Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99) Haifa, Israel - June 28-30, 1999

ADVANCED ADAPTIVE CONTROL FOR COMPLEX NONLINEAR PROCESSES

Nicolae Constantin , Ion Dumitrache

Department of Automatic Control and Systems Engineering
University Politehnica of Bucharest
Spl. Independentei Nr.313 , 77206, Bucharest , Romania.
Tel.: 40-13113241, Fax : 40-13113242

Abstract: Adaptive techniques based on neural-networks are investigated in an application to the identification and subsequent on-line control of a process exhibiting nonlinearities and typical disturbances. The method proposed consists of a novel identification technique based on extended memory adaptation (EMA) and an efficient implementation of the predictive control based on a nonlinear programming method. A forced circulation evaporator was chosen as a realistic nonlinear case study for the techniques discussed in the paper.

Keywords: adaptive control, nonlinear models, on-line control, neural networks.

1. INTRODUCTION

In recent years, there has been much research on the use of neural networks in control, especially in problems where nonlinearity dominates. With an increasing focus on performance and profit, conventional control technology cannot cope well with changing plant conditions and dynamics. A successful control system should be able to respond to changing demands and standards, and should be trying to satisfy many requirements at once- not only keeping the plant output within some limits but also doing so inexpensively, quick and efficiently. Model-based techniques utilise some parameterized model of the plant from which a suitable control-law can then be designed (Wellstead, 1991). It is common that both model and controller are linear, with the parameters of each adapted over time to account for plant nonlinearities. This approach forms the basis of many of the popular adaptive control techniques, such as pole-placement and minimum-variance self-tuning control (Wellstead, 1991) and model predictive control techniques (Clarke, 1987; Richalet, 1992). In (Narendra, 1990) it was proposed the enhancement of model-based control strategies, such as indirect adaptive control, by replacing the linear models with nonlinear representations, based on neural networks (NN). Similarly it has been shown that predictive control techniques could be improved by including a nonlinear model of the plant (Hernandez, 1990). In particular the dynamic matrix control (DMC) was extended by using a nonlinear predictive model of the process, based on feedforward neural networks. Neural networks offer a flexible structure that can map arbitrary nonlinear functions, making them ideally suited for the modelling and control of complex, nonlinear systems (Hunt *et al.*, 1992). They are particularly appropriate for multivariable applications, where they can readily characterise the interactions between different inputs and outputs. The training of these networks can be performed off-line and subsequently be augmented as part of an on-line adaptive scheme. A further benefit is that the neural architecture is inherently parallel, distributed and has the potential for real time implementation. The goal of this paper is to show that, provided with appropriate accelerated adaptation techniques and control structures, neural networks can be used to adaptively control a wide range of nonlinear processes at a useful level of performance. The method proposed consists of a novel identification technique based on extended memory adaptation (EMA) and an efficient implementation of the predictive control based on a nonlinear programming algorithm.

The organisation of the paper is as follows. In Section 2 the nonlinear system identification structures with NN and the novel adaptation technique EMA is proposed and analysed. In Section 3 an efficient implementation of predictive control using a feedforward neural network as the plant's nonlinear model is presented. In Section 4 the predictive control algorithm proposed is combined with the EMA method to form an adaptive neural predictive control (ANPC) structure. Section 5 shows the simulation results of adaptive control for a multivariable nonlinear process. Conclusions are presented in Section 6.

2. NEURAL NONLINEAR MODELING

In the areas of control and systems engineering, the feedforward networks, such as the multi-layer perceptron (MLP), radial basis function (RBF) networks, and the cerebellar model articulation controller (CMAC) network seem to have attracted most attention (Hunt, *et al.*, 1995). This is undoubtely due to their ability to represent arbitrary, multidimensional, nonlinear mappings, a powerful feature that can be readily incorporated within nonlinear modeling and control structures (Narendra, 1990). It has been proved that MLP can represent any nonlinear mapping between input and output with arbitrary accuracy (Hornik, *et al.*, 1989). A general model structure suitable for representing the dynamics of a wide range of nonlinear systems is the nonlinear auto-regressive with exogeneous signals (NARX) model defined by:

