weights between the hidden and the output layers. Applying the backpropagation algorithm implies that each parameter of the network (the weights w_{ij} and v_i) is adapted by the following relationships, where η is the learning rate:

$$
\begin{aligned}
w_{ij}(k) &= w_{ij}(k-1) - \eta \frac{\partial E(k)}{\partial w_{ij}} \\
v_i(k) &= v_i(k-1) - \eta \frac{\partial E(k)}{\partial v_i}
\end{aligned}
\tag{5}
$$

Now, in order to calculate the gradient of the cost function $E(t)$ with respect to the coefficients as it is shown in eq. 5, it is necessary to apply the chaining rule:

$$
\begin{aligned}
\frac{\partial E(t)}{\partial w_{ij}} = {} & \frac{\partial E(t)}{\partial e(k)} \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial \Delta u_k(k)} \frac{\partial \Delta u_k(k)}{\partial K_s} \frac{\partial K_s}{\partial h_i} \\
& \frac{\partial h_i}{\partial S_i} \frac{\partial S_i}{\partial w_{ij}}
\end{aligned}
\tag{6}
$$

$$
\frac{\partial E(t)}{\partial v_i} = \frac{\partial E(t)}{\partial e(k)} \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial \Delta u_k(k)} \frac{\partial \Delta u_k(k)}{\partial K_s} \frac{\partial K_s}{\partial v_i} \tag{7}
$$

where $\Delta u_k(k) = K_s \, \Delta u_f(k)$.

It is possible to solve, all the partial derivatives in eqs. 6 and 7 easily, except for $\frac{\partial y(k)}{\partial \Delta u_k(k)}$, which is unknown if a model of the process is not available. But this equation can be substituted by its sign, which is the sign of the process gain (Cui and Shin, 1992). Solving those derivatives, using the eqs. 2, 3 and 4, the weights update becomes:

$$
\begin{aligned}
w_{ij}(k) = {} & w_{ij}(k-1) + \eta e(k) sign(\frac{\partial y(k)}{\partial \Delta u_k(k)}) \\
& \Delta u_f(k) v_i h_i (1-h_i) x_j(k)
\end{aligned}
\tag{8}
$$

$$
\begin{aligned}
v_i(k) = {} & v_i(k-1) + \eta e(k) sign(\frac{\partial y(k)}{\partial \Delta u_k(k)}) \\
& \Delta u_f(k) h_i
\end{aligned}
\tag{9}
$$

The eqs. 8 and 9 can be calculated in real time, because the reduced size of the network (just four coefficients for w_{ij} and two for v_i). The most important parameter here is the learning rate η. With large values of this parameter the speed of the convergence increases, but the system can become unstable. And with low values of η the behavior is better but the response is slower.

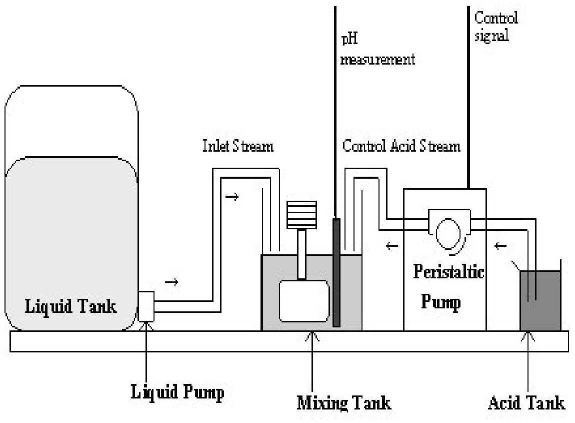

Figure 2. Laboratory plant

In this case the parameter η, is variable depending on the control error, i.e., with large errors the learning rate is faster, and with low regulation errors the learning rate is slower.

$$
\eta = \alpha |e(k)| \tag{10}
$$

with the α parameter changing in real time, from a very little value, 10^{-6} by example, at the beginning of the learning task and it can increase slowly until reach the desired behavior.

