INTERNAL MODEL CONTROL USING RECURRENT NEURAL NETWORKS FOR NONLINEAR DYNAMIC SYSTEMS

Yan Li, David Powers and Peng Wen

School of Informatics and Engineering
The Flinders University of South Australia
GPO Box 2100, Adelaide, SA 5001, Australia

Abstract: This paper presents a control method using Recurrent Neural Networks (RNNs) in an Internal Model Control (IMC) framwork, and demonstrates their effectiveness of modelling and control for nonlinear dynamic systems. Unlike existing neural IMC design methods, the proposed control scheme consists of two stages. The first stage is that using two same structure of RNNs through iterative learning techniques to obtain a desired control signal to the unknown nonlinear systems. Then a RNN is used to generate the desired control signal within IMC structure. The algorithm for the RNNs is a real time iterative learning algorithm based on two-dimensional (2D) system theory. The simulation results demonstrate the proposed control method can drive unknown systems to follow the desired trajectories very well.

Keywords: Nonlinear Control Systems, Neural Networks, Nonlinear Models,Two-dimensional systems, Multi-input/multi-output

1. INTRODUCTION

Much success has been achieved in the use of RNNs for identification and control in recent years. Many publications (Narendra and Parthasarthy, 1991; Chow and Fang, 1999; Wen *et al.*, 1996), have proved that RNNs are more powerful for nonlinear dynamic system since the architectures of RNNs themselves are presented by nonlinear dynamic system. Many approaches to train RNNs to handle time-varying input/output have been suggested or investigated by control researchers. Among those, Chow and Fang (Chow and Fang, 1999) developed a real-time iterative learning algorithm which derived by means of two-dimensional(2D) system. The algorithm is different from conventional algorithms that employ the steepest optimisation to minimise a cost function. An RNN using the algorithm can approximate any trajectory with time-varying weights. With their ability to deal effectively with time-varying input/output, recurrent neural networks are attractive for modelling and adaptive control design.

The iterative learning algorithm is employed to train our RNNs in this paper, and internal model control (IMC) is used to provide a general framework for nonlinear system control. IMC is significant (Li *et al.*, 1996) because the stability and robustness properties of the structure can be analysed and manipulated in a transparent manner, especially for non-linear systems. The proposed control scheme consists of two stages. The first stage uses two identical RNN structures through iterative learning techniques to obtain a "desired" control signal to the unknown nonlinear systems. After the desired control signal is obtained, One of the RNNs is used to generate the desired control signal within an internal model control structure. The simulation results in this paper demonstrate the control method can drive unknown systems to follow the desired trajectories very well.

2. THE ALGORITHM

The architecture of the RNNs used in this paper is fully connected. Nodes in the RNNs are generally classified into three categories (instead of layers): input, output and hidden nodes. In this paper, we use the term "processing nodes" to represent all the output and hidden nodes. There are two sets of synaptic connections in RNNs. The first set of connection links between the input and the processing nodes. Their weights constitute the inter-weights matrix $\mathbf{W_2} = \{w_{ij}\}$. The weight $w_{ij}(t)$ $\forall i \in \mathbf{U}$ and $j \in \mathbf{I}$ (where $\mathbf{U}$ and $\mathbf{I}$ are the set of processing and input nodes, respectively) denotes the strength of the connection from the j^{th} input node to the i^{th} processing node, at time t. The second set of connections form the feedback paths. Therefore, each processing node is connected to all other processing nodes, including itself. Their weights constitute the intra-weight matrix $\mathbf{W_1} = w^*_{ij}$. Similarly, $w^*_{ij}(t)$ denotes the strength of the connection from the j^{th} processing node to the i^{th} processing node ($\forall i, j \in \mathbf{U}$), at time t. Figure 1 shows the topology of RNNs.

