Adaptive Control Using Neural Networks and Approximate Models

Kumpati S. Narendra, *Fellow, IEEE,* and Snehasis Mukhopadhyay

***Abstract*—The NARMA model is an exact representation of the input–output behavior of finite-dimensional nonlinear discrete-time dynamical systems in a neighborhood of the equilibrium state. However, it is not convenient for purposes of adaptive control using neural networks due to its nonlinear dependence on the control input. Hence, quite often, approximate methods are used for realizing the neural controllers to overcome computational complexity. In this paper, we introduce two classes of models which are approximations to the NARMA model, and which are linear in the control input. The latter fact substantially simplifies both the theoretical analysis as well as the practical implementation of the controller. Extensive simulation studies have shown that the neural controllers designed using the proposed approximate models perform very well, and in many cases even better than an approximate controller designed using the exact NARMA model. In view of their mathematical tractability as well as their success in simulation studies, a case is made in this paper that such approximate input–output models warrant a detailed study in their own right.**

***Index Terms*— Approximate models, approximation bounds, control, dynamic backpropagation, identification, input–output models, neural networks.**

I. Introduction

In his book *Adaptive Control Processes—A Guided Tour*, Bellman warned against "the danger of subscribing too whole-heartedly to any particular type of mathematical model ...," and went on to say "approximations after all, may be made in two places—in the construction of the model, and the solution of the associated equations." These and similar comments made by other researchers in the area of approximation theory have strongly influenced the thinking of the control community. They emphasize the importance of representing complex systems by simple models, while being aware of the limitations of the latter. It is due to such considerations that linear time-invariant systems have been studied extensively in the past four decades. The importance of simple approximate models is also becoming increasingly apparent as greater efforts are being made to identify and control nonlinear systems. In all cases, the analytical tractability of the models proposed, and the extent to which such models find application in practical contexts, have determined their acceptance by the scientific community.

From the very beginning, it has been realized by systems theorists that most real dynamical systems are nonlinear. However, linearizations of such systems around the equilibrium states yield linear models which are mathematically tractable. In particular, based on the superposition principle, the output of the system can be computed for any arbitrary input, and alternately, in control problems, the input which optimizes the output in some sense can also be determined with relative ease. In adaptive control problems, where the plant parameters are assumed to be unknown, the fact that the latter occur linearly makes the estimation procedure straightforward. While the above facts are responsible for the popularity of linear models in theoretical studies, the fact that most nonlinear systems thus far could be approximated satisfactorily by such models in their normal ranges of operation has made them attractive in practical contexts as well. It is this combined effect of ease of analysis and practical applicability that accounts for the great success of linear models and has made them the subject of intensive study for over four decades.

In recent years, a rapidly advancing technology and a competitive market have required systems to operate in many cases in regions in the state space where linear approximations are no longer satisfactory. To cope with such nonlinear problems, research has been underway on their identification and control using neural networks based entirely on measured inputs and outputs. It is now known that under certain conditions, an exact input–output representation of the nonlinear system is given by the nonlinear autoregressive moving average (NARMA) model in a neighborhood of the equilibrium state. Even assuming that such a model is available (or has been identified using neural networks), determining the control input that results in a desired output is no longer a simple task since the output depends nonlinearly on the input. As a consequence, various approximate methods have been proposed in the literature for the determination of the control input. In view of this, we introduce in this paper two approximate input–output models derived from the NARMA model, in which the control input appears linearly. The latter makes the control problem tractable (as in the linear case), since $u(k)$ is merely the ratio of two time functions. These approximate models have been applied at Yale University, New Haven, CT, to a variety of control problems and the results have been compared with those obtained using linear identification models as well as the NARMA model. In spite of the fact that the NARMA model is an exact representation of the system, the approximate

Manuscript received January 25, 1995; revised October 10, 1996. This work was supported by NSF Grant ECS-9521405 and NSF CAREER Grant ECS-9623961.

K. S. Narendra is with the Center for Systems Science, Yale University, New Haven, CT 06520 USA.

S. Mukopadhyay is with the Department of Computer and Information Science, Purdue School of Science at Indianapolis, Indianapolis, IN 46202 USA.

Publisher Item Identifier S 1045-9227(97)02754-9.

nonlinear models are found to be at least as good as the exact model for control purposes (which is quite often much better than a linear model). This can be attributed to the fact that only approximate methods are used in practice for controlling a plant represented by a NARMA model. These empirical results merely serve to underscore the words of Bellman quoted earlier.

In view of their mathematical tractability, as well as the fact that they have performed very well in numerous nonlinear control problems, the authors believe that such approximate input–output models deserve to be investigated in detail in their own right.

II. Input–Output Models

The representation of dynamical systems using state equations is currently well known. Let a system Σ be represented by the state equations

$$\Sigma: \quad \begin{aligned} x(k+1) &= f[x(k), u(k)] \\ y(k) &= h[x(k)] \end{aligned} \tag{1}$$

where $\{u(k)\}$, $\{x(k)\}$, and $\{y(k)\}$ are discrete time sequences with $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}$, $y(k) \in \mathbb{R}$, $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$, $h : \mathbb{R}^n \to \mathbb{R}$, and $f, h \in C^\infty$. It is assumed that $f(0,0) = 0$, so that the origin is an equilibrium state of (1). The linearization Σ_L of Σ is described by the linear state equations

$$\Sigma_L: \quad \begin{aligned} x(k+1) &= Ax(k) + bu(k) \\ y(k) &= cx(k) \end{aligned} \tag{2}$$

where the $(n \times n)$ matrix A and the $(n \times 1)$ and $(1 \times n)$ vectors b and c are defined by

$$\left.\frac{\partial f(x,u)}{\partial x}\right|_{0,0} = A \quad \left.\frac{\partial f(x,u)}{\partial u}\right|_{0,0} = b \quad \left.\frac{\partial h(x)}{\partial x}\right|_{0} = c. \tag{3}$$

The problems of control related to system (1) can be stated at various levels of generality. These include the cases when we have the following:

1) f and h are known, and the state $x(k)$ is accessible;
2) f and h are unknown, but the state $x(k)$ is accessible;
3) f and h are unknown, and only the input $u(k)$ and output $y(k)$ are accessible.

In 1) the system is assumed to be completely known while 2) corresponds to an adaptive control problem where f and h have to be estimated before control is effected. However, the fact that $x(k)$ is accessible makes the problem substantially simpler. 3) corresponds to the case where system identification and control have to be carried out using only input–output data. Our primary objective in this paper is to suggest approximate methods using neural networks which are computationally very effective for dealing with control problems that arise in the third case.

A. Statement of the Problem

The problems of nonlinear control in cases 1)–3) can be stated in terms of a) stabilization; b) set-point regulation; c) tracking the output of an unforced system; and d) tracking a general output sequence $y^*(k)$ [1]. Among these, we shall consider the most general one represented by d).

Given the nonlinear system Σ defined by (1), a neighborhood $\Phi \subset \mathcal{Y}$ where $\mathcal{Y}$ is the output space of Σ, and an arbitrary sequence $\{y^*(k)\}$ such that $y^*(k) \in \Phi$, determine an input sequence $\{u^*(k)\}$ such that

$$\lim_{k\to\infty} |y(k) - y^*(k)| = 0$$

and $y(k) \in \Phi$.

Determination of the conditions under which a solution exists forms part of the theoretical control problem. This problem has been treated in earlier papers [1], [2], but for the sake of continuity, we shall state in what follows the principal concepts involved in obtaining a theoretical solution. Following this, we shall state the practical methods that are available to the designer (as well as the conditions under which they are applicable) to determine the input sequence $\{u^*(k)\}$ which will achieve the control objective. The principal aim of this paper is to present an approximate method for computing the input sequence to the plant which, in practice, is much less computationally intensive than current methods that are available for solving the control problem.

