Neural Networks and Dynamical Systems

Kumpati S. Narendra and Kannan Parthasarathy
Center for Systems Science,
Department of Electrical Engineering,
Yale University,
New Haven, Connecticut

ABSTRACT

Models for the identification and control of nonlinear dynamical systems using neural networks were introduced by Narendra and Parthasarathy in 1990, and methods for the adjustment of model parameters were also suggested. Simulation results of simple nonlinear systems were presented to demonstrate the feasibility of the schemes proposed. The concepts introduced at that time are investigated in this paper in greater detail. In particular, a number of questions that arise when the methods are applied to more complex systems are addressed. These include nonlinear systems of higher order as well as multivariable systems. The effect of using simpler models for both identification and control are discussed, and a new controller structure containing a linear part in addition to a multilayer neural network is introduced.

KEYWORDS: ***neural networks, dynamical systems, identification, control, backpropagation, dynamic backpropagation.***

INTRODUCTION

In a recent paper (Narendra and Parthasarathy [1]), we introduced several models containing neural networks for the identification and control of nonlinear dynamical systems. Prescriptive methods for the dynamic adjustment of the model parameters based on backpropagation were also suggested, and simulation results were included to demonstrate the feasibility of the proposed schemes. Since the models as well as the methods were being proposed for the

Address correspondence to K. S. Narendra, Department of Electrical Engineering, Yale University, P. O. Box 1968, Yale Station, New Haven, CT 06520.

International Journal of Approximate Reasoning 1992; 6:109–131

655 Avenue of the Americas, New York, NY 10010 0888-613X/92/$5.00

first time, the simulation studies dealt with relatively simple nonlinear plants.[1] In this paper, which can be considered a continuation of [1], many questions that arise when the methods are applied to more complex problems are discussed. These include higher order nonlinear systems as well as multivariable nonlinear systems. Further, the effect of using simpler models for both identification and control is also considered. Based on the observed simulation results, a modification is suggested in all the structures used for identification and control that includes a linear part along with the conventional multilayer neural networks.

MODELS FOR IDENTIFICATION AND CONTROL

In [1], four models of discrete-time single-input, single-output (SISO) plants were suggested and are described by the difference equations (1a)–(1d).

MODEL I

$$y_p(k+1) = \sum_{i=0}^{n-1} \alpha_i y_p(k-i) + g\big[u(k), \ldots, u(k-m+1)\big] \quad \text{(1a)}$$

MODEL II

$$y_p(k+1) = f\big[y_p(k), \ldots, y_p(k-n+1)\big] + \sum_{i=0}^{m-1} \beta_i u(k-i) \quad \text{(1b)}$$

MODEL III

$$y_p(k+1) = f\big[y_p(k), \ldots, y_p(k-n+1)\big] + g\big[u(k), \ldots, u(k-m+1)\big] \quad \text{(1c)}$$

MODEL IV

$$y_p(k+1) = f\big[y_p(k), \ldots, y_p(k-n+1); u(k), \ldots, u(k-m+1)\big], \quad m \le n \quad \text{(1d)}$$

where $[u(k), y_p(k)]$ represents the input-output pair of the plant at time k and $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^m \to \mathbb{R}$ are assumed to be differentiable functions of their arguments. The four models are shown pictorially in Figure 1, where z^{-1} is the unit delay operator. It was further assumed that f and g can be approxi-

[1] The word *plant* is used to refer to the system or process that is to be controlled.

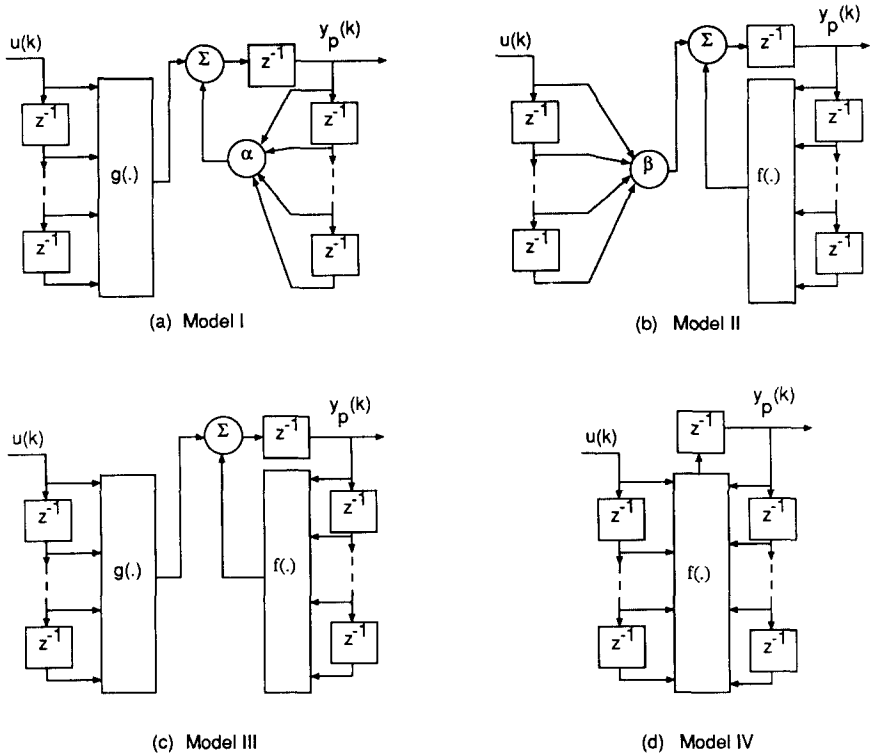

Figure 1. Recursive input-output models for nonlinear systems. (a) Model I. (b) Model II. (c) Model III. (d) Model IV.

mated to the desired degree of accuracy on compact sets by the multilayer networks used. The identification models, for identifying the input-output mapping represented by the plant, are chosen using the prior information available. In particular, the class of models to which the plant belongs and the magnitude of n and m in Eqs. (1) are needed for the choice of the identification model. Once the plant has been parametrized as described above, the parameters are adjusted using the output error and methods based on either static or dynamic backpropagation. Several simulation studies were reported in [1] in which the nonlinear plant was mostly first or second order and the exact model class (I–IV) was also known. In this paper we consider more complex examples of nonlinear plants. In some cases, we further assume that the specific class of models (i.e., I, II, III, or IV) to which the plant belongs is also unknown.