$$y(k) = f\,(y(k-1),\dots,y(k-n_y),\ u(k-d-1),\dots,u(k-d-n_u)) + e(k) \qquad (1)$$

where y(k) and u(k) are the sampled process output,

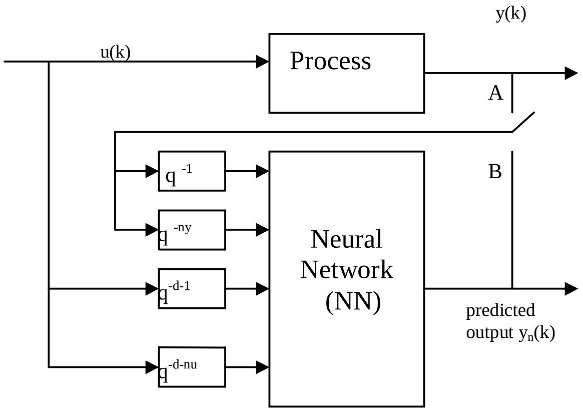

Fig. 1. Configuring a NN as an NARX model. Switch in position A gives predictor operation, position B gives model operation.

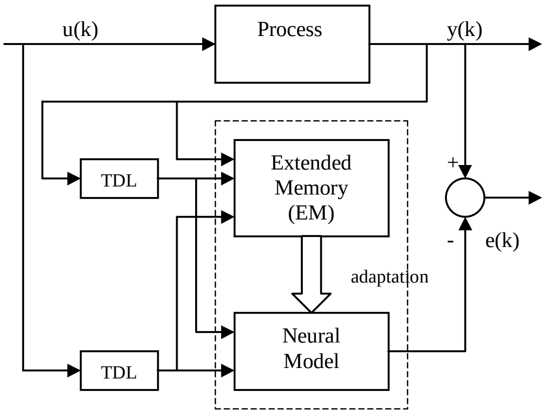

Fig. 2. Integration of extended-memory into the identification structure.

d represents the process dead-time and f(.) is an unknown nonlinear function to be identified. It is important to note a distinction between two different modes of operation of a trained neural network NARX model. In eq. (1), the network requires both input and output data from the process as inputs, and it predicts the process output a single time step into the future, $y_n(k)$. In this predictor structure, a trained network cannot, therefore, be used independently from the plant, or to provide long range predictions. This two tasks can be achived by operating a trained network in the model structure:

$$y_n(k) = f(y_n(k-1),\dots,y_n(k-n_y),\ u(k-d-1),\dots,u(k-d-n_u)) \tag{2}$$

where $y_n(k)$ is the model predicted output of the network. In eq.(2), the network is supplied with input data, u(k), and the network outputs, $y_n(k)$, are delayed and fed back to the network inputs to project predictions of the process output further into the future. The difference between these two modes of network operation are illustrated in Fig. 1.

2.1. Extended-Memory Adaptation Technique

In off-line training it is difficult to assure the conditions for a good generalisation of the neural networks hence the on-line training is always necessary in control applications. In fact, the training should ideally occur exclusively on-line, with the neural networks learning at high speed from any initial set of weights. In order to make neural control a viable alternative to industrial control of processes, there is a pressing necessity for efficient on-line training algorithms. The purpose of this section is to present a new technique, denoted Extended-Memory Adaptation (EMA), for improving the convergence characteristics of adaptive identification. To convert the learning procedure to an adaptive algorithm the single pattern generated by the tapped delay lines (TDLs) from the process sample at each time-step is applied to the NN to obtain a small increment of adaptation at each time-step as shown in Fig. 2.