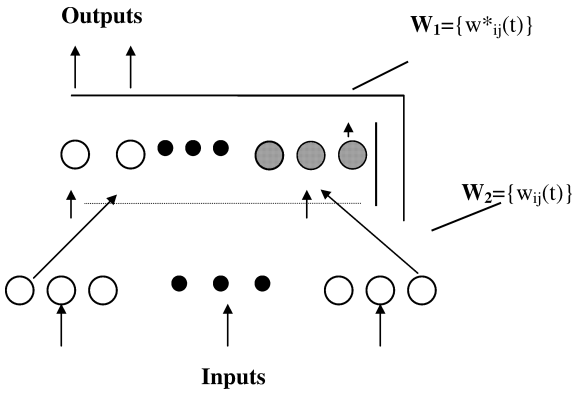

Fig. 1. The topology of RNNs

Let $\{y(t)\}$ denote the outputs of the processing nodes and $\{u(t)\}$ denote the external inputs. Then, their state-space nonlinear dynamics of the RNN is presented in the following matrix form:

$$\mathbf{y(t+1)} = f[\mathbf{W_1(t)y(t)} + \mathbf{W_2(t)u(t)}] \quad (1)$$

with initial value $\mathbf{y(0)} = y_0$.
where

$\mathbf{y(t)} = (y_1(t), y_2(t), ..., y_n(t))^T \in \mathbf{R}^n$,

$\mathbf{u(t)} = (u_1(t), u_2(t), ..., u_m(t))^T \in \mathbf{R}^m$,

$\mathbf{W_1(t)} \in \mathbf{R}^{nXn}$,

$\mathbf{W_2(t)} \in \mathbf{R}^{nXn}$

$\mathbf{f}(.)$ is a vector of a nonlinear activation function,

$\mathbf{f}(.) = (f_1(.), f_2(.), ..., f_n(.))^T$,

$\mathbf{f_i}(.) = f(\mathbf{h_i}), i = 1, 2, ..., n$

$(h_1, h_2, ..., h_n)^T = \mathbf{W_1(t)y(t)} + \mathbf{W_2(t)u(t)}$.

The main objective of the learning algorithm is to minimize the error function

$$\mathbf{E(t)} = \frac{1}{2} \sum_{k \in \mathbf{T(t)}} [e(t)]^2$$
$$= \frac{1}{2} \sum_{k \in \mathbf{T(t)}} [d_k(t) - y_k(t)]^2 \quad (2)$$

through updating the weight matrices w_{ij} and w^*_{ij}. Here, $\mathbf{E(t)}$ denotes the network error at time t. The term $e_k(t)$ represents the error between the desired output $d_k(t)$ and actual $y_k(t)$, where k belongs to the set $\mathbf{T(t)}$ of the output nodes with teaching status, at time t. $e_k(t) = 0$ when k is a hidden nodes.

In this paper, we use Chow and Fang's (Chow and Fang, 1999) iterative learning algorithm based on 2-D system theory for training RNNs. According to the idea of real-time iterative learning, an algorithm based on two-dimensional expression updates the connection weights to drive the output of the network to track the desired output within a required tolerance. The learning rule can be expressed as follows:

$$\mathbf{\Delta W(t,l)} = \mathbf{C^{-1}(t)} \left[\mathbf{d(t+1)} - \mathbf{y(t+1,l-1)}\right] . \left[\mathbf{x(t)^T x(t)}\right]^{-1} \mathbf{x(t)}^T$$
$$= \mathbf{C^{-1}e(t+1,l-1)} \left[\mathbf{x^T(t)x(t)}^{-1}\right] \mathbf{x(t)}^T \quad (3)$$

where
$\mathbf{C^{-1}(t)} = [diag(f'(\xi), f'(\xi), ..., f'(\xi))]^{-1}$,

$\xi = (\xi, \xi, ..., \xi)^T = \mathbf{W(t,l-1)x(t)}$.
$\mathbf{x(t)} = \begin{pmatrix} \mathbf{y(t)} \\ \mathbf{u(t)} \end{pmatrix}$,
$\mathbf{y(t+1,l)} = f^* \left[\mathbf{W(t,l)x(t)}\right]$,

$\mathbf{W(t,l)} = \mathbf{W(t,l-1)} + \mathbf{\Delta W(t,l)}, l = 1, 2, ..., k_t$.
Therefore, we can obtain

$$\mathbf{y(t+1)} = \mathbf{y(t+1,k_t)}$$
$$= f^* \left[\mathbf{W(t,k_t)x(t)}\right]$$
$$= f^*[\mathbf{W}(t)\mathbf{x}(t)] \quad (4)$$

It is clear that the weights of an RNN are adaptively determined under the real-time algorithm. Comparing Williams and Zipser's (Williams and Zipser, 1989) algorithm with (3), this real-time iterative learning algorithm has a dynamic learning rate of $\mathbf{C^{-1}(t)}\left[\mathbf{x(t)^T x(t)}\right]^{-1}$. In this paper, the algorithm is applied to model and control several type of nonlinear processes.