In the control problem, the controller is assumed to be a multilayer neural network with $2n$ inputs and one output if the plant order is assumed a priori to be n. The reference input r, the output $y_p(k)$, as well as its past $n - 1$ values

and the $n - 1$ past values of $u(k)$ constitute the inputs to the controller. The difficulty encountered in solving the control problem depends upon the assumptions made regarding the structure of the plant. If the plant output $y_p(k + 1)$ is assumed to depend linearly on the past values of the control unit $u(.)$, the determination of a suitable controller is considerably simplified. In this paper we consider more general cases of the plant model where simple procedures do not exist for computing the control input.

ADAPTIVE CONTROL OF NONLINEAR DYNAMICAL SYSTEMS

As mentioned in the preceding section, the identification and control models used in the simulation studies reported in [1] were based on prior information concerning the plant dynamics. The nonlinear plants that were identified and controlled were also relatively simple, being for the most part low-order SISO systems, with the output at instant $k + 1$ being affected by the input at time k in a linear fashion. In this section we consider situations when the problem is rendered more complex due to the higher order dynamics of the plant, its multivariate nature, the output being nonlinearly dependent on the input or time delays existing within the plant, so that the input $u(k)$ at time k affects the output y at time $k + i$ where $i > 1$. These different questions are discussed using several examples, with each example used to clarify a specific point.

Example 1

A nonlinear plant is described by the difference equation

$$y_p(k + 2) = f[y_p(k + 1), y_p(k)] + u(k) \tag{2}$$

where

$$f[y_p(k + 1), y_p(k)] = \frac{y_p(k + 1)y_p(k)[y_p(k + 1) + 2.5]}{1 + y_p^2(k + 1) + y_p^2(k)}$$

and a linear reference model is described by the second-order equation

$$y_m(k + 2) = 0.6y_m(k + 1) + 0.2y_m(k) + r(k)$$

where $r(k)$ is a uniformly bounded reference input. The objective, as in all model-following problems, is to determine input $u(.)$ such that $\lim_{k\to\infty} | y_p(k) - y_m(k) | < \epsilon$ for some specified value of ϵ.

The main difference between the problem considered here and those discussed in [1] is that the input to the plant at time k affects the output only at time $k + 2$ (the plant is said to have a relative degree 2). If in Eq. 2 the

right-hand side contained $u(k) + 1)$ rather than $u(k)$, a control input that would achieve the desired objective could be written by inspection as

$$u(k) = -f[y_p(k), y_p(k-1)] + 0.6y_p(k) + 0.2y_p(k-1) + r(k-1) \tag{3}$$

However, since the right-hand side contains only $u(k)$, the control law given in Eq. (3) cannot be used as it depends on future values of $y_p(.)$. As it is known that the output $k + 1$ depends on its past values as well as on the value of u at time $k - 1$, it follows that $y_p(k + 1)$ can be computed from the values of y_p at k and $k - 1$ as well as $u(k - 1)$. Hence

$$u(k) = -f[f[y_p(k), y_p(k-1)] + u(k-1), y_p(k)] + 0.6[f[y_p(k), y_p(k-1)] + u(k-1) + 0.2y_p(k) + r(k) \tag{4}$$

In other words, $u(k)$ is a nonlinear function of $y_p(k)$ and $y_p(k - 1)$ as well as $u(k - 1)$.

One way of determining the control input is to use the input-output data from the plant to determine the estimate of $\hat{f}$ of f and use $\hat{f}$ in place of f in Eq. (4). This method therefore presupposes complete identification before control is initiated.

An alternative approach is to use a multilayer network for the controller so that its output [i.e., the input to the plant $u(.)$] depends upon the past values of u as well as the past values of the output y_p of the plant. Once again it is assumed that the plant has been identified sufficiently accurately offline. Further, it is also assumed that the neural network N_c used as the controller can generate the desired control input for some values of its parameters, that is, $N_c[y_p(k), y_p(k - 1), u(k - 1)] + r(k)$ can approximate the right-hand side in Eq. (4). The manner in which the controller parameters are adjusted to achieve this is discussed later in this section.

The results of controlling the plant given in Eq. (2), using dynamic backpropagation (Narendra and Parthasarathy [2]) to adjust the parameters of a controller, are shown in Figure 2. The response of the plant without and with a controller are shown in Figures 2a and 2b, respectively. In Figure 2a, $r(k)$ is a sinusoidal signal $\sin(2\pi k/25)$, and the solid curve represents the output of the linear reference model. The output of the nonlinear plant without a controller is shown in dotted lines. The same signals are shown in Figure 2b when a controller is used that is based on the identified model.

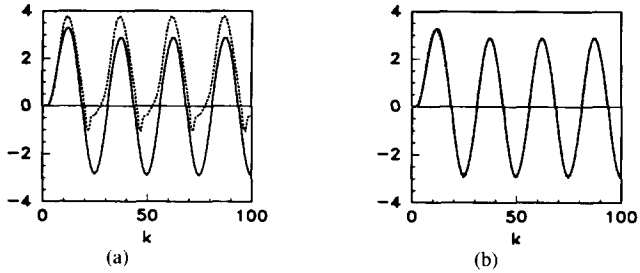

Figure 2. Example 1: (a) Outputs of the reference model and the plant without control. (b) Outputs of the reference model and the controlled plant.

Example 2: Online Control

In [1], Example 11 considered the control of a plant described by the difference equation

$$y_p(k+1) = f[y_p(k)] + g[u(k)] \tag{5}$$

where $f[y_p] = y_p/(1 + y_p^2)$ and $g[u] = u^3$.

The objective, once again, is to determine a bounded control input so that the output of the plant follows the output of the reference model, $y_m(k+1) = 0.6y_m(k) + r(k)$, sufficiently accurately.

In [1], f and g are first estimated as $\hat{f}$ and $\hat{g}$ based on random inputs to the stable plant. Following this, $\widehat{g^{-1}}$ was estimated and used to compute $u(k)$ as shown below[2]:

$$u(k) = \widehat{g_k^{-1}}[-\hat{f}_k[y_p(k)] + 0.6y_p(k) + r(k)] \tag{6}$$

From Eq. (6), it is seen that the control input $u(k)$ depends explicitly on the estimates $\widehat{g_k^{-1}}$ and $\hat{f}_k$ and is completely determined once the plant is identified. Hence, in this case, parametrization of the controller and updating of the control parameters are not needed.

