An Adaptive Control Algorithm for Multiple-Input Multiple-Output Systems Using Neural Networks

JOSE NORIEGA
Departamento de Investigación en Física
Universidad de Sonora
Rosales y Blvd. Luis Encinas, Col. Centro, CP 83000, Hillo, Son.
MEXICO

Abstract: - Many industrial processes have multiple inputs and outputs. Such systems are known by the acronym MIMO. In order to properly control such processes some linear techniques like multivariable control, robust control, etc., have been used [1]-[3]. These techniques require a *known* model of the controlled process. However, many industrial processes are poorly known and nonlinear. Therefore, a technique to model and control nonlinear and partly known systems is needed. It is known that artificial neural networks, particularly Multi-Layer Perceptrons, partly fulfill such requirements. In this paper the gradient descent optimization rule is combined with a trained neural network model for the computation of the control vector [5],[7]. The computed control vector would drive the nonlinear MIMO system outputs to the desired operating point. For this purpose, a quadratic cost function is defined. The computation of a new control vector requires the previous computation of the gradient of the process. Since direct information of the gradient of the process is not available, an accurate approximation of the systems gradient is computed from the neural network model. The general performance of the controller is illustrated using simulations. A simple mathematical model of coupled tanks systems is used for this purpose [10].

Key-Words: - Neural networks, control, stability, modeling, optimization.

1 Introduction

Conventional process control requires the previous deduction and characterization of a mathematical linear model of the process. The model must be accurate and reliable in order to achieve an acceptable control performance [1]-[3]. To produce a model, the control engineer needs to incorporate available data. This may be a difficult task since in many cases reliable information of the process may not be available or incomplete. In addition, many industrial processes have multiple inputs and outputs. Hence, the number of unknown parameters increases resulting in a more complex modeling task. The existing control methods applied to MIMO systems also need reliable linear models of the process. Some of these techniques are LQR, H_∞, frequency domain techniques (Nyquist), etc. [2]. When an analytical model of a systems is not available, alternative modeling techniques may be applied. Amongst these techniques are artificial neural networks and fuzzy logic.

Many neural network control approaches have been tested [5]-[7],[9]. Shiffman [6], illustrates an adaptive control using a direct plant control neural network using a generalized predictive control strategy. It is demonstrated that the proposed method drives the system accurately to the desired response. However, the training process of the neural network may be time consuming and it is demonstrated only for single input single output systems. Narendra [7], applied recurrent artificial neural networks to control nonlinear systems. Narendra approach to control is shown to be effective. The method has the inconvenience of requiring a long training time for the recurrent neural network controller. Narendra also illustrated that the method could be extended to MIMO nonlinear systems.

In this work, a multi-layer perceptron neural network has been used for the modeling of a MIMO process. This is a two layer neural network using a **tanh()** activation function. It is known that this type of neural network are capable to approximate the time response of a nonlinear system to an anticipated accuracy [7],[8].

The backpropagation algorithm has been applied for the training process of the neural network [6]. The gradient descent algorithm is used to compute the control vector [5]. In order to use the gradient descent rule, a quadratic cost function is defined first.

The cost function consist of the square of the difference between the desired system output and the neural network output plus the square of the difference of present and past control vectors. It is assumed that the minimization of the cost function on each iteration will reduce the control error and

would therefore lead to an stable control of the MIMO system.

For the implementation of such control scheme it is necessary to carry out the computation of the gradient of the system. Since the system is assumed to be poorly known and nonlinear, such information is not readily available. Hence, an accurate estimate of the gradient of the system is produced by computing the gradient from the neural network model.

The results illustrated in this work were generated using simulations based on a nonlinear coupled tanks model [10]. The coupled tanks simulation supplied the required data used for on line training of the neural network model.