3. INTERNAL MODEL CONTROL USING RECURRENT NEURAL NETWORKS

In this section, internal model control scheme is employed for RNNs with the real-time algorithm. There are two stages within the control scheme according to characteristics of IMC. The first step is system identification by an RNN model and the second step is neural control design based on the model inverse approach. Figure 2 shows the structure of the internal model control using RNNs.

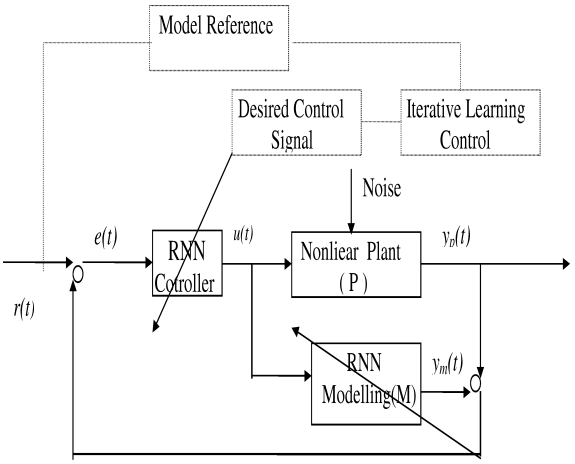

Fig. 2. The Internal Model Control Using RNNs

Stage one is an on-line identification and iterative learning control to obtain a desired control signal to the unknown nonlinear system. The idea of learning control is to utilize two RNNs, based on the same network architecture, to approximate the nonlinear system responses and to mimic the desired system response output. Then the desired control signal can be obtained by contrasting the two RNNs and the current system output and control input.

Stage 2: A recurrent neural controller is used to generate the desired control signal to the unknown nonlinear plant in an internal model control framework. The IMC structure is now well known and has been shown to underlie a number of control design techniques of apparently different origin. A unifying review of the IMC-type schemes was first presented by Garcia and Morari (Garcia and Morari, 1982). IMC has been shown to have a number of desirable properties; a detailed analysis has been given by Morari and Zafiriov (Morari and Zafiriov, 1989).

4. THE SIMULATION RESULTS

The simulations for several unknown nonlinear dynamic systems using the proposed control scheme are given in this section. The results have demonstrated that the proposed control method can drive systems to follow the desired trajectories with high accuracy.

Example 1: Consider the unknown system which is described by the difference equation:

$$y_p(k+1) = f[y_p(k), y_p(k-1)] + u(k) \quad (5)$$

where the function

$$f[y_p(k), y_p(k-1)]$$

$$= \frac{y_p(k)y_p(k-1)(y_p(k)+2.5)}{1+y_p^2(k)+y_p^2(k-1)} \quad (6)$$

is assumed to be unknown. A reference model is described by the second-order difference equation

$$y_d(k+1) = 0.2y_d(k) + 0.2y_d(k-1) + r(k) \quad (7)$$

where r(k) is a bounded reference input:

$$r(t) = 0.1\sin(\tfrac{2\pi t}{25})$$

According to the proposed control scheme, At first, two same RNNs are trained to get the desired control signal corresponding to the desired output. Figure 3 shows the output of the model and the output of the plant during the training. The tracking error is less than 0.001. The corresponding desired control signal is shown in Figure 4.

After the desired control signal is obtained, the RNN controller is trained on-line and the output is obtained using the internal model control structure. Figure 5 shows the control performance.

Example 2: The unknown nonlinear system is described by the following equation

$$y_p(k+1) = \frac{y_p(k)}{1+y_p^2(k)} + u^3(k) \quad (8)$$

The desired output is

$$y_m(k+1) = 0.1y_m(k) + r(k) \quad (9)$$

Assume the external input is a square wave as shown in Figure 6 (the dot line). According to the

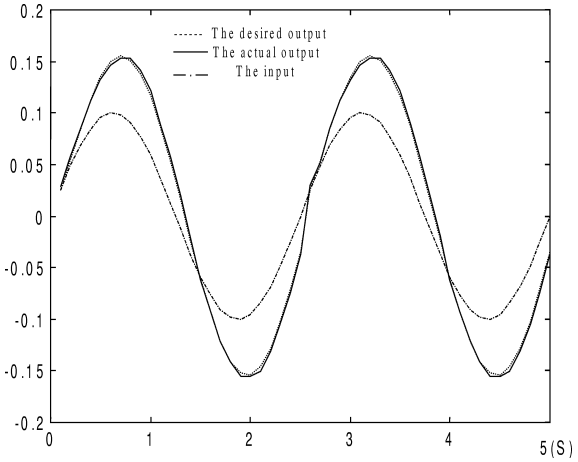

Fig. 3. The desired output and the output of the plant at stage 1 for *Example 1*