The various steps used in deriving the solution to the control problem are as follows: From (1) we have

$$\begin{aligned} y(k) &= h[x(k)] \triangleq \Psi_1[x(k)] \\ y(k+1) &= h[f(x(k), u(k))] \triangleq \Psi_2[x(k), u(k)] \\ &\cdots \quad \cdots \quad \cdots \\ y(k+n-1) &= h \circ f^{n-1}[.,.] \triangleq \Psi_n[x(k), \\ u(k), u(k+1), \cdots, u(k+n-2)] \end{aligned} \tag{4}$$

where $f^{n-1}[.,.]$ is an $(n-1)$-times iterated composition of f. Denoting the sequence $y(k), y(k+1), \cdots, y(k+n-1)$ by $Y_n(k)$ and the sequence $u(k), u(k+1), \cdots, u(k+n-2)$ by $U_{n-1}(k)$, (4) can be expressed as

$$\Psi[x(k), U_{n-1}(k)] = Y_n(k).$$

If $\frac{\partial Y_n(k)}{\partial x(k)}$ (i.e., the Jacobian of Ψ with respect to x) is nonsingular at $x = 0, U_{n-1} = 0$, it follows from the implicit function theorem that $x(k)$ can be expressed as

$$x(k) = \bar{g}[Y_n(k), U_{n-1}(k)] \tag{5}$$

(where $\bar{g} : \mathbb{R}^n \times \mathbb{R}^{n-1} \to \mathbb{R}^n$ is a smooth function) in a neighborhood of the equilibrium state $x = 0, U_{n-1} = 0$. Since, by definition, the state $x(k+n)$ depends only on the state $x(k)$ and the input sequence $U_n(k) = [u(k), u(k+1), u(k+n-1)]$, it follows from (5) that

$$\begin{aligned} x(k+n) &= g[Y_n(k), U_n(k)] \\ &= g[y(k), y(k+1), \cdots, y(k+n-1) \\ &\qquad u(k), u(k+1), u(k+n-1)] \end{aligned}$$

where $g : \mathbb{R}^{2n} \to \mathbb{R}$ is a smooth function. Since $y(k+n) = h[x(k+n)]$ this leads to the NARMA model discussed in [3] and [2] and can be expressed as

$$\begin{aligned} y(k+1) = F[&y(k), y(k-1), \cdots, y(k-n+1) \\ &u(k), u(k-1), \cdots, u(k-n+1)]. \end{aligned}$$

Relative Degree: If

$$\frac{\partial(h \circ f_e^k \circ f(x,u))}{\partial u} = 0 \quad 0 \le k \le d-2$$
$$\neq 0 \quad k = d-1$$

in a neighborhood Ω of the equilibrium state, where f_e denotes the dynamics $f(.,.)$ and f_e^k is the k-times iterated composition of f, the dynamical system is said to have a relative degree d [4], [1]. Qualitatively, it implies that an input at instant k with the initial condition $x(k) \in \Omega$ (i.e., in a neighborhood of the equilibrium state) affects the output only d units of time later. Consequently, when the relative degree is defined, it represents the delay of the system (1) from input u to output y. When the system (1) has a relative degree d, it can be shown that the input–output representation of the system is given by

$$y(k+d) = \bar{F}[y(k), y(k-1), \cdots, y(k-n+1) \\ u(k), u(k-1), \cdots, u(k-n+1)]. \quad (6)$$

Equation (6) is what is needed to identify the given system as well as to control it.

Identification Using Neural Networks: Since the function $\bar{F}[.]$ is known to exist in a neighborhood of the equilibrium state, a multilayer neural network (MNN) or a radial basis function network (RBFN) can be used to identify it. Denoting the network mapping by $N[.]$, the identification model has the form

$$\hat{y}(k+d) = N[y(k), y(k-1), \cdots, y(k-n+1) \\ u(k), u(k-1), \cdots, u(k-n+1)]$$

where $\hat{y}(k+d)$ is the estimate of $y(k+d)$. Identification is carried out at every instant k by adjusting the parameters of the neural network using the error $\hat{y}(k) - y(k) \triangleq e_i(k)$ and $2n$ inputs to the neural network which are $y(k-d), \cdots, y(k-d-n+1)$ and $u(k-d), \cdots, u(k-d-n+1)$, respectively. Since static training methods for neural networks are well-known, we shall not elaborate further on the manner in which the identification is carried out.

The Control Problem: Even though in practice the map $\bar{F}$ is approximated by a neural network, we can assume in the following discussions (for convenience) that $\bar{F}$ is known. In (6) relating inputs to outputs the control input $u(k)$ at instant k has to be chosen so that $y(k+d) = y^*(k+d)$. Since the relative degree d exists, the system is necessarily controllable and $\frac{\partial \bar{F}}{\partial u(k)} \neq 0$ along the trajectory. Hence, by the implicit function theorem

$$u(k) = G[y(k), \cdots, y(k-n+1), y^*(k+d) \\ u(k-1), \cdots, u(k-n+1)]$$

where $G : \mathbb{R}^{2n} \to \mathbb{R}$ and $G \in C^\infty$ exists. The control problem is consequently to determine (or estimate) the map G from the measured values of inputs and outputs as well as the given reference output $y^*(k+d)$. Since the controller is in the feedback loop of a dynamical system, the adjustments of the parameters of a neural network used to approximate G cannot be carried out using static back propagation. Dynamic gradient methods (dynamic back propagation [6], backpropagation through time [7], and recurrent learning [8]) are all quite slow and computationally intensive as compared to their static counterparts for the computation of $u(k)$. In practice approximate methods are generally used to determine the gradient and adjust the parameters of the controller neural network. It is to overcome this problem of computational complexity in a control problem that approximate models are introduced in this paper. The exact model given by (6) is approximated by the models suggested in the following section which permit exact control laws to be generated algebraically for the problem. Theoretical justification of the models proposed and demonstrating through simulation studies that they perform comparably to approximate solutions using exact models, are the main contributions of this paper.

III. Approximation of the NARMA Model

As shown in Section II the NARMA model is an exact representation of the nonlinear plant in a neighborhood of the equilibrium state. For reasons given toward the end of Section II the model is not convenient for the computation of a control input to the plant to track a desired reference signal. In view of this, we propose two approximations to the NARMA model called the NARMA-L1 and the NARMA-L2 models. The main feature of these models is that the control input $u(k)$ at time k (the instant of interest in the control problem) occurs linearly in the equation relating inputs and outputs. This, in turn, permits easy algebraic computation of the control inputs without requiring a separate controller neural network. The fact that the use of neural networks is restricted to the identification model implies that only static gradient methods need to be used. The equations for the two proposed approximate models are given below.

NARMA-L1 Model:

$$y(k+d) \\ = f_0[y(k), y(k-1), \cdots, y(k-n+1)] \\ + \sum_{i=0}^{n-1} g_i[y(k), y(k-1), \cdots, y(k-n+1)]u(k-i). \quad (7)$$

NARMA-L2 Model:

$$y(k+d) \\ = \bar{f}_0[y(k), \cdots, y(k-n+1), u(k-1), \cdots, u(k-n+1)] \\ + \bar{g}_0[y(k), \cdots, y(k-n+1), u(k-1), \cdots \\ u(k-n+1)]u(k). \quad (8)$$

It is seen that f_0 and g_i $(i = 0, \cdots, n-1)$ in the equation describing NARMA-L1 are only functions of the past values of the outputs, and $u(k-1), \cdots, u(k-n+1)$ as well as $u(k)$ occur linearly on the right-hand side (RHS) of (7). In contrast to this, NARMA-L2 model is described by only two terms in the RHS of (8) where both $\bar{f}_0$ and $\bar{g}_0$ are functions of $y(k), \cdots, y(k-n+1)$ and $u(k-1), \cdots, u(k-n+1)$.

In the rest of this section as well as in the following section we justify the use of these two models in identification and control problems. The questions of interest are concerned with how good the approximations are, and the conditions that the

plant has to satisfy so that these models can be used with confidence in the control problem.