3. DESCRIPTION OF THE EXPERIMENTAL SETUP

In this study the process is to neutralize an aqueous solution of sodium acetate (CH_3COONa) titrated with hydrochloric acid (HCl) in a continuous stirred tank reactor (CSTR). The experimental setup is shown in Figure 2. An overflow (not shown) system is applied on the CSTR; therefore the volume can be considered constant. The control variable u is the flowrate of the titrating stream which is feed using a peristaltic pump (ISMATEC MS-1 REGLO/6-160). The output variable y is the hydrogen ions in the effluent stream, measured as pH. The mixture pH is measured using a Ag-AgCl electrode (Kent 1180/700) and transmitted using a pH-meter (Kent EIL9143). The pH measured and the control signal are transmitted through an A/D interface (ComputerBoards CIO-AD16, 0-5V). The plant is controlled and monitored from a Pentium II computer. The MATLAB software packages of Fuzzy logic and Real Time Workshop toolboxes are used to implement the controller. The sampling time was selected to be 2 seconds, after some open loop experiments, at different points.

In order to filter the pH measurement noise, so that it does not introduce an appreciable delay in the process, an elliptic filter of order 2 has been chosen, with cut-off frequency at 0.3142 rad/s:

$$
F(s) = \frac{0.0361s^2 + 1.0289}{s^2 + 1.3771s + 1.0301} \tag{11}
$$

3.1 Plant model

The model of the plant has been obtained based on first principles, and then validated in the real plant, by carrying out experiments at different working points. Assuming that the liquid is a pure dissolution of sodium acetate, the hydrochloric acid has constant concentration, the temperature is constant (25C) and there are perfect dissolution, mixing and no additional buffering effects, the following model can be obtained (Fuente *et al.*, 2002):

$$\begin{aligned} & -x_A + 10^{-pH} - 10^{pH-14} + \frac{x_s}{1+10^{pKS+pH-14}} = 0 \\ & V\frac{dx_A}{dt} = F_A C_A - (F_A + F_S)x_A \\ & V\frac{dx_S}{dt} = F_S C_S - (F_A + F_S)x_S \\ & \tau\frac{dpH^*}{dt} = pH - pH^* \end{aligned} \tag{12}$$

where the ion concentrations within the tank (and in the effluent stream) are $x_A = [Cl^-]$, $x_S = [Na^+]$. F_A is the control acid stream flowrate, F_S the inlet stream flowrate, C_S the concentration of sodium acetate in the inlet stream, C_A the acid concentration in the control acid stream, the measured pH is pH^*, which is affected by a time constant τ, and $pKS = -log_{10}k_S$. The dissociation constants are:

$$\begin{aligned} k_w &= 10^{-14} mol^2 L^{-2} \\ k_S &= 5.7143x10^{-9} mol L^{-1} \end{aligned} \tag{13}$$

4. NEURO-FUZZY CONTROL OF A PH PROCESS

First, a classical fuzzy PI-type controller, has been designed for this plant, and it is necessary to define the fuzzy membership functions associated with the controller inputs, the control error e, and the change in the control error Δe, and with the controller output (Δu_f) based on prior knowledge about the process. The number of membership functions for each variable can vary, depending on the resolution required for that variable. Generally, more membership functions offer more degrees of freedom to the functional relationship of the controller. However, it requires more effort to implement it. Figures 3, 4 and 5 show the membership functions selected for this problem (Alonso, 2003). It should be noted that the change in the control action (Δu_f) usually requires higher resolution than other variables because the controller must be able to make aggressive and less aggressive control actions depending on the process nonlinearity. In these figures the meaning of the adjectives are: NL= negative large, NM= negative medium, NS=negative small, ZO= zero, PS=positive small, PM= positive medium and PL=positive Large.

These figures show that the discourse universe for all variables are normalized between $[-1, 1]$, and the

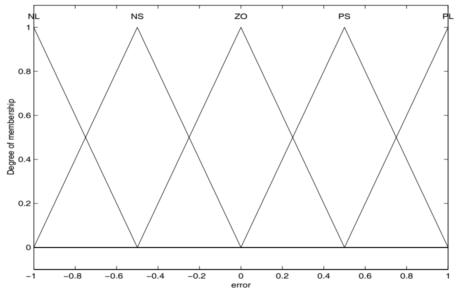

Figure 3. Membership functions definition for the control error for the fuzzy controller.