In classical adaptive control theory, the main problem is to control the feedback system even while the plant is being identified. In fact, demonstrating that the adaptive loop, with the identifier and controller operating simultaneously, is globally stable was the major contribution to the field in 1980 (Narendra et al. [3], Morse [4], Goodwin et al. [5], Narendra and Lin [6]). By assuming that the plant is stable and that it can be identified offline, this

[2] Strictly speaking, what is obtained is not $\widehat{g_k^{-1}}$ (the estimate of g_k^{-1}) but the inverse of the estimate $\hat{g}_k$. It is tacitly assumed that the two are approximately the same.

problem was entirely sidestepped in [1]. Such an approach had to be used because, in the absence of a well-developed theory for the adaptive control of nonlinear systems, very little can be said about simultaneous identification and control of such systems, starting at some initial time t_0. However, assuming that the plant to be controlled has been identified offline and that reasonably accurate initial estimates of f and g are available, simultaneous identification and control was found to be successful.

Figures 3a and 3b indicate the initial estimates of f and g in Eq. (5) obtained by offline identification over 1000 time steps. At this stage, the error between the output of the plant and the output of the reference model in response to the reference input $r(k) = \sin(2\pi k/25) + \sin(2\pi k/10)$ is seen to be quite large (Fig. 4a). The identification error between the plant and the identification model is shown in Figure 4b and also indicates that the identification process is not complete. The control in the present case is thus initiated with partial information, and the identification process is allowed to continue. The control input at every stage is computed using Eq. (6) as before, except that $\hat{g}$, $\widehat{g^{-1}}$, and $\hat{f}$ are updated online. The manner in which the output of the plant approximates the output of the reference model after 100 time steps is shown in Figure 5a, and Figure 5b shows the outputs of the plant and the identification model. Figure 6 shows the same signals after 10,000 steps, and it is seen that the plant output, the output of the identification model, and the output of the reference model are practically indistinguishable. The experiment shows that identification and control may proceed simultaneously in a stable fashion provided that the initial estimate of the plant is sufficiently accurate.

Adjustment of Controller Parameters Using Dynamic Backpropagation

The examples treated in [1], as well as Examples 1 and 2 discussed earlier here, use specific information concerning the structure of the nonlinear plant to determine the control input. In particular, the control term enters linearly in Example 1 while $g[u]$ is invertible in Example 2. In more general cases such

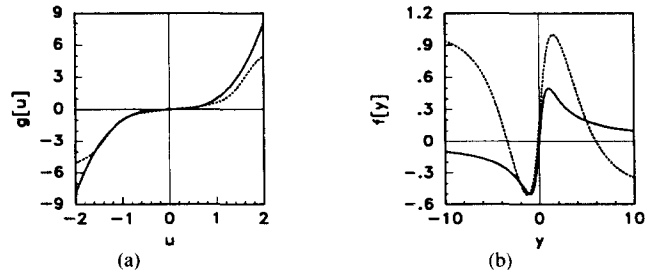

Figure 3. (a) Initial estimate of g. (b) initial estimate of f.

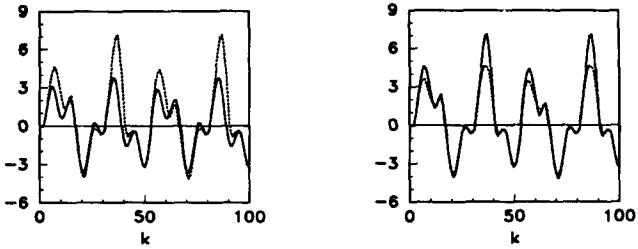

Figure 4. Example 2: (a) Responses of the reference model and the plant. (b) Responses of the plant and the identification model.

information may not be available, and the controller parameters have to be adjusted based only on the available input-output data. We introduced a method for extending backpropagation to dynamic systems in [1] and it was discussed in detail in [2]. However, we include here the most general structure for the plant and controller and indicate how the control parameters are updated in such cases before applying it in Example 3 to a specific problem of nonlinear control.

Let a plant be described by the difference equation

$$y_p(k+1) = f\left[y_p(k), \ldots, y_p(k-n+1); u(k), \ldots, u(k-m+1)\right]$$

which corresponds to model IV in [1]. Using the method outlined in [1], this is identified using a dynamical system with multilayer neural networks as

$$\hat{y}_p(k+1) = N_f\left[y_p(k), \ldots, y_p(k-n+1); u(k), \ldots, u(k-m+1)\right]$$

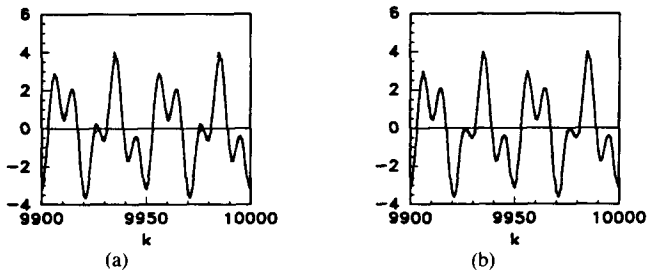

Figure 5. (a) Responses of the refernce model and the controlled plant. (b) Outputs of the plant and the identification model.

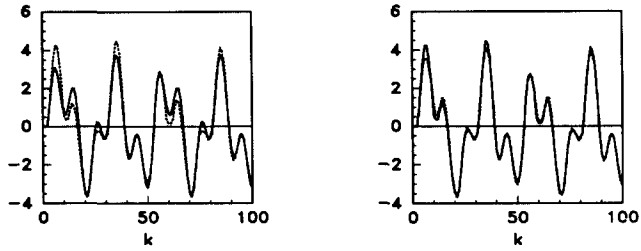

Figure 6. (a) Outputs of the reference model and the plant after 10000 steps. (b) Outputs of the plant and the identification model.

We now assume that a multilayer neural network which inputs $r(k)$, $y_p(k)$, $y_p(k-1), \ldots, y_p(k-n+1), u(k-1), \ldots, u(k-m+1)$ exists that can generate the desired control input $u(k)$ such that the output of the plant follows the output of a reference model asymptotically. Hence, the problem of control is to determine an adaptive law for updating the parameters of the controller based on the measured signals of the system as well as the error between the plant and reference model outputs. For the purpose of determining $u(k)$, it is assumed that the process has been identified completely (or at least sufficiently accurately as in Example 2) and that the identification model can be represented in feedback form as

$$\hat{y}_p(k+1) = N_f[\hat{y}_p(k), \ldots, \hat{y}_p(k-n+1); u(k), \ldots, u(k-m+1)]$$

(note that this representation is needed to adjust the parameters of the controller as described below). The control input $u(k)$ is then given, as described earlier, by

$$u(k) = N_c[\hat{y}_p(k), \ldots, \hat{y}_p(k-n+1); u(k-1), \ldots, u(k-m+1); r(k); \Theta]$$

where Θ represents the set of controller parameters. In the dynamic backpropagation approach, the partial derivatives of a performance index based on the output error $e(k) \triangleq \hat{y}_p(k) - y_m(k)$ with respect to the parameters $\theta \in \Theta$ are determined, and the parameters in turn are adjusted along the negative gradient of a performance function. For further details, the reader is referred to Narendra and Parthasarathy [2].