It is clear from (7) and (8) that the RHS of these equations describing the NARMA-L1 and NARMA-L2 models are different Taylor expansions of the RHS of the NARMA model. In view of its importance in the discussions to follow, a relatively detailed account of the principal results of Taylor's theorem is given in the appendix. In this section, we merely use the appropriate results wherever necessary.

NARMA-L1: In this case the function $\bar{F}$ in (6) is expanded in Taylor's series around $(y(k), y(k-1), \cdots, y(k-n+1), u(k) = 0, u(k-1) = 0, \cdots, u(k-n+1) = 0)$. For ease of exposition we shall use the notation $\bar{F}(Y, U)$ to denote the scalar function of $2n$ variables where $Y = [y(k), y(k-1), \cdots, y(k-n+1)]$ and $U = [u(k), u(k-1), \cdots, u(k-n+1)]$. We assume that $(Y, U) \in \mathcal{K}_Y \times \mathcal{K}_U$ where $\mathcal{K}_Y \times \mathcal{K}_U$ is a compact set in $\mathbb{R}^n \times \mathbb{R}^n$. From (20) and (21) in the appendix it follows that the first Taylor polynomial of $\bar{F}[Y, U]$ around $(Y, \mathbf{0})$ is given by $f_0[Y] + g[Y]^T U$ where the scalar function f_0 is given by

$$f_0[Y] = \bar{F}[y(k), y(k-1), \cdots, y(k-n+1), 0, 0, \cdots 0]$$

and $g^T = [g_0, g_1, \cdots, g_{n-1}]$ where the scalar function g_i is

$$g_i = \left.\frac{\partial \bar{F}}{\partial u(k-i)}\right|_{(y(k), y(k-1), \cdots, y(k-n+1), 0, 0, \cdots, 0).}$$

The remainder $R_1[Y, U]$ in this case is given by

$$R_1[Y, U] = \bar{F}[Y, U] - [f_0[Y] + g[Y]^T U]$$

and, as given in the Appendix, is bounded by

$$R_1[Y, U] \leq \frac{M_1 ||U||^2}{2} \tag{9}$$

where M_1 is the maximum matrix norm of the Hessian matrix $\frac{\partial}{\partial U}(\frac{\partial f}{\partial U})^T$ when evaluated over the compact domain $\mathcal{K}_Y \times \mathcal{K}_U \subset \mathbb{R}^n \times \mathbb{R}^n$.

Since M_1 is bounded, it follows from the inequality (9) that in the domain $\mathcal{K}_Y \times \mathcal{K}_U$, the approximation can be made as accurate as desired by the choice of $\mathcal{K}_U$. For example, if the identification error is to be less than ϵ, $|u(k)|$ must be chosen less than δ for all k where $\delta \leq \sqrt{\frac{2\epsilon}{M_1 n}}$. In practice, the magnitude of δ can be large if the elements of the Hessian matrix $\frac{\partial}{\partial U}(\frac{\partial f}{\partial U})^T$ are small.

NARMA-L2: In this case the approximation is made by using a Taylor expansion of $\bar{F}[y(k), y(k-1), \cdots, y(k-n+1), u(k), u(k-1), \cdots, u(k-n+1)]$ around the scalar $u(k)$. Denoting by W the vector $[y(k), y(k-1), \cdots, y(k-n+1), u(k-1), \cdots, u(k-n+1)]$ and expressing $\bar{F}[Y, U] = \bar{\bar{F}}[W, u(k)]$, (18) and (19) from the appendix can be used directly to derive the first Taylor polynomial $P_1[W, u]$ as

$$P_1[W, u] = \bar{\bar{F}}(W, 0) + \left.\frac{\partial \bar{\bar{F}}}{\partial W}\right|_{(W,0)} u(k).$$

The remainder $\bar{\bar{F}}[W, u] - P_1[W, u] \triangleq R_2[W, u]$ is bounded by

$$R_2[W, u] \leq \frac{M_2 u^2}{2} \tag{10}$$

where M_2 is the maximum value of $\frac{\partial^2}{\partial u^2}\bar{\bar{F}}[W, u]$. Once again, since a continuous function attains a maximum in a compact set, the value of M_2 is bounded and hence the approximation can be made as accurate as desired by decreasing the amplitude of the input u. This magnitude of u can be large in practice if the magnitude of $\frac{\partial^2}{\partial u^2}\bar{\bar{F}}[W, u]$ is small.

Comment: It is evident that the Taylor expansion used to derive NARMA-L2 is around $[W, 0]$ and consequently, the model is theoretically more general than NARMA-L1. However, in practice, the amplitude bound on the input $u(k)$ is specified for all k, and hence, in effect, the regions of validity of the two models may be the same.

Approximation Using Neural Networks: If the NARMA-L1 model is used to approximate a given plant, $(n+1)$ networks are needed to approximate the functions f_0 and $g_i\,(i = 0, 1, \cdots, n-1)$. Each of the functions is scalar-valued and has n arguments $y(k), y(k-1), \cdots, y(k-n+1)$. In contrast to this the NARMA-L2 model requires only two neural networks to approximate the functions $\bar{f}_0$ and $\bar{g}_0$. Each of the functions, however, has $(2n-1)$ inputs. From a practical stand-point, the NARMA-L2 model is found to be simpler to realize than the NARMA-L1 model. However, the NARMA-L1 model may be more amenable to theoretical analysis than the NARMA-L2 model.

Parallel and Series-Parallel Models: It is well known that system identification can be carried out using either parallel models or series-parallel models [10]. In general, most identification schemes use the series-parallel model. It is this fact that makes the approximate NARMA-L1 and NARMA-L2 models proposed in this paper applicable to both identification and control. If $\hat{y}(k)$ is an estimate of $y(k)$, the parallel NARMA model is described by the equation

$$\begin{aligned}\hat{y}(k+d) = &\hat{\bar{F}}[\hat{y}(k), \hat{y}(k-1), \cdots, \hat{y}(k-n+1)\\ &u(k), u(k-1), \cdots, u(k-n+1)].\end{aligned}$$

In contrast to this, the series-parallel model utilizes the plant output to obtain the estimate at time $k+d$. It is represented by an equation of the form

$$\begin{aligned}\hat{y}(k+d) = &\hat{\bar{F}}[y(k), y(k-1), \cdots, y(k-n+1)\\ &u(k), u(k-1), \cdots, u(k-n+1)].\end{aligned}$$

Strictly speaking, only the parallel model is a true model of the system since it can predict the output, given only the input to the system. The series-parallel model is used to predict future values of the output using current values of inputs and outputs. Since, at every instant k, the delayed inputs and outputs in the series-parallel model are known, the bound on the error can be computed from instant to instant using the expansions for R_1 and R_2 given earlier in (9) and (10). The bound is determined by the remainder in the Taylor expansions given earlier. In contrast to this, the parallel model is a dynamic feedback system and hence the estimate at any instant depends upon the estimate at the previous instants. Further, since the stability of the feedback loop has not been demonstrated, it cannot be assumed that the estimates will remain bounded.

A. Examples

The ideas presented in this section are best illustrated by considering specific examples. In this section we consider two examples of increasing complexity in which a nonlinear dynamical system is identified using the NARMA model and the NARMA-L1 and NARMA-L2 models. For comparison purposes, identification using a linear model is also included.

In the following section, the control problem is discussed using the approximate models proposed. The same two examples are also used in that section to illustrate the behavior of the approximate models in a control context.