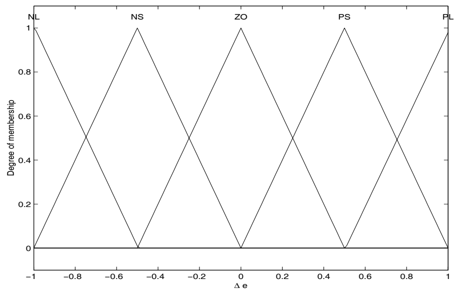

Figure 4. Membership functions definition for the change of control error for the fuzzy controller.

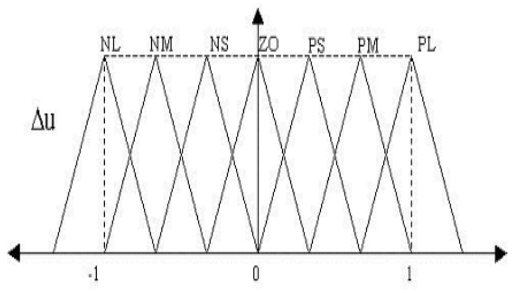

Figure 5. Membership functions definition for the output of a classical fuzzy PI-type controller.

membership functions are symmetric in this interval, because all the variables are scaled as:

$$\begin{aligned} e^* &= \frac{e_m}{K_e} \\ \Delta e^* &= \frac{\Delta e_m}{K_d} \\ \Delta u_f^* &= \frac{\Delta u}{(1+K_s)} \end{aligned} \tag{14}$$

where $e_m = \frac{e}{emax}$, and $\Delta e_m = \frac{\Delta e}{\Delta emax}$, with $emax$ and $\Delta emax$ are the maximum values of the control error and change in the control error, respectively. And K_e and K_d are scaling factors for e and Δe respectively and are tuning parameters for the fuzzy controller, and K_s the scale factor corresponding to the output of the controller (Δu_f) is calculated by the neural network in real time. Moreover, as it is shown in (Qin and Borders, 1994), these parameters are related with the tuning parameters of a PI control as follows: $K_p = 0.5((1+K_s)/K_d)$ and $T_i = \Delta t(K_e/K_d)$ where K_p is the proportional gain, T_i the integral time and Δt

Table 1. Fuzzy rules for a fuzzy PI-type controller

	Δe				
e	NL	NS	ZO	PS	PL
NL	*PL*	*PL*	*PL*	*PM*	*ZO*
NS	*PL*	*PL*	*PM*	*ZO*	*NM*
Z0	*PM*	*PM*	*ZO*	*NM*	*NL*
PS	*PS*	*ZO*	*NM*	*NL*	*NL*
PL	*ZO*	*NS*	*NM*	*NL*	*NL*

the sampling time. In this work these parameters are selected as follows: $S_e = 1.25$ and $S_{\Delta e} = 22$, by a trial and error procedure.

The next step is to define the system knowledge in terms of fuzzy rules (Alonso, 2003), which is shown in Table 1, where the rules can be interpreted as:

$$\{If \quad e \quad is \quad PL \quad and \quad \Delta e \quad is \quad PL \quad then \quad make \quad \Delta u_f \quad NL\} \tag{15}$$

This fuzzy controller has been implemented using the Fuzzy Logic Toolbox of MATLAB, and the data is acquired/sent to the real plant through the Real Time Toolbox of MATLAB, as it is shown in Figure 6. Where it is possible to see the filter (eq. 11) to filter the pH measurement, the fuzzy controller, the control action saturation because that control action only can move between $[0-100\%]$, a S-function where it is defined the neural network and its training algorithm explained in Section 2, and a filter in the change of the control error (Δe) in order to filter the noise high frequencies, with transfer function:

$$F_1(s) = \frac{-0.1s}{s+0.1} \tag{16}$$

The negative gain in this transfer function is due to use the output process derivative instead the control error derivative, as expressed in the following equations:

$$Error = reference - output.$$
$$\frac{dError}{dt} = \frac{-doutput}{dt} \tag{17}$$

To examine the control performance, the set point of pH is changed from 3.5 to 5, 6 and finally 4. Figure 7 depicts the dynamic response with the set point changes, Figure 8 shows the control action in that situation and Figure 9 shows the neural network output (K_s). One can see that positive and negative step changes result in a different time responses and different behaviour, due to the non-linearity of the process. And it is possible to see, that the value of K_s only change when there is a set point change, and the output of the systems goes to a new value. And in each operation point the controller gain K_s is stabilized around a different value, showing the plant non-linear gain. By the way, these results show that the real plant controlled with this neuro-fuzzy control operates adequately.