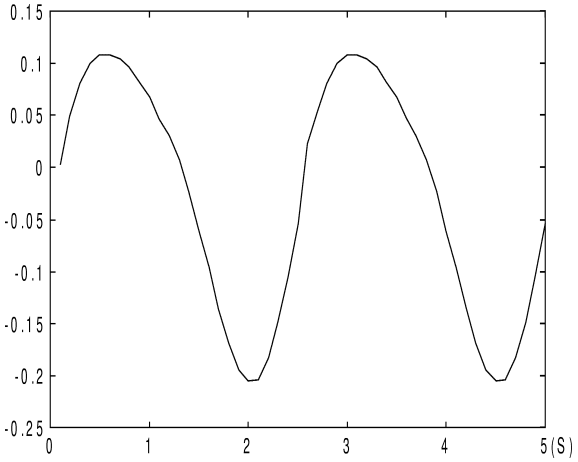

Fig. 4. The desired control signal for *Example 1*

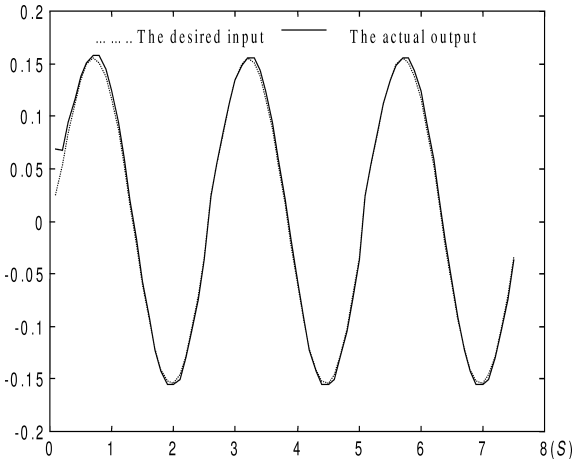

Fig. 5. The desired output and the actual output for *Example 1*

proposed control scheme, the desired control signal and the actual control signal which are shown in Figure 6 are obtained. The desired output and output of the plant obtained by internal model control are shown in Figure 7.

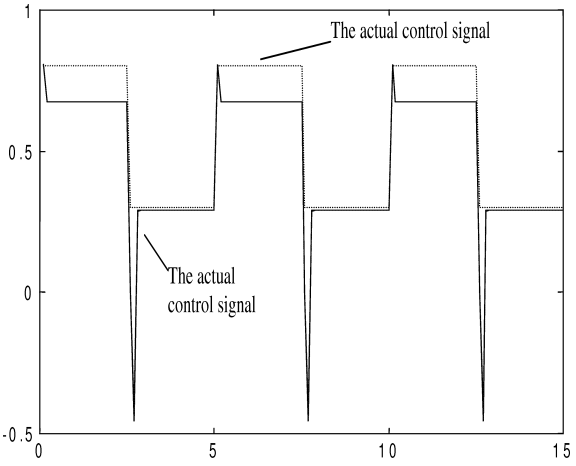

Fig. 6. The desired control signal, the actual control signal and external input for *Example 2*

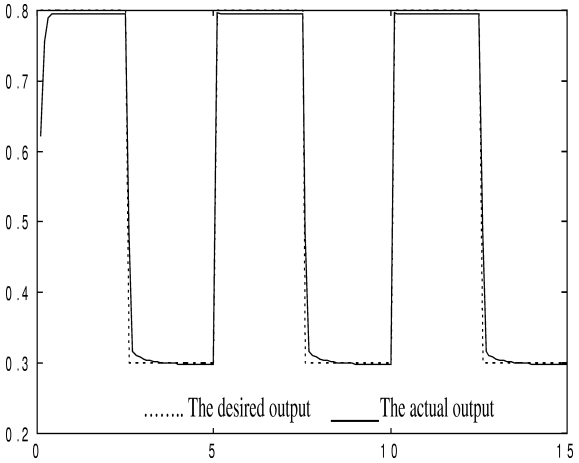

Fig. 7. The desired output and actual output for *Example 2*

Figure 8 is the performance with noise. It is observed that the outputs of the plants can track the desired trajectories very well. In *Example 1* and *2*, the structure of all the RNNs is one input, one hidden and one output node. They are all minimal networks. It is investigated that a small RNNs can implement a task as that a large or possibly infinite feedforward system does.