The rest of the paper is organized in five sections. Section two is focused on the description of the neural network model. Section three describes the mathematical development of the control algorithm from the proposed cost function. Section four illustrates the simulation results for normal control conditions. Section five illustrates the unstable operation due to poor selection of the controller parameters. Finally, section six are the conclusions of this work.

2 Neural Network Model

Artificial neural networks have been successfully applied in the modeling of dynamic systems [9]. To produce an accurate predictive model, the data supplied to the neural network must be properly organized. This implies the use of previous knowledge of the process.

The NARMAX (Nonlinear Auto Regressive Moving Average with eXogenous input) framework has been combined with multi layer perceptrons to produce nonlinear dynamical models [6],[7].

The neural network architecture is embedded in the NARMAX framework. Part of the architecture of the neural network (number of neurons in the hidden layer and the number of inputs) is defined by the orders of the process and the number of inputs and outputs.

In this work a multi-layer perceptron is trained using the backpropagation algorithm. The training algorithm requires appropriate excitation to the inputs of the process to produce meaningful data. As a result of the training process, the weights of the neural network would retain information of the response of the process. The predictive neural network model is illustrated in Figure 1.

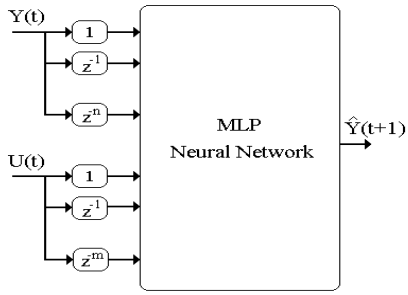

Figure 1, NARMAX framework for predictive model

Figure 1, depicts the orders of the system as ***n*** and ***m***. The model output is a prediction of the next output sample from the process. The accuracy of the model depends on the training of the neural network, a proper choice of ***n*** and ***m*** and a proper selection of training data sets. Figure 2 illustrates the interconnection of the neural network model and the process inputs and outputs.

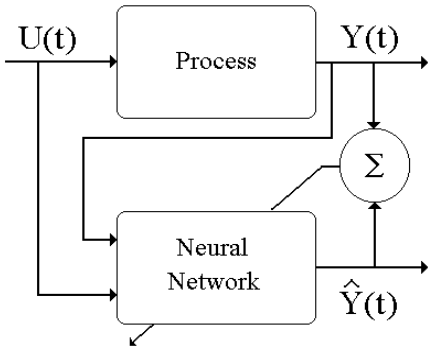

Figure 2, Predictive neural network model and process

The variables shown in Figure 2 are, the vector of the process inputs $U(t) = (u_1(t), u_2(t), ..., u_m(t))$, the vector of process outputs $Y(t) = (y_1(t), y_2(t), ..., y_n(t))$ and the estimation of the process outputs $\hat{Y}(t) = (\hat{y}_1(t), \hat{y}_2(t), ..., \hat{y}_n(t))$. In general the neural network model is written as

$$\hat{Y}(t+1) = W^o(\tanh(W^i P)) \qquad (1)$$

This is a two layer neural network where, W^o, W^i y P are the synaptic weights of the output layer, the weights of the input layer and the input vector respectively.

3 Control Algorithm

The control algorithm is based on the direct application of the gradient descent rule to a quadratic cost function. A successful minimization step of the quadratic cost function would lead to the

computation of a new control vector that would reduce the difference of the system output and the desired system output.

For this purpose the quadratic cost function is defined as.

$$J = \frac{1}{2}[E_s{}^T Q E_s + E_c{}^T R E_c] \quad (2)$$

Where the error E_s is defined as.

$$E_s = Y_d(t+1) - \hat{Y}(t+1) \quad (3)$$

This is the difference between the desired process output and the actual neural network output. The error E_c is.

$$E_c = U_c(t) - U_c(t-1) \quad (4)$$

Which is the difference of the actual control vector and the previous control vector. It is necessary to indicate that it is expected that the error E_c progressively reduces to cero during the control action. Finally, Q and R are positive semi-definitive weighting matrices.