To make the procedure adaptive over time the context of application of learning rule has been changed from batch to single pattern presentation environment. A Multiple-Input Multiple-Output (MIMO) model suitable for identification can be expressed in terms of $\theta(k)$, the set of unknown parameters, past values of the inputs and outputs, and modelling errors e(k) :

$$y(k) = f(\theta(k), y(k-1),\dots,y(k-n_y),\quad u(k-1),\dots,u(k-n_u)) + e(k) \tag{3}$$

At time step k each weight w_{ji} , connecting the input of node i on each layer, to the output of node j on the preceding layer, is updated using :

$$\begin{aligned} w_{ji}(0) &= \gamma_{ji} \\ w_{ji}(k) &= w_{ji}(k-1) + \alpha\, e_i(k) + \beta(w_{ji}(k-1) - w_{ji}(k-2)) \end{aligned} \tag{4}$$

where $\alpha \in R^+$ is the learning rate , $0 < \beta < 1$ is the momentum factor, $e_i(k)$ is the backpropagated error value corresponding to the input to the sigmoid function calculated for node i during the presentation of the pattern $\theta(k)$ derived from process time k. The addition of a momentum term in the learning rule can greatly accelerate convergence. The random variable γ_{ji} is independently selected for each weight from a distribution with zero mean and small variance.

This recursive equation is applied to each weight at each time-step k of the plant operation. A distinction should be made between iteration q and time step k. The updated weight w_{ji} (q,k) is calculated using the extended recursive equation set below:

$$\begin{aligned} w_{ji}(0,0) &= \gamma_{ji} \\ w_{ji}(0,0) &= w_{ji}(1,k-1) \\ w_{ji}(q,k) &= w_{ji}(q-1,k) + \alpha\, e_i(k) + \beta\,(w_{ji}(q-1,k) - w_{ji}(q-2,k)) \end{aligned} \tag{5}$$

The method proposed retains some process patterns that can represent an approximation to the nonlinear process dynamics. The extended memory (EM) accepts a new pattern $\theta(k)$ from the process at each time step k and discards the oldest pattern $\theta(k-n_p)$. The EM elements $\theta(k)$, $\theta(k-1)$, ... , $\theta(k-n_p+1)$ are each used in n_c cycles to update the weights at each time-step. The recursive equation set can be written as follows:

$$\begin{aligned} w_{ji}(0,0) &= \gamma_{ji} \\ w_{ji}(0,0) &= w_{ji}(1,k-1) \\ w_{ji}(q,k) &= w_{ji}(q-1,k) + \alpha\, e_i(p) + \beta\,(w_{ji}(q-1,k) - w_{ji}(q-2,k)) \end{aligned} \tag{6}$$

where: l is a cycle counter for n_c cycles of the pattern memory which are performed at time step k , p indicated the time step origin of the pattern selected from the EMA is chosen in random order from its index range (k - n_p +1 , k). At each selection of p and execution of the recursion , the iteration count q is incremented, up to $n_c n_p$. By applying the supervised learning rule using all the patterns of the EM for learning at each time step, the rule drives the weight vector of the network towards one that minimises the error corresponding to the approximate mapping represented in the EM patterns. The random order of presentation ensures that the sampled pattern order does not adversely affect the learning rule dynamics. The EM adaptation algorithm is summarised as follows:

1. sample inputs and outputs of the process.
2. shift and update the TDL elements.
3. if TDL are not full, restart at step 1 at next time step.
4. construct a training pattern from elements of TDLs and the latest process output sample.
5. execute neural network to calculate one-step ahead prediction.
6. insert the new pattern into the EM.
7. if the EM size > n_p , remove the oldest pattern.
8. train network for n_c cycles of up to n_p EM patterns presentation (randomised order).

2.2. EM parameters

The learning rule is repeated n_p x n_c times at each time-step. The parameter n_p determines the length of moving average effect on the patterns. It is set as a balance between speed of adaptation and accuracy of model. It is analogous to the forgetting factor commonly used in linear adaptive control theory (Mills *et al.*, 1996). Using various situation typical values of n_c and n_p for various nominal conditions and objectives have been compiled in table 1.