For the problem under discussion, if θ is a typical parameter of $N_c[.]$, $\partial e(k)/\partial\theta = \partial\hat{y}_p(k)/\partial\theta$ is given by the linearized difference equations

$$\frac{\partial\hat{y}_p(k+1)}{\partial\theta} = \frac{\partial N_f[.]}{\partial\hat{y}_p(k)}\frac{\partial\hat{y}_p(k)}{\partial\theta} + \cdots + \frac{\partial N_f[.]}{\partial\hat{y}_p(k-n+1)}$$

$$\frac{\partial\hat{y}_p(k-n+1)}{\partial\theta} + \frac{\partial N_f[.]}{\partial u(k)}\frac{\partial u(k)}{\partial\theta} + \cdots$$

$$+ \frac{\partial N_f[.]}{\partial u(k-m+1)}\frac{\partial u(k-m+1)}{\partial\theta} \tag{7a}$$

and

$$\frac{\partial u(k)}{\partial\theta} = \frac{\partial N_c[.]}{\partial\hat{y}_p(k)}\frac{\partial\hat{y}_p(k)}{\partial\theta} + \cdots + \frac{\partial N_c[.]}{\partial\hat{y}_p(k-n+1)}\frac{\partial\hat{y}_p(k-n+1)}{\partial\theta}$$

$$+ \frac{\partial N_c[.]}{\partial u(k-1)}\frac{\partial u(k-1)}{\partial\theta} + \cdots + \frac{\partial N_c[.]}{\partial u(k-m+1)}$$

$$\frac{\partial u(k-m+1)}{\partial\theta} + \frac{\partial N_c[.]}{\partial\theta} \tag{7b}$$

Since the required partial derivatives $\partial N_f[.]/\partial\hat{y}_p(i)$, $\partial N_c[.]/\partial\hat{y}_p(i)$, $\partial N_f[.]/\partial u(i)$, and $\partial N_c[.]/\partial u(i)$ can be generated using backpropagation, the above linearized equations can be implemented to obtain $\partial\hat{y}_p(k)/\partial\theta$ online. This in turn is used to adjust the parameter θ with a small step size η. The dynamical system given in Eq. (7), which generates the desired partial derivatives $\partial\hat{y}_p(k)/\partial\theta$, is referred to as the sensitivity model.

The fact that the plant is assumed to belong to model class IV necessitates the use of a general controller structure and dynamic backpropagation. From Eq. (7) it is clear that the determination of the partial derivative of $\hat{y}_p(k)$ with respect to a single parameter θ involves the setting up of a complex sensitivity model. This accounts for the choice of simpler models (discussed later) to represent the plant wherever possible.

EXAMPLE 3 The method of dynamically adjusting the parameters of the controller discussed above is now applied to determine a controller for the problem considered in Example 2 but without using the inverse $\hat{g}^{-1}$ of the estimate $\hat{g}[\,]$. The model of the plant together with the controller now has

the form

$$\hat{y}_p(k+1) = N_f[\hat{y}_p(k)] + N_g[N_c[\hat{y}_p(k), r(k)]]$$

where $N_f[.]$ and $N_g[.]$ have been determined by the identification procedure outlined earlier. Hence, our objective is to determine the parameters of the controller so that the output error is minimized in some sense. The partial derivative of the error $e(k) \triangleq \hat{y}_p(k) - y_m(k)$ with respect to a typical parameter θ of the controller (i.e., θ is a weight in the neural network N_c) is given by the output of a sensitivity model described by a linear time-varying difference equation. If a different sensitivity model is used for each of the parameters, the gradient of a performance index with respect to the control parameter vector can be determined and the parameters adjusted along the negative gradient using a sufficiently small step size.

Figure 7 shows the output of the plant after the adjustment process is completed. The output error is seen to be negligible, and the performance compares favorably with that obtained in Example 2 using the inverse of the estimate of g.

System Identification

In the examples of plant identification considered in [1], it was assumed that the plant model was in one of the standard forms I–IV and that the designer knew which of the specific sets it belonged to. Since the relevant nonlinear functions can be approximated to any degree of accuracy by a multilayer neural network, it follows that the existence of a solution is ensured at the outset. In contrast to that, we consider in this section plants that are not in standard form (i.e., models I–IV) so that the existence of a solution is not assumed a priori. In such cases our interest is in determining the degree to which one of the models can approximate the given plant.

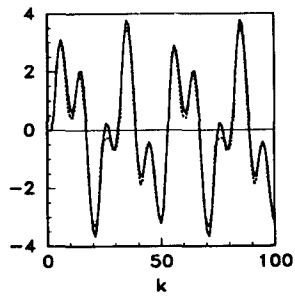

Figure 7. Example 3: Responses of the reference model and the controlled plant.

EXAMPLE 4 In this case the plant is described by the second-order nonlinear difference equations

$$x_1(k+1) = \frac{x_1(k) + 2x_2(k)}{1 + x_2^2(k)} + u(k)$$

$$x_2(k+1) = \frac{x_1(k)x_2(k)}{1 + x_2^2(k)} + u(k)$$

$$y_p(k) = x_1(k) + x_2(k)$$

The plant is described by a state-space model and does not admit any of the input-output equations (1a)–(1d). Thus, the plant does not belong to any of the standard models given in [1]. Hence, we choose the most general representation considered in [1], that is, model IV, and represent the identification model as

$$\hat{y}_p(k+1) = N[y_p(k), y_p(k-1), \ldots, y_p(k-m+1); u(k), u(k-1), \ldots, u(k-m+1)]$$

where m is a suitably chosen integer.

The parameters of N can be adjusted using static backpropagation based on the error $e(k)$, where

$$e(k) \triangleq \hat{y}_p(k) - y_p(k)$$

The first choice of the value of m is 2 because the plant is known to be of second order [i.e., two past values of input and output are used to generate $\hat{y}_p(k+1)$]. The network $N \in \mathcal{N}^3_{4,30,20,1}$, the step size $\eta = 0.05$, and an i.i.d. random input with a uniform distribution over the interval $[-1, 1]$ was used to estimate the plant parameters. The output of the plant $y_p(k)$ and the estimated output $\hat{y}_p(k)$ for a test input $u(k) = \sin(2\pi k/25)$ are shown in Figure 8a. The improvement in the identification when m was increased to 3, 5, and 10 are shown in Figures 8b, 8c, and 8d, respectively, and is seen to be monotonic. The best results were obtained with a network $N \in \mathcal{N}^3_{20,40,20,1}$ to which 10 past values of the output and 10 past values of the input were used as the input vector.