Example 3: The above two examples are SISO plants. In this example, an MIMO plant which is described by the following equation,

$$\begin{bmatrix} y_{p1}(k+1) \\ y_{p2}(k+1) \end{bmatrix} = \begin{bmatrix} \dfrac{y_{p1}(k)}{1+y_{p2}(k)} \\ \dfrac{y_{p1}(k)y_{p2}(k)}{1+y_{p2}^2(k)} \end{bmatrix} + \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix} \quad (10)$$

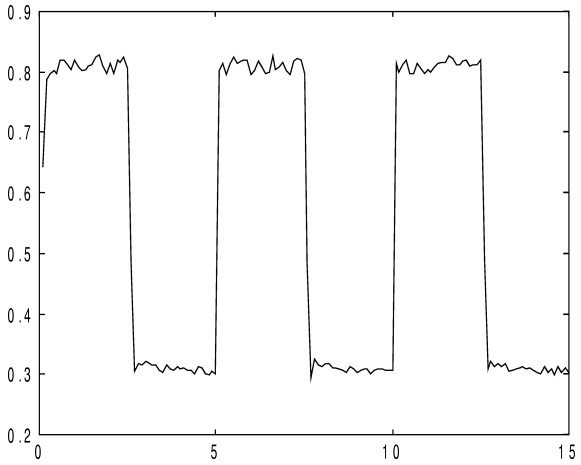

Fig. 8. The performance with noise for *Example 2*

The output of the reference model is

$$\begin{bmatrix} y_{m1}(k+1) \\ y_{m2}(k+1) \end{bmatrix} = \begin{bmatrix} 0.3 & 0.2 \\ 0.1 & 0.6 \end{bmatrix} + \begin{bmatrix} 0.3r_1(k) \\ 0.1r_2(k) \end{bmatrix} \quad (11)$$

Here, the external inputs have the form of

$$\begin{bmatrix} r_1(k) \\ r_2(k) \end{bmatrix} = \begin{bmatrix} \sin(\frac{2\pi k}{25}) \\ \cos(\frac{2\pi k}{25}) \end{bmatrix} \quad (12)$$

The desired control signals with respect to the desired outputs and the actual control signals U_{d1}, U_{d2} and U_{i1}, U_{i2} are shown in Figure 9 and 10.

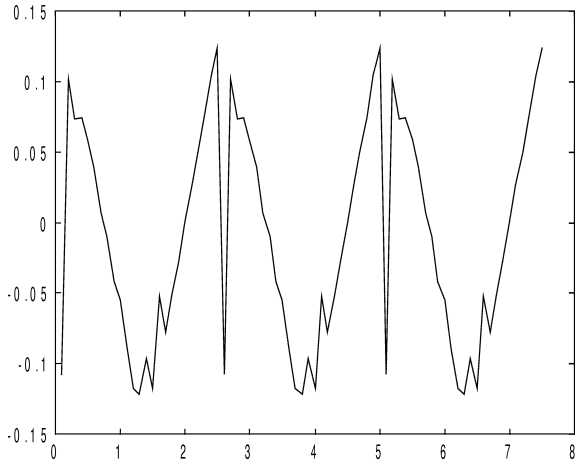

Fig. 9. The actual control signal U_{i1} and the desired control signal U_{d1} for *Example 3*

Figure 11, 12 show the final control results, two outputs Y_{p1}, Y_{p2} and the desired outputs Y_{d1}, Y_{d2}. In this example, the structure of the two RNNs consists of two input nodes, two hidden nodes and two output nodes. It is observed that the errors between the actual outputs of the plant and the desired outputs are larger than those in the SISO