Example 1—Identification of a First-Order Plant: The plant is described by the equation

$$y(k+1) = \sin[y(k)] + u(k) * (5 + \cos[y(k) * u(k)]). \quad (11)$$

Three different series-parallel models (linear, NARMA-L2, and NARMA) were used simultaneously to identify the given plant using random inputs. The models are described by the equations

$$\begin{aligned} \text{Linear:} \quad & \hat{y}_1(k+1) = \alpha y(k) + \beta u(k) \\ \text{NARMA-L2:} \quad & \hat{y}_2(k+1) = N_1[y(k)] + N_2[y(k)]u(k) \\ \text{NARMA:} \quad & \hat{y}_3(k+1) = N_3[y(k), u(k)] \end{aligned} \quad (12)$$

(note: for this example the NARMA-L1 and NARMA-L2 models are the same) where $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ are the outputs predicted by the linear, NARMA-L2, and NARMA models, respectively. α and β are the scalar adjustable parameters of the linear model and N_1, N_2, and N_3 are multilayer neural networks used in the NARMA-L1 and NARMA-L2 models. In the simulation studies, N_1 and N_2 were multilayer neural networks chosen from the class $\mathcal{N}^3_{2,20,10,1}$ (i.e., three-layer networks with two inputs, including a bias input held constant at one, 20 nodes in the first hidden layer, ten nodes in the second hidden layer, and one output node). Similarly, N_3 was chosen to belong to $\mathcal{N}^3_{3,20,10,1}$.

The identification of the plant was carried out using the three models and a random input sequence $\{u(k)\}$ with $|u(k)| \leq 1$. Three different sets of experiments were carried out with $|u(k)| \leq 0.1, 0.5$, and 1 respectively to study the relative performance of the models with increasing amplitudes of input and output. The parameters α and β of the linear model were adjusted using the least-squares method while the parameters of the networks N_1, N_2, and N_3 were adjusted using static backpropagation. The results are shown in Figs. 1(a)–(c).

In all three cases described above, the adjustment of the parameters was stopped after 499 700 steps and the models were then tested over the following 300 steps. Fig. 1(a) shows the errors when $|u(k)| \leq 0.1$. Even in this case, the error with the linear model is found to be significant. Both NARMA and NARMA-L2 models result in errors very close to zero. Fig. 1(b) and 1(c) depicts the output errors when $|u(k)| \leq 0.5$ and $|u(k)| \leq 1$, respectively. As expected, in all cases the linear model is significantly worse than the nonlinear models. A slight improvement in identification using the NARMA model over that obtained by the NARMA-L2 model is evident only in the third case.

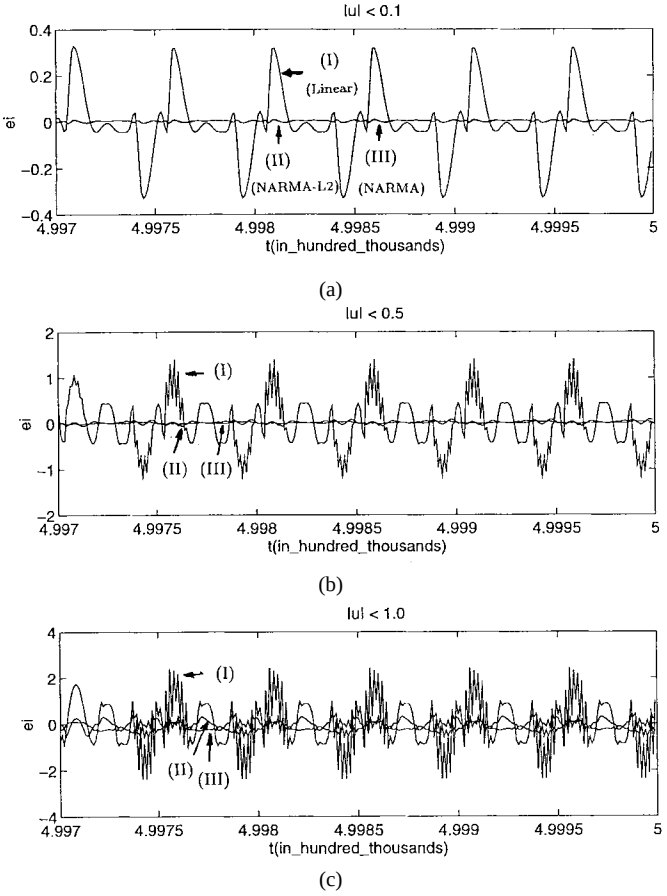

Fig. 1. Example 1: Identification errors for a first-order plant for different bounds on input amplitude: (a) plant input range $[-0.1, 0.1]$, (b) plant input range $[-0.5, 0.5]$, and (c) plant input range $[-1, 1]$. In all cases, (I): linear model, (II): NARMA-L2 model, and (III): NARMA model.

To further illustrate how the outputs predicted by the various models compare with the actual plant output, we consider the third case (i.e., $|u| < 1.0$) in more detail in Fig. 2. In Fig. 2(a), the actual plant output with the test plant input is shown. The output clearly shows very nonlinear behavior. Fig. 2(b) the model output and the identification error are shown when the linear model is used. The linear model, as expected, is totally inadequate to predict the plant output. As evident from Fig. 2(b) and (c), both NARMA-L2 and NARMA models resulted in model outputs that are very close to the actual plant output. The identification error with the NARMA model was only marginally less than that with the NARMA-L2 model.

Example 2—Identification of a Second-Order Plant: In the second example a more complex second-order plant was chosen for identification. The plant in this case is described by the difference equations

$$\begin{aligned} x_1(k+1) &= 0.1 * x_1(k) + 2\frac{u(k) + x_2(k)}{1 + (u(k) + x_2(k))^2} \quad (13) \\ x_2(k+1) &= 0.1 * x_2(k) + u(k)\left(2 + \frac{u^2(k)}{1 + x_1^2(k) + x_2^2(k)}\right) \\ y(k) &= x_1(k) + x_2(k). \end{aligned}$$

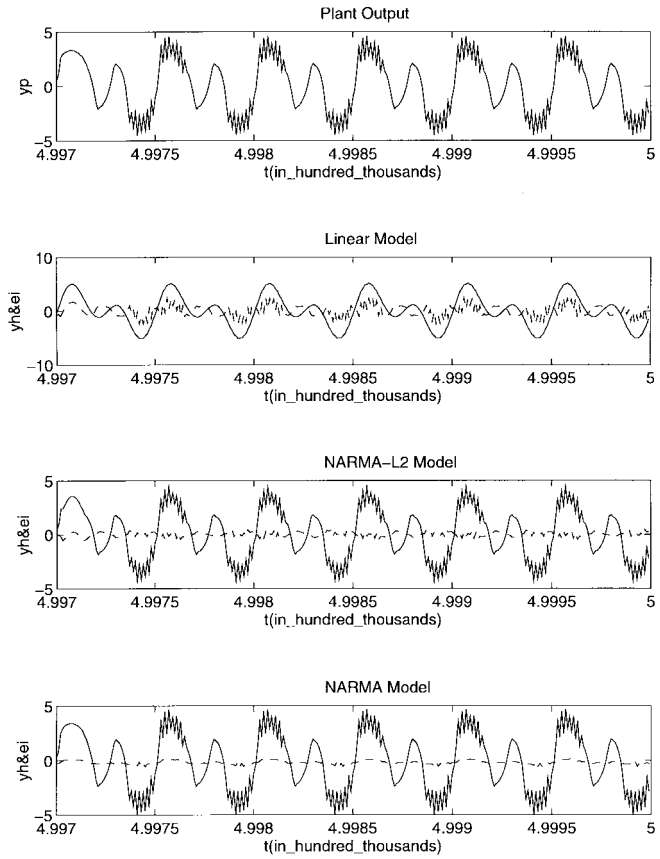

Fig. 2. Example 1: Plant and model outputs with plant input in the range $[-1, 1]$. In all of the plots, solid: model output and dashed: identification error.