The gradient descent rule requires the computation of the gradient of the cost function. The computation of the control vector is therefore expressed as.

$$U_c(t+1) = U_c(t) - \eta_c \frac{\partial J}{\partial U_c(t)} \quad (5)$$

Here, $U_c(t) = (u_1, u_2, ..., u_m)$ is the control vector and η_c, is the control gain. The gradient of the cost function is.

$$\frac{\partial J}{\partial U_c(t)} = -\frac{\partial \hat{Y}(t+1)}{\partial U_c(t)} Q E_s(t) + IREc \quad (6)$$

Where $\hat{Y}(t) = (\hat{y}_1, \hat{y}_2, ..., \hat{y}_n)$ is the neural network vector output. The gradient of the neural $\partial \hat{Y}(t+1) / \partial U_c(t)$ is expressed in the Equation (7).

$$\frac{\partial \hat{Y}(t+1)}{\partial U_c(t)} = \{W^o \sec h^2(W^i P(t))\} \{(W^i \frac{\partial P(t)}{\partial U_c(t)})\} \quad (7)$$

Here, the operation in between keys indicates an element by element multiplication for the two vectors. The input vector is.

$$P(t) = (y_1, y_2, ..., y_n, u_1, u_2, ..., u_m) \quad (8)$$

The result of Equation (7) is represented in a matrix known as the Jacobian. The Jacobian of the neural network model is.

$$\frac{\partial \hat{Y}(t+1)}{\partial U_c(t)} = \begin{bmatrix} \frac{\partial y_1(t+1)}{\partial u_1(t)} & \frac{\partial y_2(t+1)}{\partial u_1(t)} & \cdots & \frac{\partial y_n(t+1)}{\partial u_1(t)} \\ \frac{\partial y_1(t+1)}{\partial u_2(t)} & \frac{\partial y_2(t+1)}{\partial u_2(t)} & \cdots & \frac{\partial y_n(t+1)}{\partial u_2(t)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1(t+1)}{\partial u_m(t)} & \frac{\partial y_2(t+1)}{\partial u_m(t)} & \cdots & \frac{\partial y_n(t+1)}{\partial u_m(t)} \end{bmatrix} \quad (9)$$

Finally, the control vector is computed by adding expressions from Equation (7) into Equation (9) and the resulting expression is substituted in Equation (5). As a results, the control vector is computed using Equation (10).

$$U(t+1) = U(t) + \eta_c [\frac{\partial \hat{y}(t+1)}{\partial U(t)} Q E_s + IRE_c] \quad (10)$$

Here, the gradient of the cost function is combined with Equation (6) and the gradient descent rule as shown in Equation (5). This control algorithm is graphically represented in Figure 3 in block diagram form.

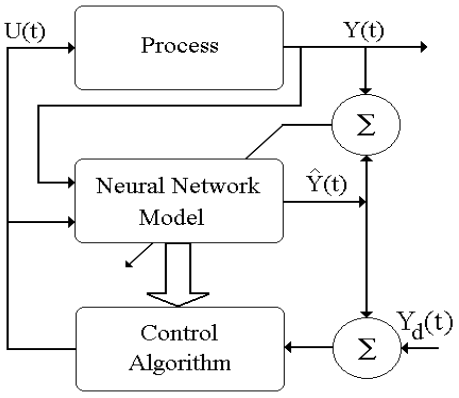

Figure 3, Block diagram of the controller

This diagram shows the control loop and the neural network training loops. It is assumed that the training stage took place previously to the activation of the control stage. The weight matrices Q and R are chosen to compensate the measured output vector and the measured control vector. Such a control algorithm requires few computational resources for its implementation. It is also easy to program in a digital computer.

However, it is not possible to guarantee the stability of the controlled process using this algorithm. This is due to the fact that the gradient

descent rule may get stuck in local minima [9]. In addition, the gradient descent rule is an slow converging algorithm. Therefore, the computation of the control vector may not be appropriate for processes with a rapid response.