This simulation is a typical example of a plant whose structure is not in standard form but that nevertheless can be identified accurately by model IV. Empirical observations of similar situations reveal that model IV can be used to

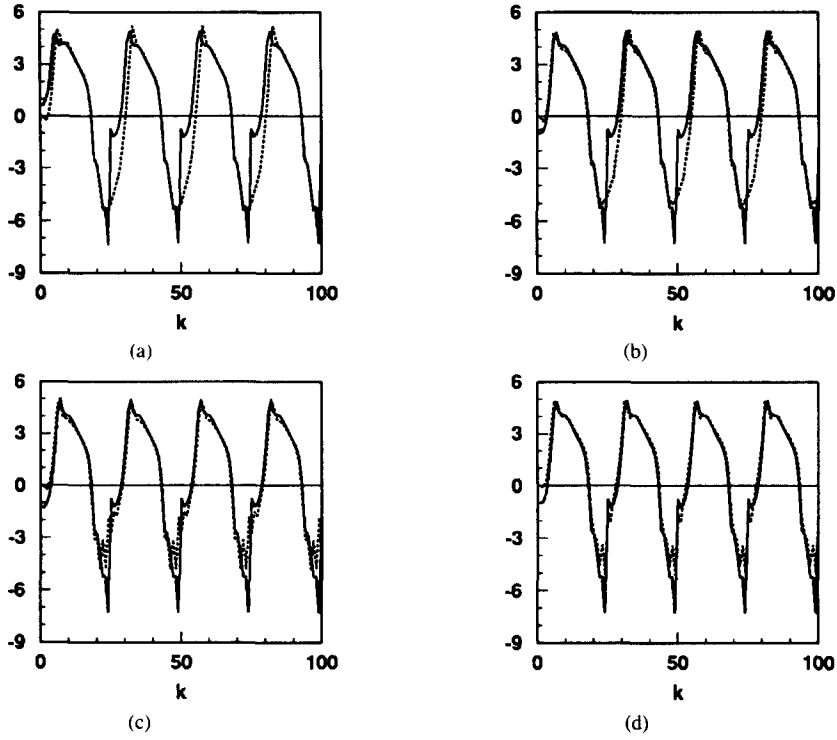

Figure 8. Outputs of the plant and identification model for a test input $u(k) = \sin(2\pi k/25)$. (a) $m = 2$ (b) $m = 3$ (c) $m = 5$ (d) $m = 10$.

identify a rather large class of nonlinear systems. It also brings into focus the fact that very little theory currently exists in this area to guide us in the choice of models.

Use of Simpler Models for Identification and Control

The choice of the identification model used to approximate the input-output map represented by the plant is naturally determined by the a priori information available regarding the plant structure. If, for example, it is known that the plant model belongs to model class III, the structure of the identification model is chosen to correspond to this. In fact, in all the examples presented in [1], such information concerning the plant model was assumed to be available.

When the class of models to which the plant belongs is not known, it is reasonable from a practical standpoint to assume the simplest model for

identification and control purposes and increase the complexity of the model only when the approximation is not sufficiently accurate. In this section we present simulation studies on a plant using simpler identification and control models.

EXAMPLE 5 In this case the plant is described by the the third-order difference equation

$$y_p(k+1) = \frac{y_p(k)\,y_p(k-1)\,y_p(k-2)\,[y_p(k-2)-1]\,u(k-1)+u(k)}{1+y_p^2(k-1)+y_p^2(k-2)}$$

Although the plant belongs to model IV, it is assumed that this information is not available a priori. In the absence of such information, the following models of increasing generality were used to identify the system:

i) $\hat{y}_p(k+1) = w_1 y_p + w_2 y_p(k-1) + w_3 y_p(k-2) + w_4 u(k) + w_5 u(k-1) + w_6 u(k-2)$

(ii) $\hat{y}_p(k+1) = N[y_p(k), y_p(k-1), y_p(k-2)] + w_4 u(k) + w_5 u(k-1) + w_6 u(k-2)$

(iii) $\hat{y}_p(k+1) = w_1 y_p(k) + w_2 y_p(k-1) + w_3 y_p(k-2) + N[u(k), u(k-1), u(k-2)]$

(iv) $\hat{y}_p(k+1) = N_1[y_p(k), y_p(k-1), y_p(k-2)] + N_2[u(k), u(k-1), u(k-2)]$

(v) $\hat{y}_p(k+1) = N[y_p(k), y_p(k-1), y_p(k-2), u(k), u(k-1), u(k-2)]$

The identification model (i) is linear and was found to perform very poorly. In particular, the output of the linear system follows the output of the plant with a small error as long as the parameters are adjusted continuously, but yields a very large (and even exponentially growing) error when the parameters are frozen at any terminal time.

In the case of (ii), the parameters of the multilayer neural network $N \in \mathcal{N}^3_{3,20,10,1}$ as well as the linear gains were adjusted using a gradient algorithm. In Figure 9, the response of the model is shown for an input $u(k) = \sin(2\pi k/100)$. During the interval [0, 99,800] the parameters are adjusted using backpropagation. For $k > 99{,}800$, the values of the parameters are frozen and used in a parallel (rather than a series-parallel) model. It is seen that a member of model class II approximates the plant quite accurately.

The identification models (iii), (iv), and (v), belonging to model classes I, III, and IV, respectively, also behaved in a similar manner, yielding a similar order of approximation. This indicates that simpler models may be adequate for identification purposes. Model II, which is linear in the input u, is best

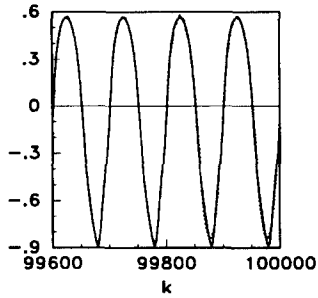

Figure 9. Example 5: Outputs of the plant and the identification model.

suited for control purposes because control can be directly effected using the identification model. Hence, the control of the given nonlinear plant can be attempted using model (ii). Figure 10 shows the response of the given nonlinear system using such a model when a controller is designed using the methods outlined in earlier examples. Although the performance was not satisfactory for large reference inputs (and is sometimes even unstable), Figure 10 shows a very small output error when $r(k) = 0.5 \sin(2\pi k/100)$ and the reference model is $y_m(k+1) = 0.6y_m(k) + r(k)$.