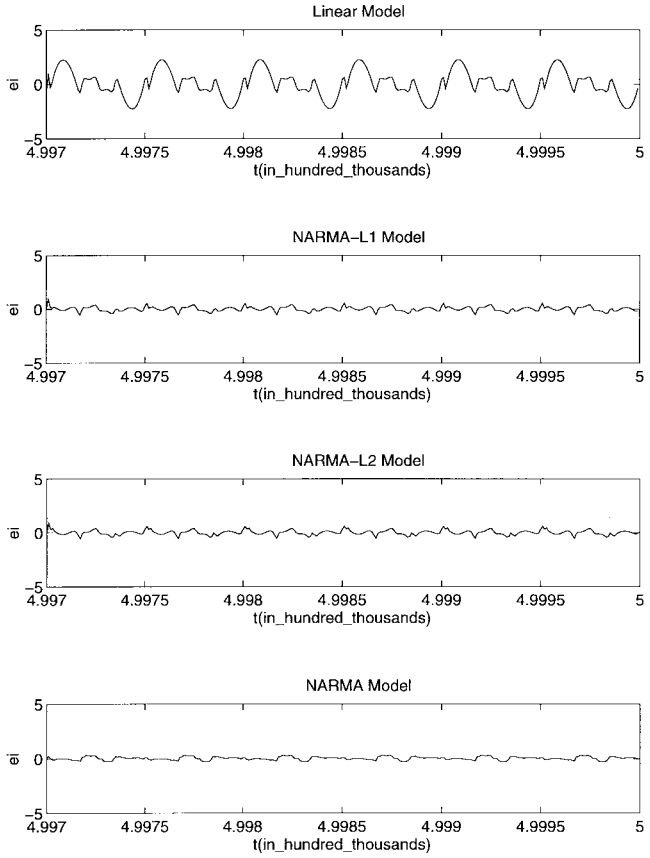

Fig. 3. Example 2: Identification errors for a second-order plant using different input–output models.

It is assumed that the state variables are not accessible and that the identification has to be carried out using only input–output data. As in Example 1 the system was identified using different models. Since the NARMA-L1 and NARMA-L2 models are different for this problem, four models were used. The equations describing each model are given below:

$$\begin{aligned}
\text{Linear}: \hat{y}(k+1) &= \hat{\theta}_1 y(k) + \hat{\theta}_2 y(k-1) \\
&\quad + \hat{\theta}_3 u(k) + \hat{\theta}_4 u(k-1) \\
\text{NARMA}: \hat{y}(k+1) &= N[y(k), y(k-1), u(k), u(k-1)] \\
\text{NARMA-L1}: \hat{y}(k+1) &= N_1[y(k), y(k-1)] \\
&\quad + N_2[y(k), y(k-1)]u(k) \\
&\quad + N_3[y(k), y(k-1)]u(k-1) \\
\text{NARMA-L2}: \hat{y}(k+1) &= N_4[y(k), y(k-1), u(k-1)] \\
&\quad + N_5[y(k), y(k-1), u(k-1)]u(k).
\end{aligned}$$

A random input with amplitude uniformly distributed over the interval $[-2, 2]$ was used for the identification process which was carried out in all cases over 500 000 time steps. Following this the models were tested using an input signal

$$u(k) = \sin\frac{2\pi k}{10} + \sin\frac{2\pi k}{25}.$$

The plant output oscillated between -5 and 5 in response to this input. The resulting identification errors due to the various models are shown in Fig. 3. Four figures are presented to illustrate the performances of the various nonlinear models relative to that of the linear model. It is evident from Fig. 3(a) that the linear model results in large identification errors (maximum magnitude of about 2.4). The error using the NARMA model [shown in Fig. 3(d)] is small (maximum magnitude of about 0.4). As expected, the NARMA model resulted in the minimum rms identification error. However, the approximate nonlinear models (NARMA-L1 and NARMA-L2) were very close in performance to the NARMA model and were far superior to the linear model. Fig. 3(b) shows that the identification error with NARMA-L1 (maximum magnitude of 0.6) was comparable to that with NARMA. The similarity in performance between NARMA-L2 and NARMA-L1 is also apparent from Fig. 3(b) and (c).

The two examples given in this section are typical of many studies carried out on low order nonlinear systems. In all cases, when the linear model performs poorly, the NARMA-L1 and L2 models yield performances comparable to the NARMA model. In the following section we consider the control problem of the same plant using the different models.

IV. Adaptive Control Using NARMA-L1 Model

As stated in Section II-A, our principal objective is to determine a control input $u^*(k)$ which will result in the output $y(k)$ of the plant (1) tracking a specified sequence $\{y^*(k)\}$ sufficiently accurately. From the earlier theoretical discussions, as well as the examples given, it is evident that if identification is our only objective, the NARMA model is superior to all

other models considered. However, as shown in this section, the approximate models NARMA-L1 and NARMA-L2 may be preferable for the control problem.

Adaptive Control: Before proceeding further, a few comments concerning the control problem are in order. If the plant is described by the state equations (1) where f and h are known and the initial state $x(0)$ is known, we have a control problem in which the state variables are known at every instant of time. If f and h are unknown and the plant has to be controlled using only input–output data, we have an adaptive control problem. However, if the plant is (bounded-input bounded-output) stable, it can be first identified using a model and the latter can then be used for control. Strictly speaking, (in the terminology of linear adaptive control theory) the above problem is not considered an adaptive control problem. The latter problem occurs only when the identification and control are carried out simultaneously and thereby raise questions of stability of a different kind. The adaptive control problem becomes truly complex when the plant is unstable, making it necessary to carry out the above operations rapidly enough so that the overall system is stable.

In the control of nonlinear systems using neural networks, we are interested in both classes of control problems discussed above. It is perhaps worth stating that even when f and h in (1) are known, they are generally approximated by neural network maps N_f and N_h to streamline the process of determining controllers. When f and h are unknown and the state variables are not accessible, the function $\bar{F}$ in (6) is approximated using neural networks as described in the previous section. In all cases, the investigation of stability of adaptive controllers used to control unstable plants is beyond the scope of this paper and will be considered later. In the following, we consider the feasibility of controlling stable nonlinear plants using neural networks to achieve improved tracking performance. At present, the latter is of great interest to industry.

How do the approximate models perform in a control context? Can conditions be determined that are sufficient for a solution to the control problem to exist? Can these conditions be deduced from the properties of the linearized system Σ_L? Assuming that a solution exists, is the overall controlled plant stable? Robust?—these are some of the questions that require our attention. Such questions have been addressed elsewhere in the context of control using the NARMA model [2], [1]. In this section we indicate how the arguments have to be modified when the NARMA-L1 and NARMA-L2 models are used.

Zero Dynamics: Given a nonlinear dynamical system with a well defined relative degree d, it has been shown [4], [1] that the state equations (1) can be transformed into the normal form given below so that the output of the system is the same as the first state variable $z_1(k)$, i.e.,

$$\begin{aligned} z_1(k+1) &= z_2(k) \\ z_2(k+1) &= z_3(k) \\ &\cdots \cdots \quad \cdots \\ z_{d-1}(k+1) &= z_d(k) \\ z_d(k+1) &= f_1[\overline{z(k)}, \eta(k), u(k)] \\ \eta(k+1) &= D[\overline{z(k)}, \eta(k), u(k)] \end{aligned}$$

where $\overline{z(k)} = [z_1(k), \cdots, z_d(k)]^T$, $\eta(k) \in \mathbb{R}^{n-d}$ and $z(k)^T = [\overline{z(k)}^T, \eta(k)^T]$. If the output of the system is to be identically zero, i.e., $\overline{z(k)} = 0$ for $k > k_0$, it follows that

$$\begin{aligned} f_1[0, \eta(k), u(k)] &= 0 \\ \eta(k+1) &= D[0, \eta(k), u(k)]. \end{aligned}$$

For a detailed treatment of the control problem, the reader is referred to [1]. It has been shown that if a relative degree exists (i.e., $(\partial f_1/\partial u(k)) \neq 0$ in a neighborhood of the equilibrium state), then $u(k) = G_1[\eta(k)]$ where $G_1 : \mathbb{R}^{n-d} \to \mathbb{R}$ is a smooth function. This in turn yields

$$\eta(k+1) = \bar{D}[\eta(k)] \tag{14}$$

which is an unforced dynamical system of degree $n - d$. If (14), known as the zero dynamics of the system, is uniformly asymptotically stable, it follows that the stabilizing control input is bounded and tends to zero. In [1], it is further shown that the same analysis can also be used to track a desired sequence $\{y^*(k)\}$ when the amplitude of $y^*(k)$ is sufficiently small.