4 Simulations

The performance of the control algorithm described herein is illustrated by simulations, using a nonlinear coupled tanks model. Two tests are presented. First, the response of the control algorithm is simulated for previous defined values of the desired response vector. Second, the control algorithm is tested for disturbances. Also a description of the mathematical model of the nonlinear coupled tanks is presented.

4.1 Nonlinear coupled tanks model

The nonlinear coupled tanks process consist of a pair of tanks having constant transversal area and coupled through a tube with a constant flow resistance. In this case, the first tank is heated by an electric heater. For our purpose the liquid contained in the tanks is water, therefore density ρ is 1 and the specific heat c_p is 80.

The inputs to the process are: temperature of the input flow $T_e(t)$ indicated by u_3, input flow rate $F_e(t)$ indicated by u_1 and input heat $Q_e(t)$ denoted by u_2. The process outputs are: the level of water in tank 2 $L_2(t)$ indicated by $y_1(t)$ and water temperature in tank 2 $T_s(t)$ indicated by $y_2(t)$. It is assumed that both tanks are well isolated to reduce heat leaks. It is also assumed that the temperature of the input flow is not controllable and it is used as a disturbance input during simulations. Therefore, this process consist of two inputs and two outputs. The coupled tanks process is illustrated in Figure 4.

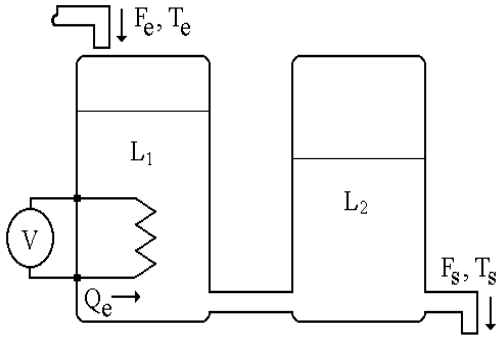

Figure 4, Coupled tanks process

The analytical model of the coupled tanks process is described by a set of four differential equations for the states and two expressions for the outputs of the process [10]. These equations are illustrated below.

$$
\begin{aligned}
\dot{x}_1 &= -\frac{gk_1}{a_1}x_1 + \frac{gk_1}{a_2}x_2 + u_1 \\
\dot{x}_2 &= \frac{gk_1}{a_1}x_1 - \frac{g(k_1+k_2)}{a_2} \\
\dot{x}_3 &= \frac{gk_1x_3}{a_1x_1}x_2 - \frac{gk_1}{a_1}x_3 - c_pu_1u_2 - u_3 \\
\dot{x}_4 &= -\frac{gk_1x_3}{a_2x_1}x_2 + \frac{gk_1}{a_1}x_3 - \frac{gk_2}{a_2}x_4 \\
y_1 &= \frac{x_2}{a_2\rho} \\
y_2 &= \frac{x_4}{c_px_2}
\end{aligned}
\qquad (11)
$$

Here the parameters a_1 and a_2 are the corresponding areas of transversal section of tanks 1 and 2 respectively, k_1 and k_2 are the inverse of the flow resistances between the two tanks an at the flow output in tank 2 respectively and g is the acceleration due to gravity. The process outputs are y_1 and y_2 which represent the level in tank 2 and water temperature in tank 2 respectively.

For our simulation purpose the parameters are set to a_1=5, a_2=10, k_1=1 y k_2=1. Therefore, the parameters of the set of differential equations are

$$
\begin{aligned}
\frac{gk_1}{a_1} &= 1.9620 \\
\frac{gk_1}{a_2} &= \frac{gk_2}{a_2} = 0.9810 \\
\frac{g(k_1+k_2)}{a_2} &= 1.9620
\end{aligned}
\qquad (12)
$$

To simulate the response of the process a conventional numeric integration algorithm is used. Since the control algorithm is discrete, the integration algorithm is arranged to compute the states on each desired sample time. The results of the integration process are fed to the neural network and to the controller as illustrated in Figures 2 and 3.