Identification of Unforced Nonlinear Systems

Thus far, the identification schemes have used plant input-output data to identify the unknown nonlinear plants. In this section we consider cases where the unknown plant is nonlinear but is not forced by an external input. In Example 6, we consider the famous Van der Pol equation, which exhibits a limit cycle. Our objective in this case is to determine a feedback system with a

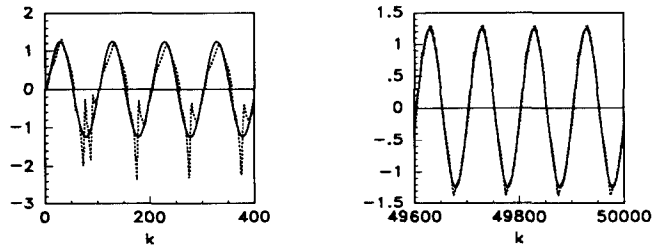

Figure 10. Example 5: Outputs of the reference model (solid line) and the plant (dotted line) which is controlled using model (ii). (a) Response for $k \in [0, 400]$ and (b) response for $k \in [49600, 50000]$.

multilayer neural network as a component whose output approximates the limit cycle exhibited by the given system. Example 7 discusses the problem of predicting a chaotic time series generated by an autonomous delay differential equation. Our objective in this case is to use four samples, $x(t)$, $x(t - t_1)$, $(x(t - 2t_1)$, $x(t - 3t_1)$, and predict the value of $x(t + T)$ for some specified values of t_1 and T.

EXAMPLE 6 Consider the van der Pol equation

$$\ddot{x}(t) + \epsilon\left[x^2(t) - 1\right]\dot{x}(t) + x(t) = 0 \tag{8}$$

Equation (8) exhibits a stable limit cycle for every positive value of ϵ. We assume that at every instant $x(t)$, $\dot{x}(t)$, and $\ddot{x}(t)$ can be measured. As stated earlier, the objective is to determine a dynamical system described by the equation

$$\ddot{z} + N[z, \dot{z}] = 0$$

whose output $z(t)$ approximates $x(t)$ at every instant $t \in \mathbb{R}^+$, where $N: \mathbb{R}^2 \to \mathbb{R}$ is realized by a multilayer neural network.

The neural network N used for this purpose belongs to the class $\mathcal{N}^3_{2,20,10,1}$. The inputs to the network are $x(t)$ and $\dot{x}(t)$, respectively, and the output of the network $N[x, \dot{x}]$ is compared with $-\ddot{x}(t)$ to generate the error $e(t)$ used to adjust the parameters of the network, where

$$e(t) \triangleq N\left[x(t), \dot{x}(t)\right] + \ddot{x}(t)$$

Several initial conditions were chosen, and the system was run for only a finite amount of time to ensure satisfactory approximation of the nonlinear function. Since Eq. (8) exhibits a limit cycle, if the system is not reinitialized after a finite amount of time the nonlinear function is approximated only in the vicinity of the limit cycle. The error $e(t)$ was found to be approximately zero after 500,000 trials. At this stage the neural network was used as a component in the feedback path as shown in Figure 11 to generate $z(t)$.[3] The trajectories of $[x(t), \dot{x}(t)]$ and $[z(t), \dot{z}(t)]$ are shown in Figures 12a and 12b, respectively, for the initial condition $(-2.9, -2.8)$.

Comment. The assumption that $\ddot{x}(t)$ can be measured directly greatly simplified the problem and allowed the use of static backpropagation for determining the parameters of N. If, however, only $x(t)$ [or $x(t)$ and $\dot{x}(t)$] can be measured every instant, dynamic backpropagation has to be used based on the error $z(t) - x(t)$ [or $z(t) - x(t)$ and $\dot{z}(t) - \dot{x}(t)$].

[3] The symbol $\frac{1}{s}$ is used to denote an integrator.

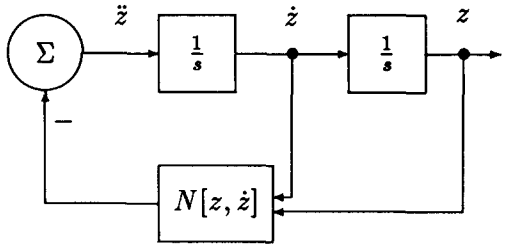

Figure 11. System used to generate $z(t)$.

EXAMPLE 7 A chaotic time series is generated by the integration of the Mackey–Glass differential equation

$$\dot{x}(t) = -bx(t) + a\frac{x(t-\tau)}{1 + x^{10}(t-\tau)} \tag{9}$$

where $a = 0.2$, $b = 0.1$, and $\tau = 17$ as suggested by Moody and Darken [7]. The function $x(t)$, obtained by integrating Eq. (9), is shown in Figure 13a and is quasi-periodic, as no two cycles are the same. The objective is to use four samples separated by $\Delta = 6$ units $[x(t), x(t-6), x(t-12), x(t-18)]$ to predict the value of $x(t+85)$.

For purposes of prediction we use a discrete-time dynamical model containing a three-layer neural network belonging to the class $\mathcal{N}^3_{4,20,10,1}$. The output of the network $\hat{y}(k) = \hat{x}(k+85)$ is related to the values $x(k)$, $x(k-6)$, $x(k-12)$, and $x(k-18)$ by the nonlinear equation

$$\hat{y}(k) = N[x(k), x(k-6), x(k-12), x(k-18)]$$

To train the network, the observed values of x at k, $k-6$, $k-12$, and $k-18$ as well as at $k+85$ are used. The time k is increased by 0.1, and the

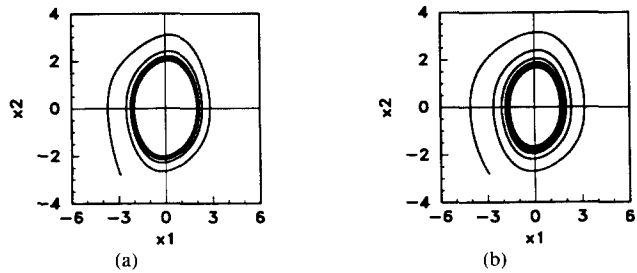

Figure 12. Behavior of the systems for initial condition $(-2.9, -2.8)$. (a) Actual system (b) Identified system.