In summary, if the plant has a well-defined relative degree and uniformly asymptotically stable zero dynamics, it can, in theory, be controlled to track a desired output. In the following we assume that these conditions are satisfied by the plant.

Control Using NARMA, NARMA-L1, and NARMA-L2 Models: In Section II-A it was stated that if the plant is described by the NARMA model

$$\begin{aligned} y(k+d) = \bar{F}[&y(k), y(k-1), \cdots, y(k-n+1), \\ &u(k), u(k-1), \cdots, u(k-n+1)] \end{aligned}$$

there exists a controller of the form

$$\begin{aligned} u(k) = G[&y(k), y(k-1), \cdots, y(k-n+1), y^*(k+d), \\ &u(k-1), \cdots, u(k-n+1)] \end{aligned}$$

so that the output can be tracked in exactly d steps. It was further stated that determining the parameters of G (or the neural network used to approximate it) is computationally intensive, since it has to be carried out in a feedback loop.

Let $\bar{F}$ be approximated by the NARMA-L1 model or NARMA-L2 model using (7) or (8). If g_0 in (7) and $\bar{g}_0$ in (8) are sign definite in a neighborhood of the origin, the control input to the plant can be computed readily (without backpropagation) in the same neighborhood as (a) shown at the bottom of the next page for NARMA-L1 and (b) shown at the bottom of the next page for NARMA-L2. In both cases, the control input is computed to make the output of the approximate model follow exactly the desired output $y^*(k+d)$. Also the desired input can be computed algebraically from the identification model and hence a separate controller neural network is not needed. Further, since the error between the identification model output and the plant output is determined by the residual terms which are bounded, the plant output is also bounded.

A. Examples

In this section we consider the same examples that we considered in Section III-A in which two different plants were identified using different models. The principal difference is that in the present case the control input has to be determined to track a desired output. For an example of controlling a multivariable plant using approximate models, the reader is referred to [11].

Example 3: Consider the first-order plant described in Example 1 (11). The objective in the present example is to control the plant so as to track a desired output given by

$$y^*(k) = 2\sin\left(\frac{2\pi k}{50}\right) + 2\sin\left(\frac{2\pi k}{100}\right).$$

Control was attempted using the three models: linear, NARMA-L2 (which, in the first-order case, is the same as NARMA-L1), and NARMA. The details concerning the simulation are not presented here due to space constraints. We merely present the results on the simulation. The control errors with different models are shown in Fig. 4. As in the case of identification, the linear model gives poor performance in control as well (with maximum control error of about 1.2). The NARMA-L2 and NARMA models, as expected, provided very similar control performances which were substantially better than that of the linear model. The controller design procedure with the NARMA model, however, was more complex than that with the NARMA-L2 model.

To provide further insight into the control process using the different models, we show in Fig. 5 the coefficients of $u(k)$ during the testing phase for all three models used. This appears in the denominator in the algebraic computation of the control input in the cases of the linear as well as NARMA-L2 models. For the linear model, this coefficient was a constant [estimated to have a value of 5.7143 as seen from Fig, 5(a)]. For the NARMA-L2 model, however, this was estimated by a neural network as a function of $y(k)$. In Fig. 5(b) this coefficient is shown for the NARMA-L2 model as a function of time with values computed along the trajectory using the corresponding neural network. Since the NARMA model is realized by a neural network representing a general function of $u(k)$ and $y(k)$, the coefficient was made to correspond to the partial derivative of the function with respect to $u(k)$ evaluated along the trajectory. This was easily computed by a modification of the back-propagation algorithm and is shown in Fig. 5(c). As seen from Fig. 5(b) and (c), the coefficient realized by the NARMA-L2 model resembles very closely the partial derivative of $u(k)$ in the NARMA model. The corresponding coefficient of the linear model, constrained to be a constant, cannot approximate the Jacobian sufficiently closely. This, in turn, is one of the factors explaining the large control error obtained when the linear model is used [see Fig. 4(a)].

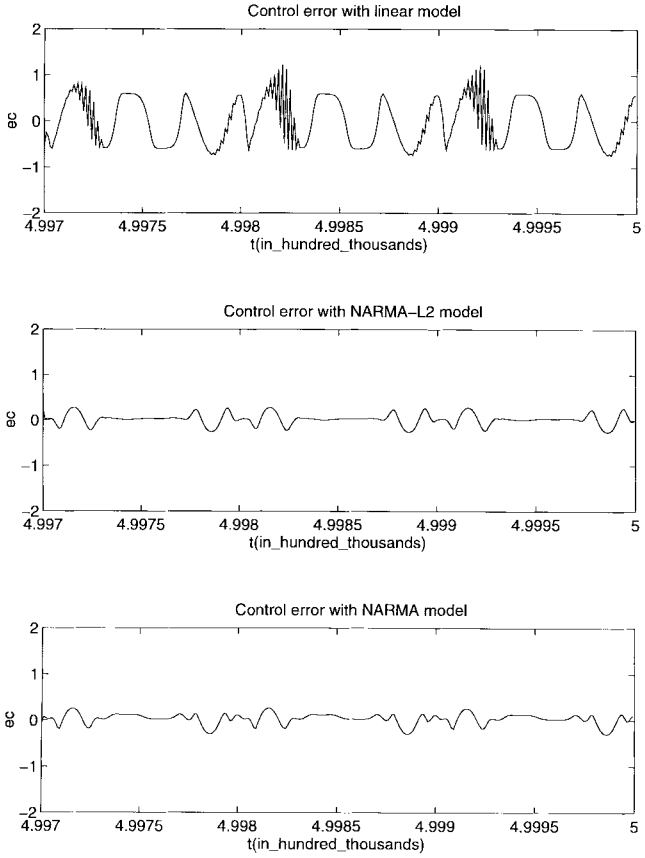

Fig. 4. Example 3: Control of a first-order plant using different input–output models: the control errors.

Example 4: Consider the second-order nonlinear plant described in Example 2 ([13]). In Example 2, this plant was identified using the linear, NARMA, as well as approximate NARMA-L1 and NARMA-L2 models. In this example, our objective is to study how the different models for a specific plant perform when used in a control problem. As mentioned earlier, it is in the control problem that the advantages of the approximate models over the NARMA model become apparent.

The procedure adopted to control the plant using the different models is described briefly below. The plant was first identified open loop with random inputs for 20 000 time steps. This was followed by closed-loop identification and control for 479 700 time steps using a random reference trajectory. The magnitude of the reference output was gradually increased (up to 8) until the linear model was found to be inadequate. After this the models and the controller (in case of NARMA model)

$$u(k) = \frac{[y^*(k+d) - \bar{f}_0[y(k),\cdots,y(k-n+1),u(k-1),\cdots,u(k-n+1)]}{\bar{g}_0[y(k),\cdots,y(k-n+1),u(k-1),\cdots,u(k-n+1)]} \quad \text{(a)}$$

$$u(k) = \frac{[y^*(k+d) - f_0[y(k),\cdots,y(k-n+1)] - \sum_{i=1}^{n-1} g_i[y(k),\cdots,y(k-n+1)]u(k-i)]}{g_0[y(k),\cdots,y(k-n+1)]} \quad \text{(b)}$$

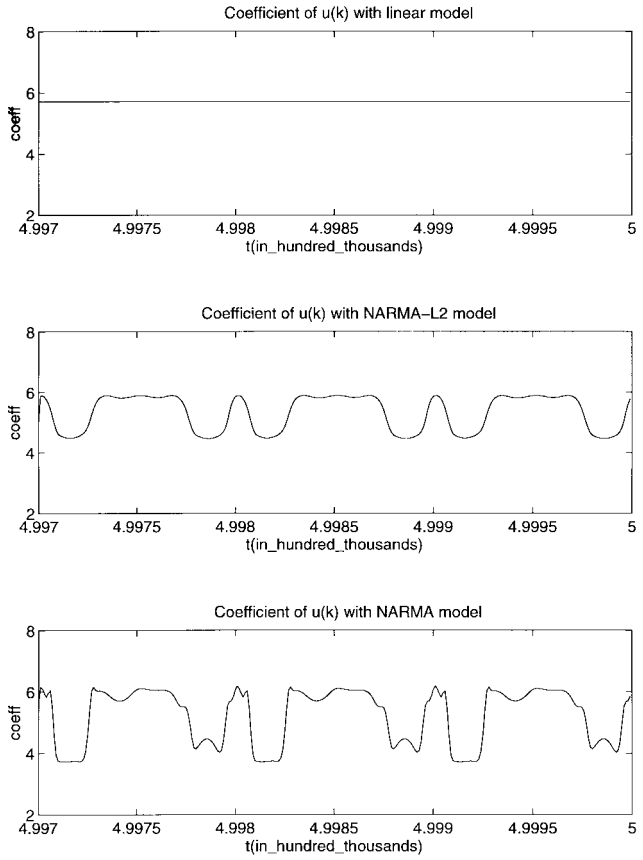

Fig. 5. Example 3: the coefficients of $u(k)$ for linear, NARMA-L2, and NARMA models.