4.2 Control of the process

The control algorithm in Equation (10) requires few adjustments in order to be applied to the coupled tanks. First, it is necessary to define matrices Q and R. In this case both matrices are of size 2x2 and the elements are

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1.7 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.7 \end{bmatrix} \quad (13)$$

The elements in the two matrices were chosen as a result of a sequence of simulation tests. The objective of these tests was to produced an acceptable response of the controlled process. A control gain of $\eta_c = 0.45$ was selected.

Like in the case of the weighting matrices, the value of the control gain was selected for the purpose of illustrating an acceptable control performance, not an optimal control performance. The desired outputs of the process were defined as T_s = 4.35 (indicated by y_1) and F_s = 6.35 (indicated by y_2) for the temperature of water in tank 2 and water level in tank 2 respectively. Figure 5 illustrates the response of the controlled process

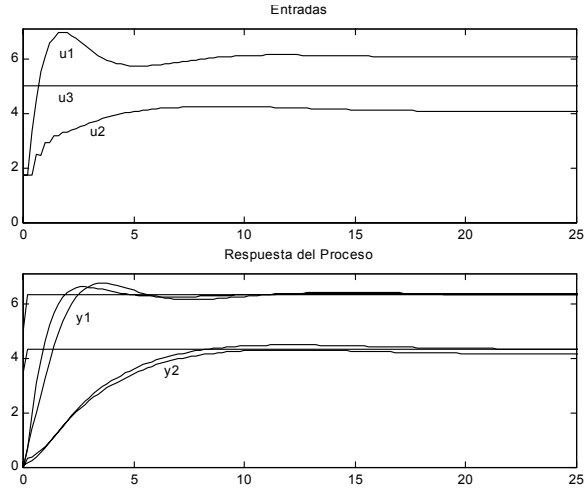

Figure 5, Response of the controlled process

It can be seen that the response of the process reaches the desired outputs with a small overshoot. The inputs u_1 and u_2 are the process inputs and the input u_3 is an uncontrollable input used for the simulation of disturbances. In Figure 5 the controlled process is unaffected by perturbations and therefore, the input u_3 remains constant.

4.3 Response to input disturbances

Some simulated disturbances were applied to the controlled process to study its performance. The simulated disturbances consisted of the introduction of an abrupt change of the temperature of the input u_3 of the process. It was assumed that the change in the temperature would produce a noticeable change of the output of the coupled tanks process. In this experiment the matrices in Equation (13) take the same value as in the experiment described in section 4.1. The disturbance is an abrupt change from $T_e(t)$=5 to $T_e(t)$=1.75. Figure 6 illustrates the simulated response of the controlled process

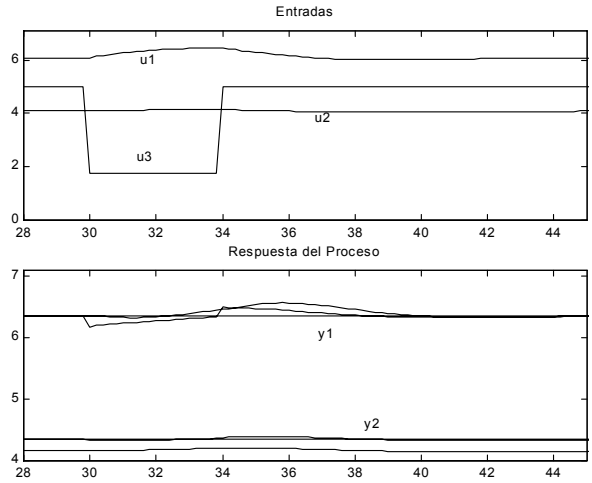

Figure 6, Response to a disturbance

It can be seen that the disturbance produces a noticeable change in the outputs of the process. However, the control algorithm produces appropriate control vectors to compensate the effect of the disturbance. It is evident that the effect of the disturbance is larger at the output y_1.