Table 1 Typical values for parameters n_c and n_p

condition	n_p	n_c
base case	50	10
fast adaptation	20	15
low computation	20	10
complex mapping	80	10
noisy samples	90	15

In the case study simulations we demonstrate greatly enhanced identification performance in adaptive control structures and also the possibility of starting identification and control at the same time with no a priori knowledge of the plant.

3. NONLINEAR PREDICTIVE CONTROL

The complexity of developing a phenomenological model for the process and the nonlinearities of its dynamics, make very attractive the use of an artificial neural network for the identification and predictions of the plant outputs. Fixed NN models, identified off-line have been used with various model-based control structures (Saint Donat *et al.*, 1991).

Prediction using neural networks

The prediction algorithm uses the output of the plant 's model to predict the plant's dynamics to an arbitrary input from the current time k to some future time k+n. This is accomplished by time shifting equations for $y_n(k)$ and $net_j(k)$ by n resulting in

$$y_n(k+n) = \sum_{j=1}^{nh} w_j f_j\Big((net_j(k+n)\Big) + b \tag{7}$$

and

$$\begin{aligned} net_j(k+n) = {} & \sum_{i=0}^{nu} w_{j,i+1} \begin{cases} u(k+n-i), & n-N_u < i \\ u(n+N_u), & n-N_u \geq i \end{cases} + \\ & + \sum_{i=1}^{\min\{n,ny\}} w_{j,nu+i+1} y_n(k+n-i) + \sum_{i=n+1}^{ny} w_{j,nu+i+1} y(k+n-i) + b_j \end{aligned} \tag{8}$$

where $f_j(.)$ is the output function for the j^{th} node of the hidden layer, $net_j(n)$ is the activation level of the j^{th} node's output function, nh is the number of hidden nodes in the hidden layer, w_j is the weight connecting the j^{th} hidden node to the output node ,$w_{j,i}$ is the weight connecting the i^{th} input node to the j^{th} hidden node, b_j the bias on the j^{th} hidden node, b the bias on the output node. The first summation of (8) breaks the input into two parts represented by the conditional.

The condition where $n-N_u < i$ handles the previous future values of u up to $u(k+N_u-1)$. The condition where $n-N_u > i$ sets the inputs from $u(k+N_u)$ to $u(k+n)$ equal to $u(k+N_u)$. The next summations of (8) handles the recursive part of prediction. This feeds back the network output $y_n(k)$ for n or n_y times, which ever is smaller. The last summation of (8) handles the previous values of y. Using quadratic cost function and the predicted model it is possible to calculate the optimal control strategy for an nonlinear model predicted by using NN. The cost function in predictive control is chosen as :

$$J=\sum_{j=N_1}^{N_2}\left[r(k+j)-y_n(k+j)\right]^2+\sum_{j=1}^{N_u}\lambda(j)\left[\Delta u(k+j)\right]^2 \tag{9}$$

where r is the required process output, y_n is the NN model output, u is the process input, N_1 and N_2 define the prediction horizon, N_u the control horizon and λ is a sequence of control weighting factors. A suitable choice for N_1 is to make it equal to the process delay between input and output, d+1 in eq.(1). N_2 is then set to define the prediction horizon beyond this point and it represent the number of time-steps in the future for which the process response is recursively predicted as a result of the proposed control action sequence u(k) , ..., u $(k+N_u-1)$ executed over the control horizon N_u . Control actions after the control horizon are held constant equal to the value $u(k+N_u-1)$. With Newton-Raphson (NR) method, J is minimized iteratively to determine the best

$$\underline{u}(n)=\left[u(k+1)\; u(k+2)\ldots u(k+N_u)\right]^T \tag{10}$$

The improved convergence rate of NR is computationally costly, but is justified by the high convergence rate. The NR update rule for $\underline{u}(n+1)$ is:

$$\underline{u}(n+1)=\underline{u}(n)-\left(\frac{\partial^2 J}{\partial \underline{u}^2}(n)\right)^{-1}\frac{\partial J}{\partial \underline{u}}(n) \tag{11}$$

In order to avoid the computation of the inverse of the Hessian matrix, equation (11) is rewritten in the form of a system of linear equations: $\frac{\partial^2 J}{\partial \underline{u}^2}(n)\left(\underline{u}(n+1)-\underline{u}(n)\right)=\frac{\partial J}{\partial \underline{u}}(n)$ which is solved for $x=\underline{u}(n+1)-\underline{u}(n)$ by using the LU decomposition. When solving for x, calculation of each element of the Jacobian and Hessian is needed for each NR iteration.