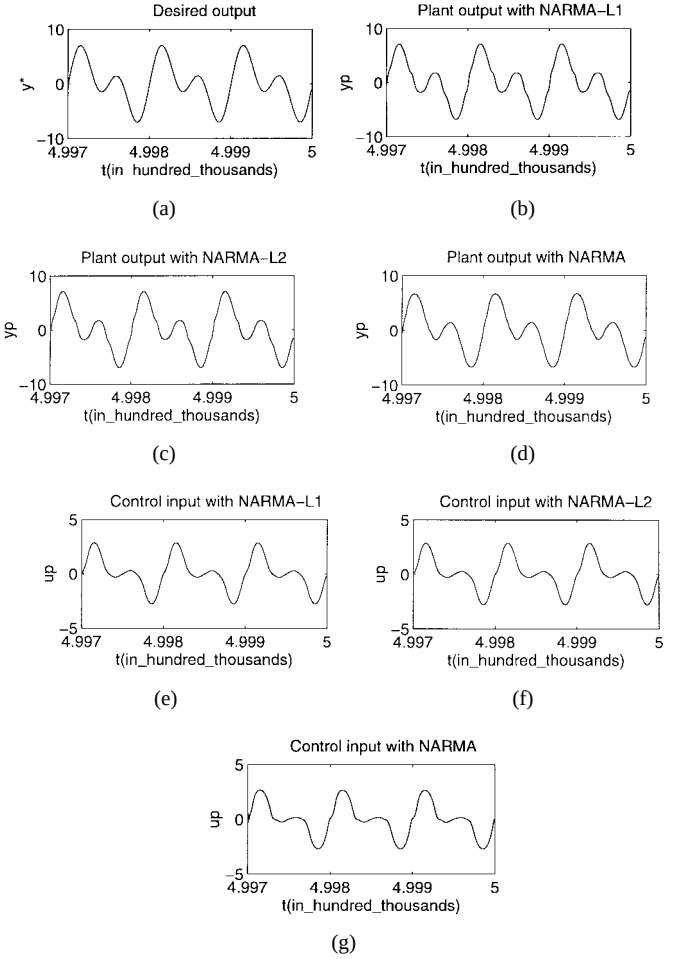

Fig. 6. Example 4: Control of a second-order plant using different input–output models.

were fixed and tested for a sinusoidal reference output given by

$$y^*(k) = 4\sin\left(\frac{2\pi k}{50}\right) + 4\sin\left(\frac{2\pi k}{100}\right).$$

In the case of linear, NARMA-L1, and NARMA-L2 models, the control inputs were computed algebraically, based on the respective identification models. In the case of the NARMA model, however, a separate controller network with suitable structure was used whose parameters were updated using an approximate gradient method.

Fig. 6 shows how the various models perform in the control problem. Fig. 6(a) shows the reference output, i.e., the desired plant output for the tracking problem. By the very choice of the magnitude of the reference output, the linear model performed poorly resulting in a large control error (maximum magnitude of about 2.5). This is not shown in Fig. 6 due to space limitations. From Fig. 6(b)–(d), it is seen that the plant outputs resulting from control using the three nonlinear models were all very close to the desired output. As mentioned earlier, the primary advantage of using the approximate NARMA models is the ease with which the control input can be determined.

Fig. 6(e)–(g) shows the control inputs resulting from the use of NARMA-L1, NARMA-L2, and NARMA models respectively. The control inputs are also observed to be very similar and the small differences in their tracking performances [in Fig. 6(b)–(d)] can be attributed to the corresponding small differences in the control inputs.

V. Comments and Conclusion

In the preceding sections the NARMA-L1 and NARMA-L2 models were introduced as approximations of the NARMA model for the representation of nonlinear dynamical systems. In Section III, the conditions under which these models may be expected to perform satisfactorily were discussed, and in Section IV simulation results were presented where the two approximate models were shown to compare favorably with the NARMA model in both identification and control contexts. The theoretical and practical advantages of the two models may now be summarized to make a case to support the authors' view that these models should be studied in their own right in the future.

The availability of a NARMA model for a nonlinear plant does not automatically imply a method of determining the control input to track a desired output. If a neural network is used as a controller, the parameters of the latter have to be adjusted to achieve on-line control. This involves dynamic gradient methods which are computationally intensive. In contrast to that, since the control input $u(k)$ occurs linearly in the NARMA-L1 and NARMA-L2 models, it can be computed directly from the identification model.

Since dynamic gradient methods are computationally intensive, it is common practice to use approximate gradient

methods (usually based on static gradients) to adjust control parameters, when a NARMA model is used to identify the system. Hence, even though the NARMA model results in better identification of the unknown plant, the NARMA-L1 and NARMA-L2 models may actually result in better control. This has been verified in simulation studies in the past in addition to the examples discussed in the paper.

Perhaps of greatest significance for the use of neural networks in the control of nonlinear dynamical systems is the fact that the NARMA-L1 and NARMA-L2 models are more tractable analytically than the NARMA model. If the stability, controllability and observability, as well as the zero dynamics of dynamical systems can be studied for the class of systems represented by these approximate models, the results can be extended to NARMA models using robustness arguments. It is believed that this approach may provide a handle for attacking the stable adaptive control problem of nonlinear plants.

This paper has merely introduced the above two classes of models. A considerable amount of work remains to be done to develop provably stable design methodologies based on such models.

APPENDIX
TAYLOR'S THEOREM

In this appendix, we summarize some results concerning Taylor expansion and analysis of functions that are extensively used in the rest of the paper. The material presented is based on [12]. In all cases, the function to be expanded will be assumed to be scalar-valued. However, depending on whether the argument is scalar or vector, different cases will be considered.

A. Taylor Expansion of $f(z)$ (z is a Scalar)

Let $f(z)$ be an infinitely differentiable scalar function of a scalar variable z. For each integer $k \geq 0$, the kth Taylor polynomial of f is defined as

$$p_k(z) = \sum_{n=0}^{k} \frac{f^n(0)}{n!} z^n$$

where $f^n(0) = \frac{d^n f(z)}{dz^n}|_{z=0}$.

The remainder of the kth Taylor polynomial is defined by

$$R_k(z) = F(z) - p_k(z).$$

Taylor's theorem states that the remainder is small for small z. In particular, the remainder $R_k(z)$ is given by

$$R_k(z) = \int_0^z f^{(k+1)}(\eta) \frac{(z-\eta)^k}{k!} d\eta.$$

When the $(k+1)$th derivative of f with respect to z is bounded in the desired domain of approximation

$$|R_k(z)| \leq \frac{M_k(z_0)}{(k+1)!}|z|^{k+1} \quad \text{for } |z| \leq z_0 \tag{15}$$

where

$$M_k(z_0) = \max\{|f^{(k+1)}(z)| : |z| \leq z_0\}.$$

When the infinite Taylor series $\sum_{n=0}^{\infty} \frac{f^n(0)}{n!} z^n$ converges to $f(z)$, then f is called analytic. For our purposes, we will not be concerned with whether the function f is analytic or even whether the infinite Taylor series converges at all. In all the analysis carried out in the paper, we will only consider a fixed partial sum with $k = 1$. The error estimate given by (15) is valid for any finite k irrespective of whether f is analytic or not.