5 Stability

The stability of the control algorithm depends on the appropriate selection of the elements of Q and R and the chosen control gain η_c. In order to illustrate the stability of the process two simulated cases were studied. First, the effect of the parameters of Q and R is depicted on the first simulated results.

In this case both matrices were chosen to drive the process to an unstable operation. Second, the control gain η_c was adjusted to a large value such that the response of the process reached an unstable regime.

5.1 Unstable response due to Q and R

The matrices Q and R weight the effect of the inputs and outputs of the process in the cost function. Adjusting these matrices allows to achieve an acceptable response of the controlled process.

Nevertheless, setting those matrices to incorrect values may cause an unstable response of the process. To illustrate this, both matrices have been adjusted to drive the process to an unstable condition. The resulting adjusted matrices are

$$Q = \begin{bmatrix} 2.7 & 0 \\ 0 & 3.9 \end{bmatrix}$$

$$R = \begin{bmatrix} 1.8 & 0 \\ 0 & 1.8 \end{bmatrix} \tag{14}$$

In this case the control gain is kept constant at $\eta_c = 0.45$. The simulation results are illustrated in Figure 7.

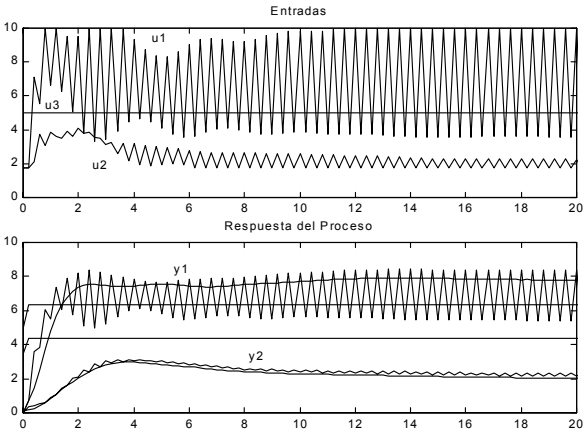

Figure 7, Unstable response due to poor Q and R

It can be seen that the response of the controlled process is unstable. In order to chose appropriate values for Q and R the control engineer requires to carry out a previous sequence of experiments.

5.2 Unstable response due to η_c

A selection of inappropriate control gain may lead to an unstable response or to an slow controlled response. A sequence of experiments will be required in order to select a control gain to produce a desired controlled response.

In this case, the purpose of the simulated experiments is to demonstrate the effect of a large control gain over the controlled process stability. For this purpose the weighting matrices in Equation (13) are kept unchanged. The control gain is set to $\eta_c = 0.6$ to drive the process to an unstable response. The result of the simulation is shown in Figure 8.

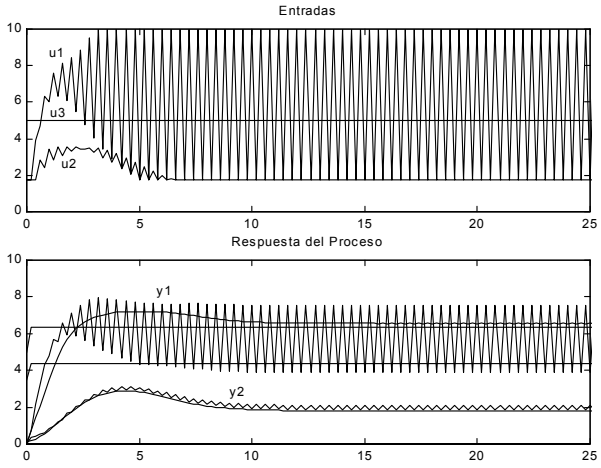

Figure 8, Unstable response due to poor η_c

Again, the response of the controlled process is oscillatory but this time it is only due to a poor choice of the control gain.