4. ADAPTIVE NEURAL PREDICTIVE CONTROL

A schematic of the neural-network predictive control (NNPC) structure is illustrated in Fig. 3. This structure has a number of attributes that make it an appealing strategy to adopt in comparison to other neural-network control schemes. The NNPC does not require training of networks in addition to a forward mode, as required with internal model control (Hunt *et al*., 1992). A major advantage of the NNPC is that the process input is optimally computed to take into account of future predictions of the process response up to a defined horizon. This enable the scheme to anticipate future process outputs, which achieves smooth transient responses to set-point changes and the ability to take early corrective action against disturbances. The adaptive scheme (Fig. 3) is obtained by combining with the EM adaptation presented in Section 2. Adaptive Neural Predictive Control (ANPC) algorithm is as follows :

1. sample inputs and outputs of the process.
2. shift and update the TDL elements.
3. if TDL are not full , restart at step 1 at next time step.
4. use the NN model to predict the estimate y_n.
5. construct a training pattern from elements of TDLs and the latest output sample.
6. insert the new pattern in EM (if the EM size > n_p remove the oldest pattern).
7. train network for n_c cycles of up to n_p EM patterns (randomised order).

8. use TDL values and proposed control actions to predict future process trajectory.
9. evaluate performance objective function.
10. improve proposed control actions and go to step 10 , until optimised.
11. output the control actions calculated for time-step k to the process.

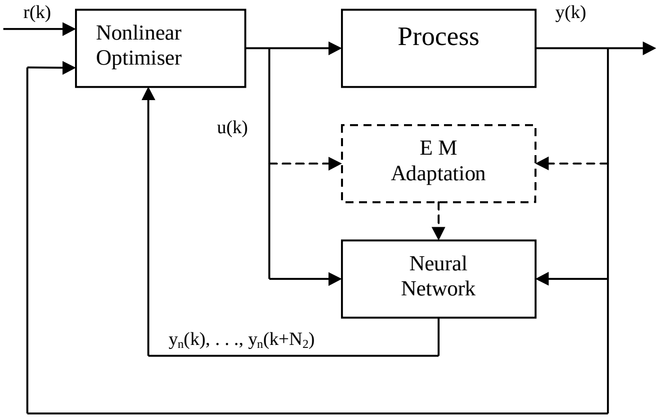

Fig. 3. Neural network (adaptive) predictive control scheme.

The adaptation of control strategy to the parameters changes of the model it is also considered. In this case the nonlinear model is constructed and a control strategy is adapted minimising a quadratic cost function in predictive control sense.

5. CASE STUDY. SIMULATION RESULTS.

Neural control techniques have already been introduced in many industries, in particular to the chemical and biochemical processes (Gomm *et al.*, 1997; Mills *et al.*, 1996). In this paper it is considered a complex process simulation of a forced circulation evaporator system. The process diagram, variables, their description, normal operation states with measurements units and also the simulation equations are presented in (Newley and Lee, 1989). In simulations was used Netlib library with description from (Mills *et al.*, 1996). There are three control actions P10(u1) , F20(u2) , F2(u3) , three major states X2(y1) , P2(y2) , L2(y3) and five possible disturbance sources F1 , F3 , X1 , T1 , and T20. The resulting process simulation is dynamically nonlinear, unstable, and multivariable with strong interactions. An input-output neural model with $n_y=2$ and $n_u=2$ was selected. The final network topology was selected to be 12-15-3 . The EM parameters were set to $n_c=20$ si $n_p=50$.