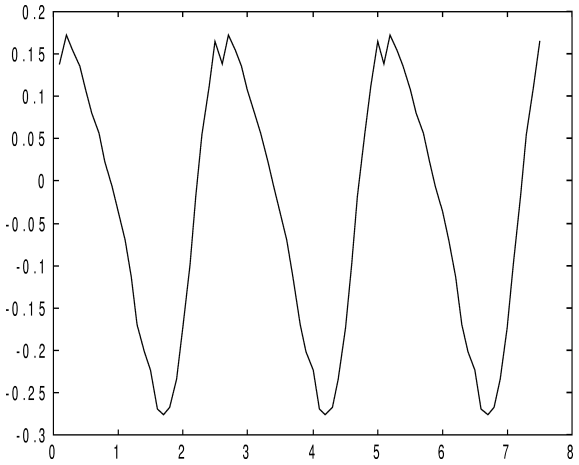

Fig. 10. The actual control signal U_{i2} and the desired control signal U_{d2} for *Example 3*

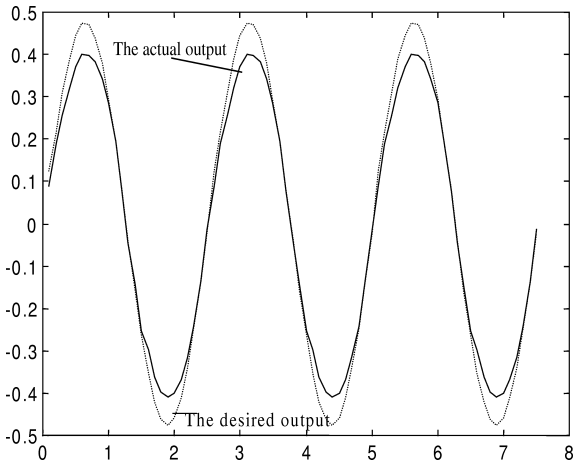

Fig. 11. The output Y_{p1} of the plant and the desired output for *Example 3*

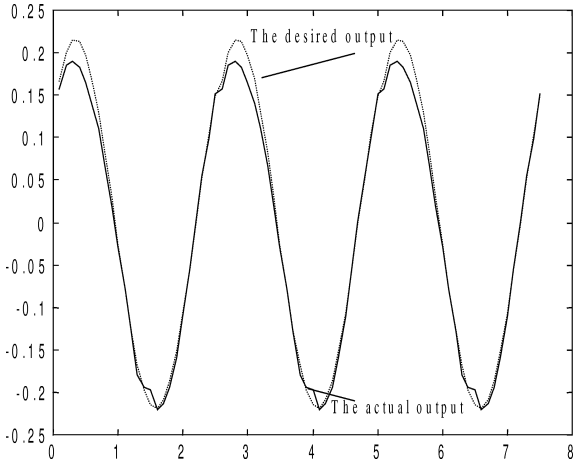

Fig. 12. The output Y_{p2} of the plant and the desired output for *Example 3*

situations. The main reason for this is that the errors between each of the two actual outputs and their corresponding desired outputs cannot reach the specified error tolerance simultaneously.

5. CONCLUSION

The paper presents a control method of using RNNs in an IMC framwork, and demonstrates their effectiveness for modelling and control of nonlinear dynamic systems. Unlike existing neural IMC design methods, the proposed control scheme consists of two stages. The first stage uses two same structure of RNNs through iterative learning techniques to obtain a desired control signal to the unknown nonlinear systems. Then one of the RNNs is used to generate the desired control signal within IMC structure. The algorithm for the RNNs is a real time iterative learning algorithm based on two-dimensional (2D) system theory. The simulation results for both the SISO and MIMO situations demonstrate the proposed control method can drive unknown systems to follow the desired trajectories very well.

6. REFERENCES

Chow, T.W.S. and Y. Fang (1999). A recurrent neural-network-based real-time learning control strategy applying to nonlinear systems with unknown dynamics. *IEEE Transactions on industrial electronics* **45(1)**, 151–161.

Garcia, C. and M. Morari (1982). Internal model control-1: A unifying review and some new results. pp. 308–326.

Li, Y., A. B. Rad and Y K Wong (1996). Model based control using artificial neural network. *Proceedings of the 1996 IEEE international symposium on intelligent control* **1**, 15–18.

Morari, M. and E. Zafiriov (1989). *Robust Process Control*. Prentice-Hall.

Narendra, K.S. and K. Parthasarthy (1991). Gradient methods of the dynamical systems containing neural networks. *IEEEE Transactions on Neural Networks* **2(2)**, 7–21.

Wen, P., C.K. Ng and Y. Li (1996). Dynamic linear square backpropagation algorithm for recurrent neural networks. *The Fourth International Conference on Control, Automation, Robotics and Vision* **1**, 3–6.

Williams, R. and D. Zipser (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation* **1**, 270–280.