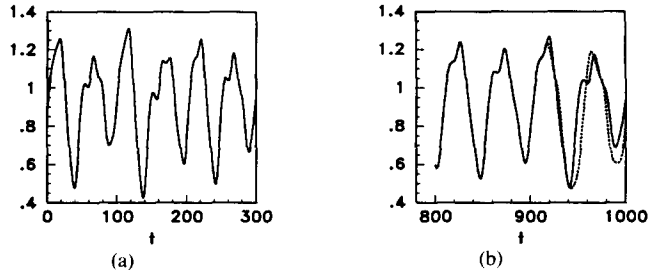

Figure 13. (a) Mackey–Glass chaotic time-series. (b) Prediction using a multilayer network.

process is repeated. Starting at time $t = -103$ and assuming that $x(t)$ has been observed over the interval $[-103, 915]$, the value of $x(k + 85)$ is predicted on the basis of $x(k)$, $x(k - 6)$, $x(k - 12)$, and $x(k - 18)$, and the error $\hat{y}(k) - x(k + 85)$ is used to update the parameters of N using backpropagation. The test sequence consists of the values of $x(t)$ over the interval $[830, 915]$ and is used to predict the values of $x(t)$ over the interval $[915, 1000]$; for example, the values of x at 812, 818, 824, and 830 are used to predict the value of x at $t = 915$. Figure 13b shows $x(t)$ (solid line) as well as the predicted values (dotted line) over the interval specified. The step size used in the gradient method was set to $\eta = 0.25$.

Comment. In the prediction described above, it is tacitly assumed that the value of x at time k can be predicted fairly accurately on the basis of a finite number of past values. In more realistic situations, the value of a function x_1 at instant k may be a function of not only its past values but also the past values of other functions $x_2, x_2, \ldots, x_N$. In this case the multilayer neural network must map an Nn dimensional vector to the real line as shown in Figure 14.

Identification of Multivariable Systems

The use of multilayer neural networks to identify mappings $f: \mathbb{R}^n \to \mathbb{R}^m$ is no more complex conceptually than the identification of functions mapping $\mathbb{R}^n$ into the real line. Hence the procedure outlined for identifying SISO systems also carries over to the multi-input, multi-output (MIMO) case. In [1], a multivariable version of model II was presented. In the following we describe the extensions of model IV to the multivariable case.

Let $u(k) \in \mathbb{R}^m$ and $y_p(k) \in \mathbb{R}^n$ represent the input and output, respectively, at time step k. The plant in this case (corresponding to model IV) has the form

$$y_p(k + 1) = f[y_p(k), y_p(k - 1), \ldots, y_p(k - n_1 + 1); u(k), u(k - 1), \ldots, u(k - m_1 + 1)] \tag{10}$$

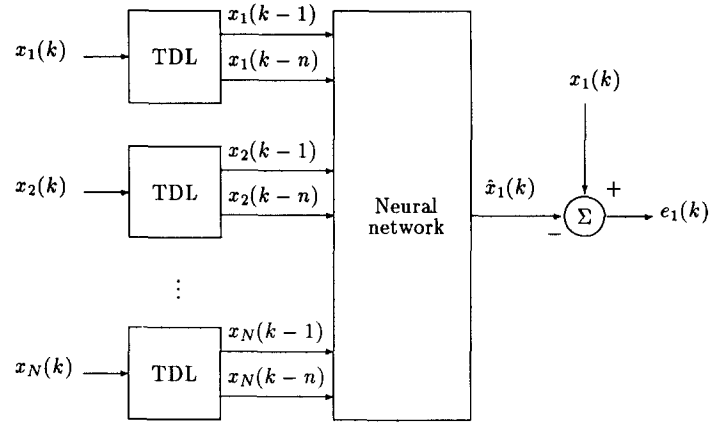

Figure 14. Time-series prediction in a general case.

where the function f maps $\mathbb{R}^{nn_1+mm_1}$ to $\mathbb{R}^n$. The identification model in this case consists of $n + m$ tapped delay lines in the forward and feedback paths corresponding to all inputs and outputs, respectively. The neural network maps the $(nn_1 + mm_1)$-dimensional input vector to $\mathbb{R}^n$.

EXAMPLE 8 To demonstrate the fact that identification of MIMO nonlinear systems can be carried out in the same fashion as in the SISO case using multilayer neural networks, the input and output of the plant were assumed to be two-dimensional vectors $u(k) = [u_1(k), u_2(k)]^T$ and $y_p(k) = [y_p(k), y_{p2}(k)]^T$. The difference equation describing the plant was assumed to be of the form

$$\begin{bmatrix} y_{p1}(k+1) \\ y_{p2}(k+1) \end{bmatrix} = \begin{bmatrix} f_1[y_{p1}(k), y_{p2}(k), u_1(k), u_2(k)] \\ f_2[y_{p1}(k), y_{p2}(k), u_1(k), u_2(k)] \end{bmatrix}$$

where the unknown functions f_1 and f_2 have the form

$$f_1(y_{p1}, y_{p2}, u_1, u_2) = \frac{0.8y_{p1}^3 + u_1^2 u_2}{2 + y_{p2}^2}$$

and

$$f_2(y_{p1}, y_{p2}, u_1, u_2) = \frac{y_{p1} - y_{p1}y_{p2} + (u_1 - 0.5)(u_2 + 0.8)}{1 + y_{p2}^2}$$

A three-layer neural network with 20 nodes in the first hidden layer and 10 nodes in the second hidden layer was used to approximate the unknown

function $f = [f_1, f_2]^T$. The input to the neural network is a four-dimensional vector, and the output at any time instant is in $\mathbb{R}^2$.

Figure 15 shows the outputs of the plant and the identification model when identification is performed using random input signals, where the values of both $u_1(t)$ and $u_2(t)$ are uniformly distributed over the interval $[-1, 1]$. The gradient method used a step size of $\eta = 0.25$ and was terminated after 50,000 steps. The test inputs used to generate the responses shown in Figure 15 are $u_1(= \sin(2\pi k/250)$ and $u_2 = \cos(2\pi k/250)$.

Comment. The reader would have realized by now that the use of more general identification models necessitates the use of dynamic backpropagation for control purposes. Since the latter is known to be a tedious process involving a sensitivity model for each of the control parameters adjusted, it naturally follows that the effort expended to control a multivariable system of the type discussed in this example can be substantial. This in turn stresses the practical importance of the comments made earlier regarding the use of simpler models for identification purposes.