A sequence of test cases using the proposed method were conducted to verify multivariable performance. Each of these cases consisted of three distinct components:

- initial start-up adaptation and stabilization ;
- setpoint following ;
- disturbance rejection ;

The first stage of the test reflects the mode of start-up. The initial states and setpoints are at the standard operation values, each corresponding to zero after the normalisation. After a period was allowed for initial adaptation and stabilisation of the process, a sequence of alternating setpoint step (for X2) set to 1 , -1 , 1 , -1 was introduced. Finally it was introduced a large, unmeasured disturbance to the evaporator. The feed flow to the evaporator, F1, was reduced 25% from 10.0 to 7.5 Kg/min. The performance of each of the cases and their individual components were compared using the Integrated Time-weighted Absolute Error (ITAE) measure. In this experiment the control rate-of-change weight was set to 0.5 ($\lambda = 0.5$). The plot of the three outputs is shown in Fig.4 and the control actions, for $N_2 = 5$ and $N_u = 3$ are presented in Fig. 5.

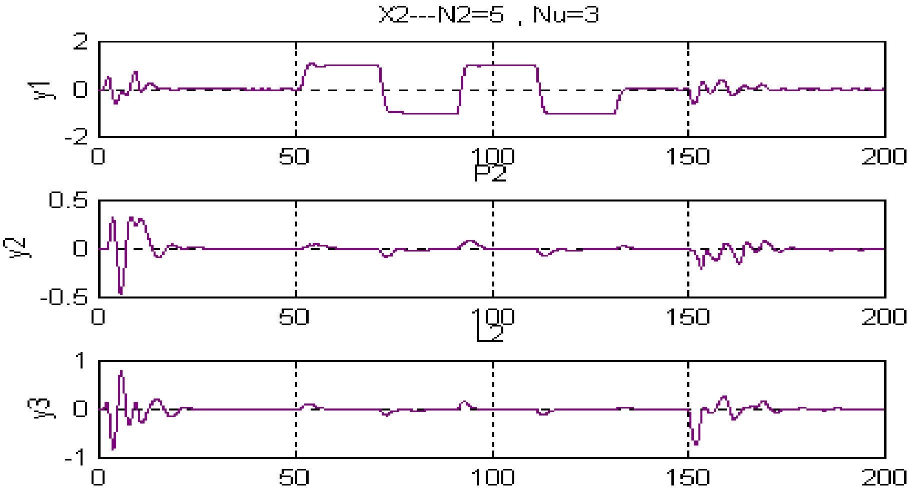

Fig. 4. Process response for ANPC with N_u=3 , N_2=5 and λ = 0.5.

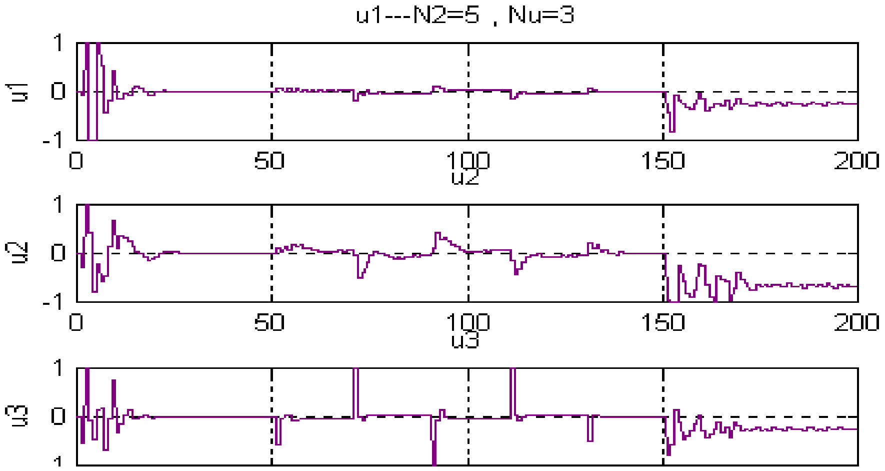

Fig. 5. Control actions for ANPC with N_u=3 , N_2=5 and λ = 0.5.