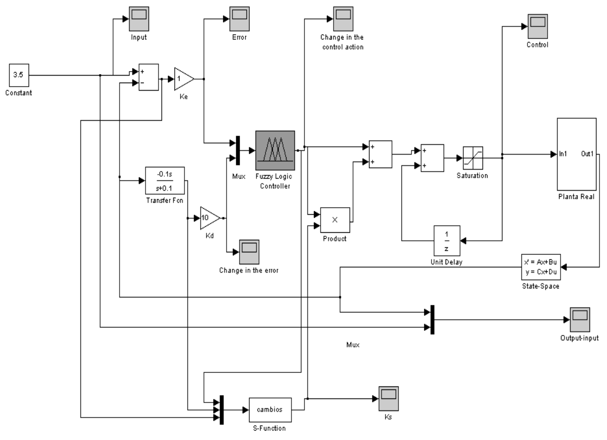

Figure 6. MATLAB file generated for the control of the real plant using the neuro-fuzzy controller designed in this work.

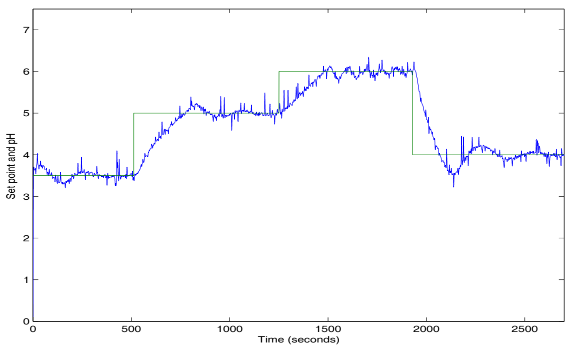

Figure 7. Step response of the pH process controlled with the neuro fuzzy controller

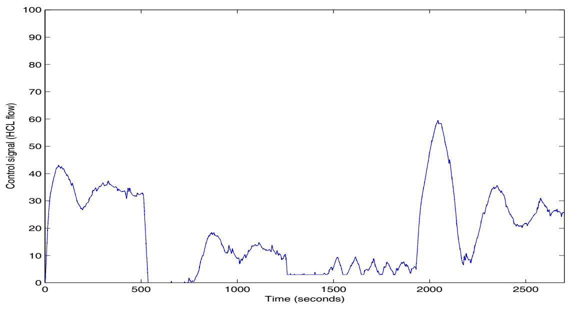

Figure 8. Control actions

Figure 10 shows the performance of the controller when there exist disturbances. At time $t = 350$ seconds, when the system is the stationary point of $pH = 3.5$, the water flowrate has been increased a value of 10% in its magnitude. It is possible to see in Figure 10 that the system does not change appreciably, only there are modifications in the control action. With this disturbance a change in the set point is carried out at time 730 seconds, form $pH = 3.5$ to $pH = 4$, the results show that this change is easily recovered. Figure 11 shows the control actions for that situation and Fig. 12 shows the temporal evolution of K_s.

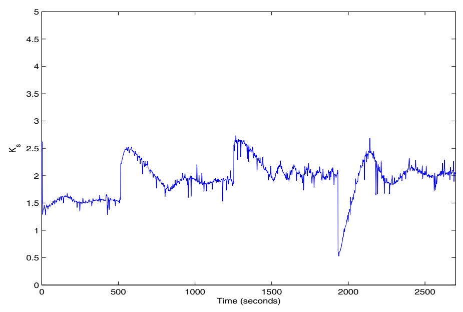

Figure 9. Neural network output (K_s)