MULTILAYER NETWORKS WITH LINEAR COMPONENTS

As pointed out in [1], the systems generated by the procedures outlined earlier can be considered as generalized neural networks. They consist of the usual linear operations of summation, multiplication by a scalar, and delay, along with a simple nonlinear function (e.g., the sigmoid function) that is known. Further, the multilayer neural networks used for identification and control purposes, by Narendra and Parthasarathy [1, 2] and Narendra and Mukhopadhyay [8] as well as in this paper, are nonlinear maps in which linear terms are not explicitly included. However, identification of linear systems using such networks has shown that the approximations are quite satisfactory. Hence, over compact sets in the input space, such multilayer neural networks

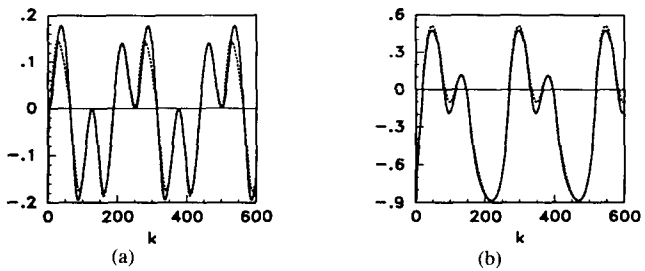

Figure 15. Identification of MIMO system using random inputs. (a) y_{p1} and $\hat{y}_{p1}$ (b) y_{p2} and $\hat{y}_{p2}$.

may be adequate to represent linear plants and controllers. However, there are two compelling reasons for including linear terms in such networks.

Most of the studies reported in [1] as well as in this paper are motivated by the results in classical adaptive control theory. The latter deals with time-invariant plants with unknown parameters for which linear controllers have to be designed. The explicit inclusion of linear terms in the multilayer neural network, we believe, would provide a smooth transition from the well-developed adaptive control theory for linear time-invariant systems to the adaptive control of time-invariant nonlinear systems that is attempted here using neural networks.

The second reason for using linear terms in neural networks is motivated by practical considerations. It is well known that linear control theory has found wide application in practical systems and that linear adaptive control has resulted in improved performance in many industrial applications. This implies that real-world systems can be approximated over certain ranges of the input reasonably well by using linear models. Hence, the addition of linear components to multilayer neural networks may be useful in obtaining first approximations to the processes under consideration before more accurate descriptions using nonlinear terms are attempted.

Finally, the adjustment of the parameters of the linear components of neural networks does not add significantly to the computational burden of the back-propagation procedure. In view of these above considerations, our decision is to include a linear part in all multilayer neural networks used for identification and control. The structure of such a modified network is shown in Figure 16. L represents the linear part whose output is $W^T x$, and N denotes the nonlinear multilayer neural network whose output $N(x)$ is described in [1], where $x \in \mathbb{R}^n$.

CONCLUSION

In this paper, the concepts introduced in [1] are applied to several examples in adaptive control. The simulation studies reveal that neural networks can be

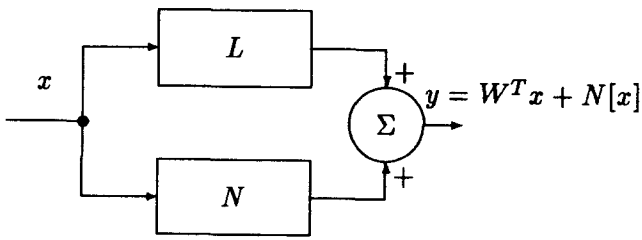

Figure 16. Modified multilayer neural network.

used effectively for the identification and control of relatively complex nonlinear systems. The principal conclusion of the paper can be expressed using the eight examples presented. Example 1 reveals that the methods suggested in [1] can be extended to systems with "relative degree" greater than 1. Example 2 reveals that when an approximate model of the plant is available, both identification and control can be carried out concurrently. In Example 3, the process of computing the inverse of a nonlinear map in a control problem is avoided by using dynamic backpropagation. A system that is not in the standard form (i.e., models I–IV) is identified using an identification model belonging to model class IV in Example 4, to emphasize the generality of the latter class. The practical difficulty of using dynamic backpropagation methods for adjusting controller parameters motivates the use of simpler representations for the unknown plant. This is discussed in Example 5. The identification and prediction of unforced nonlinear systems are discussed in Examples 6 and 7, respectively, and in Example 8 it is shown that all the methods suggested can be readily extended to multivariable systems.

While the simulation studies described above are very promising, it is also clear that theoretical investigation of nonlinear systems of the types described in this paper are still in the initial stages. This is to a great extent unavoidable owing to the complex nature of neural networks used as well as the large class of nonlinear systems that are being considered. The simulation results, however, provide the motivation for undertaking such theoretical investigations. For mathematical tractability, it may be necessary to limit ourselves to substantially more restrictive classes of nonlinear systems.

The models as well as the methods suggested here have their origins in conventional adaptive control theory, which deals with the control of linear time-invariant systems with unknown parameters. However, while the parametrizations used to represent the unknown plant and the existence of a parametrized controller to achieve the desired objective can be theoretically established in the latter case, they are merely assumed to exist in the problems investigated here. As a consequence, whereas the stability of the overall system could be established for linear systems, similar results are nowhere in sight for the general nonlinear problems discussed here. In the absence of such theory, the best that can be accomplished is to use well-established gradient methods for the dynamic adjustment of control parameters as described in Example 3.

ACKNOWLEDGMENT

The research reported here was supported by the National Science Foundation under grant ECS-8912397 and by Sandia National Laboratories under contract 86-0253.

References

1. Narendra, K. S., and Parthasarathy, K., Identification and control of dynamical systems using neural networks, *IEEE Trans. Neural Networks* **1**(1), 4–27, 1990.
2. Narendra, K. S., and Parthasarathy, K., Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Trans. Neural Networks* **2**(2), 252–262, 1991.
3. Narendra, K. S., Lin, Y. H., and Valavani, L. S., Stable adaptive controller design. Part II. Proof of stability, *IEEE Trans. Autom. Control* **25**:440–448, 1980.
4. Morse, A. S., Global stability of parameter adaptive systems, *IEEE Trans. Autom. Control* **25**, 433–439, 1980.
5. Goodwin, G. C., Ramadge, P. J., and Caines, P. E., Discrete time multivariable adaptive control, *IEEE Trans. Autom. Control* **25**, 449–456, 1980.
6. Narendra, K. S., and Lin, Y. H., Stable discrete adaptive control, *IEEE Trans. Autom. Control* **25**, 456–461, 1980.
7. Moody, J., and Darken, C. J., Fast learning in networks of locally-tuned processing units, *Neural Comput.* **1**(2), 281–294, 1989.
8. Narendra, K. S., and Mukhopadhyay, S., Associative learning in random environments using neural networks, *IEEE Trans. Neural Networks* **2**(1), 20–31, 1991.