The process was quickly stabilized and the measurements had varied at most by only 5, 6 and 20% respectively from their operating values. The 20 % variation of y3 is associated with the separator level, the unstable state of the process. The closed-loop responses indicate very good control of all outputs.

The initial stabilization tends to have fewer, lower amplitude, slower cycles in comparison with the case when the rate-of-change weight was set to zero. Several others experiments were realised to investigate the effects of different values of the design variables within the control structure (Constantin, N.,1997).

The results obtained were in accord with those expected from using a predictive control methodology. It is seen that with no weighting of the control input good set-point tracking is achieved; however, the control input exhibits large fluctuations and would cause undue actuator wear in practice. For larger values of λ , movement of the manipulated variable is dampened at the expense of poorer set-point tracking. It was observed that increasing the prediction horizon had a number of desirable effects. Good set-point tracking was maintained , whilst the control effort was considerably reduced. These results demonstrate that improvements can be achieved with neural network in practice. The ANPC method is better than the other approaches due to fast adaptation and ability to address process nonlinearity. For time-varying processes, adaptation is the only means by which the performance of the controller can be maintained.

6. CONCLUSIONS

This paper has discussed issues concerning the application of neural networks to the modelling and control of nonlinear processes. A novel identification method is proposed to greatly accelerate the real-time learning of nonlinear input-output mappings by feed-forward neural networks. This identification method is combined with a multistep predictive control to provide an adaptive neural predictive control (ANPC) strategy. The method is superior due to fast adaptation and ability to address process nonlinearities. It is a viable adaptive control approach which can achieve high performance when applied to nonlinear processes with high complexity.

REFERENCES

Chen, S. and S.A. Billings (1989). Representation of nonlinear systems: The NARMAX model. *Int.J.Contr.*, **49**(3), 1013-1032.

Clarke, D.W., C.Mohtadi, and P.S.Tuffs (1987). Generalized predictive control-Part1: The basic algorithm. *Automatica*, **23**, 137-148.

Constantin, N. (1997). Robust Adaptive Selftuning Control. PhD. Thesis, University Politehnica, Bucharest.

Constantin, N. and I. Dumitrache (1997). Multimodel Adaptive Control. *Proc. Inter. Conf. CSCS11*, **1**, pp. 259-264, Bucharest.

Gomm, J.B., J.T. Evans and D. Williams (1997). Development and performance of a neural-network predictive controller. *Control Eng. Practice,* vol. **5**, No.1, pp. 49-59.

Hornik, K., M. Stinchombe and H. White (1989). Multilayer feedforward networks are universal approximators. *Neural Networks,* **2**, 359-366.

Hunt, K.J., G.W. Irwin and K. Warwick (1995). *Neural Network Engineering in Dynamic Control Systems*. London: Springer-Verlag.

Hunt, K.J., D. Sbarbaro, R. Zbikowski and P.J. Gawthrop (1992). Neural network for control systems. *Automatica*, **28**(6), 1083-1112.

Mills, P.M., A.Y. Zomaya and M.O. Tade (1996). *Neuro-Adaptive Process Control - A Practical Approach*. John Wiley & Sons.

Narendra, K.S. and K. Parthasarathy (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. on Neural Networks,* **1**(1), 4-27.

Newell, R. and P. Lee (1989). *Applied Process Control : A Case Study*. Englewood Cliffs, New Jersey: Prentice Hall.

Richalet, J. (1992). Observations on model-based predictive control. *Control Eng. Practice,* Aug., 38-41.

Saint Donat, J., N. Bhat and T.J. McAvoy (1991). Neural net based model predictive control. *Int. J. Control,* **54**(6), 1453-1468.

Wellstead, P.E. and M.B. Zarrop (1991*). Self-tuning Systems*. New York:Willey.