6 Conclusion

The control algorithm described in the previous sections have been tested under simulated conditions. It has been demonstrated that the algorithms perform satisfactorily under simulated operating conditions. The control algorithm is easy to implement in computer software and requires few computational resources. Further trials would be required to validate the algorithm under experimental conditions.

On the other hand, the stability of the control algorithm was simply demonstrated using simulations. It is however necessary to develop an stability analysis for this algorithm. The second method of Lyapunov may be used for this purpose. It is necessary to remember that it is not possible to guarantee the stability of the control algorithm. The gradient descent rule may get stuck in local minima. As a result the required control vector may not be computed accurately. Finally, the present control algorithm will be combined with a fault detection and diagnosis algorithm to produce a fault tolerant control.

7 Acknowledgments

The author wishes to express his gratitude to CONACYT for the support that made this work possible, through grant I28288-A. Also, support from Departamento de Investigación en Física de la Universidad de Sonora is gratefully acknowledged.

References:

[1] Stengel R.F., Optimal Control and Estimation, *Dover*, 1994.

[2] Maciejowski J.M., Multivariable Feedback Design, *Addison Wesley,* 1991.

[3] Lakhmi C. Jain and Clarence W. Silva, Intelligent Adaptive Control: Industrial Applications, *CRC Press,* 1999.

[4] Alberto A. Behar, Self-Tuning Controller, *5th IFAC Workshop on Algorithms and Architectures for Real-Time Control,* 15-17 April, Cancun México, 1998.

[5] Noriega J.R. and H. Wang, A Direct Adaptive Neural Network Control for Unknown Nonlinear Systems and Its Application, *IEEE Transactions on Neural Networks,* Vol. 9, pp. 27-34, January, 1998.

[6] Shiffmann W. H. and H. W. Geffers, Adaptive Control of Dynamic Systems by Backpropagation Networks, *Neural Networks,* No. 6, Vol. 4, pp. 517-524, 1993.

[7] Narendra K. and K. Parthasarathy, Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks,* No. 1, Vol. 1, pp. 4-27, March, 1990.

[8] Widrow B. and M. A. Lehr, 30 Years of Adaptive Neural Networks: Perceptron, Madaline and Backpropagation, *Proceedings of the IEEE,* No. 78, Vol. 9, pp. 1415-1441, September, 1990.

[9] Brown M. and C. Harris, Neurofuzzy Adaptive Modelling and Control, *Prentice Hall,* 1994.

[10] Gawthrop Peter and Smith Lorcan, Metamodelling for Bond Graphs and Dynamic Systems, *Prentice Hall,* 1996.

[11] Tan Y. and Keyser R.D., Neural-Network-based adaptive predictive control, *Advances in MBPC,* 1993, pp. 77-88.

[12] Clarke D.W. Mohtadi C. and Tuffs P.S., Generalized predictive control: Parts I and II, the basic algorithm, *Automatica,* vol. 23, no. 2, pp. 137-160, 1987.

[13] Polycarpou M.M. and Ioannou P.A., Modelling, identification and stable adaptive control of continuous time nonlinear dynamical systems using neural networks, *Proc. Amer. Contr. Conf.,* 1992, pp. 36-40.

[14] Chen F.C. and Liu C.C., Adaptively controlling nonlinear continuous time systems using multi-layer neural networks, *IEEE Transactions on Automatic Control,* vol. 39, pp. 1306-1310, 1994.

[15] Isidori A. and Byrnes C.I., Output regulations of nonlinear systems, *IEEE Transactions on Automatic Control,* vol. 35, pp. 131-140, 1990.

[16] Wang H., Liu G.P., Harris C.J. and Brown M., Advanced Adaptive Control, Oxford: *Pergamon,* 1995.