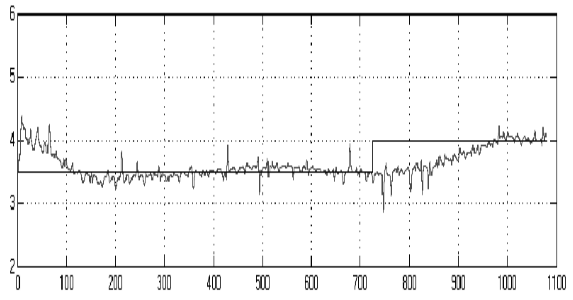

Figure 10. Step response of the pH process controlled with the neuro fuzzy controller, when a perturbation exits at $t = 350$ s

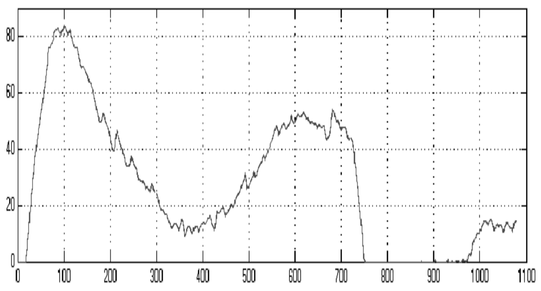

Figure 11. Control actions

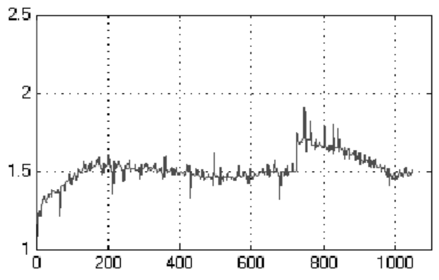

Figure 12. Neural network output

5. CONCLUSIONS

This paper has presented the design of a novel neuro-fuzzy controller with gain scheduling for a pH process at laboratory scale. Which uses a classical PI-type fuzzy controller with a neural network at the output of the controller, trained in real time to calculate the gain of the controller. The inputs to the controller are: the control error and the change of the control error and in order to implement the PI-type fuzzy non-linear regulator, the controller output signal is the change in the control action. The results obtained show that the performance of this neuro-fuzzy controller is very adequate to control highly non-linear process, without the necessity of a complex mathematical model of the system.

6. ACKNOWLEDGEMENTS

This work has been supported by the research national agency of Spain (CICYT) throughout the project DPI2003-09373

REFERENCES

Alonso, M. (2003). Control neuro-difuso de una planta de pH. Technical report. Dpto. Ingeniería de Sistemas y Automática. Universidad de Valladolid. Valladolid, Spain (In spanish).

Biasizzo, K.K., I. Skrjanc and D. Matko (1997). Fuzzy predictive control of highly nonlinear pH process. *Computers Chem. Eng.* **21**, s613 – s618.

Cui, X. and K.G. Shin (1992). Direct control and coordination using neural networks. *IEEE Trans. on Systems, Man and Cybernetics* **23**(3), 686–697.

Edgar, C. R. and B.E. Postlethwaite (2000). MIMO fuzzy internal model control. *Automatica* **34**, 867–877.

Fuente, M.J., C. Robles, O. Casado and Fernando Tadeo (2002). Fuzzy control of a neutralization process.. In: *IEEE Conf. On Control Applications Glasgow, U.K.*

Gomm, J.B., S.K. Doherty and D. Williams (1996). Control of pH in-line using a neural predictive strategy. In: *UKACC, International Conference on Control*.

Klatt, K.U. and S. Engell (1996). Nonlinear control of neutralization process by gain-scheduling trajectory control. *Ind. Eng. Chem. Res.* **35**, 3511–3518.

Loh, A.P., K.O. Looi and K.F. Fong (1995). Neural network modelling and control strategies for pH process. *Journal of Process Control* **6**, 355–362.

Palancar, M.C., J.M. Aragon, J.A. Miguens and J.S. Torrecilla (1996). Application of a model reference adaptative control system to pH control. effects of lag and delay time.. *Ind. Eng. Chem. Res.* **35**, 4100–4110.

Qin, S. J. and G. Borders (1994). A multiregion fuzzy logic controller for nonlinear process control. *IEEE Trans. on Fuzzy Systems* **2**, 74 – 81.

Tadeo, F., A. Holohan and P. Vega (1998). L1 optimal regulation of a pH control plant. *Computers Chem. Eng.* **22**, 459–466.