The remainder of the Appendix discusses several cases when the scalar-valued function has many arguments. In such cases, due to notational difficulties, we present only an expansion up to $k = 1$. As stated before, only $k = 1$ expansions are used in the analysis in the body of this paper.

B. Taylor Expansion of $f(x, z)$ (x and z Scalars)

The desired expansion (with respect to only z) is given by

$$p_1(x, z) = \sum_{n=0}^{1} a_n(x) z^n \tag{16}$$

where

$$a_n(x) = \left| \frac{\partial^n}{\partial z^n} f(x, z) \right|_{z=0}.$$

When $f(x, z)$ has continuous partial derivatives up to second order with respect to z for $|z| \leq z_0$ and $|x| \leq x_0$, the remainder $R_1(x, z) = f(x, z) - p_1(x, z)$ can be expressed as

$$|R_1(x, z)| \leq M(x_0, z_0) \frac{z^2}{2} \tag{17}$$

where $M(x_0, z_0)$ is the maximum value of $|\frac{\partial 2}{\partial z^2} f(x, z)|$ for $|x| \leq x_0$ and $|z| \leq z_0$.

C. Taylor Expansion of $f(\mathbf{x}, z)$ ($\mathbf{x}$ is a Vector, and z is Scalar)

In this case the scalar-valued function $f(\mathbf{x}, z)$ depends continuously upon a vector variable $\mathbf{x} = (x_1, \cdots, x_m)$ in addition to a scalar z. However, we expand it in Taylor series only in terms of z. This is essentially case (ii) extended to the vector x case. The expansion (upto $k = 1$) will be given by

$$p_1(\mathbf{x}, z) = \sum_{n=0}^{1} a_n(\mathbf{x}) z^n \tag{18}$$

where

$$a_n(\mathbf{x}) = \left| \frac{\partial^n}{\partial z^n} f(\mathbf{x}, z) \right|_{z=0}.$$

If $\mathbf{x}$ takes values in the compact set $\mathcal{K} \subset \mathbb{R}^m$ and $|z| \leq z_0$, assuming as before that $f(\mathbf{x}, z)$ has continuous partial derivatives of second order with respect to z in those domains, the remainder $R_1(\mathbf{x}, z) = f(\mathbf{x}, z) - p_1(\mathbf{x}, z)$ can be expressed as

$$|R_1(\mathbf{x}, z)| \leq M(\mathcal{K}, z_0) \frac{z^2}{2} \tag{19}$$

where $M(\mathcal{K}, z_0)$ is the maximum value of $|\frac{\partial^2}{\partial z^2} f(\mathbf{x}, z)|$ for $\mathbf{x}$ in $\mathcal{K}$ and $|z| \leq z_0$.

D. Taylor Expansion of $f(\mathbf{x},\mathbf{z})$ (Both $\mathbf{x}$ and $\mathbf{z}$ are Vectors)

The $k=1$ expansion in this case ($\mathbf{x}\in\mathbb{R}^m$ and $\mathbf{z}\in\mathbb{R}^n$) is given by

$$p_1(\mathbf{x},\mathbf{z}) = a_0(\mathbf{x}) + \mathbf{z}^T a_1(\mathbf{x}) \tag{20}$$

where $a_0 : \mathbb{R}^m \rightarrow \mathbb{R}$ is given by $a_0(\mathbf{x}) = f(\mathbf{x},\mathbf{z})_{\mathbf{z}=\mathbf{0}}$ and $a_1 : \mathbb{R}^m \rightarrow \mathbb{R}^n$ is given by $a_1(\mathbf{x}) = \frac{\partial f(\mathbf{x},\mathbf{z})}{\partial \mathbf{z}}_{\mathbf{z}=\mathbf{0}}$ ($\frac{\partial f}{\partial \mathbf{z}}$ is the $(n\times 1)$ vector $[\frac{\partial f}{\partial z_1},\cdots,\frac{\partial f}{\partial z_n}]^T$). The remainder $R_1(\mathbf{x},\mathbf{z}) = f(\mathbf{x},\mathbf{z}) - p_1(\mathbf{x},\mathbf{z})$, as before, can be bounded above when $\mathbf{x}\in\mathcal{K}_x$ and $\mathbf{z}\in\mathcal{K}_z$ ($\mathcal{K}_x\subset\mathbb{R}^m$ and $\mathcal{K}_z\subset\mathbb{R}^n$ are compact sets)

$$R_1(\mathbf{x},\mathbf{z}) \leq M(\mathcal{K}_x,\mathcal{K}_z)\frac{\|\mathbf{z}\|^2}{2} \tag{21}$$

where $M(\mathcal{K}_x,\mathcal{K}_z)$ is the maximum matrix norm of the Hessian matrix $\frac{\partial}{\partial \mathbf{z}}(\frac{\partial f}{\partial \mathbf{z}})^T$ when evaluated over $\mathbf{x}\in\mathcal{K}_x$ and $\mathbf{z}\in\mathcal{K}_z$.

In the analysis carried out in the body of this paper, the upper bounds of the remainder is used as a performance measure of a particular model. Thus, if a model M_1 results in a smaller upper bound on the remainder than a second model M_2, it is expected that M_1 is better than M_2 (unless, of course, by accident the actual error is much less than its known upper bound).

ACKNOWLEDGMENT

The first author would like to thank his graduate student J. B. D. Cabrera with whom he has had numerous discussions on the nonlinear tracking problem. This paper has used many results from a report that the first author coauthored with Mr. Caberera.

REFERENCES

[1] J. B. D. Cabrera and K. S. Narendra, "On regulation and tracking in nonlinear discrete-time systems—Part I: State variables accessible," Center Syst. Sci., Yale Univ., New Haven, CT, Tech. Rep.

[2] A. U. Levin and K. S. Narendra, "Control of nonlinear dynamical systems using neural networks: Part II—Identification and Part III—Control," *IEEE Trans. Neural Networks*, vol. 4, Mar. 1993, also *IEEE Trans. Neural Networks*, vol. 7, Jan. 1996, pp. 30–42; also Center Syst. Sci., Yale Univ., New Haven, CT, Tech. Rep. 9116 and 9117.

[3] I. J. Leontaritis and S. A. Billings, "Input–output parametric models for nonlinear systems. Part I: Deterministic nonlinear systems," *Int. J. Contr.*, vol. 41, no. 2, pp. 303–328, 1985.

[4] H. Nijmeijer and A. J. Van der Schaft, *Nonlinear Dynamic Control Systems*. New York: Springer-Verlag, 1990.

[5] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Mar. 1990.

[6] ——, "Gradient methods for the optimization of dynamical systems containing neural networks," *IEEE Trans. Neural Networks*, vol. 2, pp. 252–262, Mar. 1991.

[7] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560 1990.

[8] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computa.*, vol. 1, no. 2, pp. 270–280, 1989.

[9] A. Isidori, *Nonlinear Control Systems*. New York: Springer-Verlag, 1985.

[10] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[11] K. S. Narendra and S. Mukhopadhyay, "Adaptive control of nonlinear multivariable systems using neural networks," *Neural Networks*, vol. 7, no. 5, pp. 737–752, 1994.

[12] J. A. Murdock, *Perturbation—Theory and Methods*. New York: Wiley, 1991.

Kumpati S. Narendra (S'55–M'60–SM'63–F'79) received the Ph.D. degree from Harvard University, Cambridge, MA, in 1959.

At present he is Professor of Electrical Engineering and Director of the Center for Systems Science at Yale University. He is the coauthor of three books, *Frequency Domain Criteria for Absolute Stability* (New York: Academic, 1973), *Stable Adaptive Systems* (Englewood Cliffs, NJ: Prentice-Hall, 1989), and *Learning Automata—An Introduction* (Englewood Cliffs, NJ: Prentice-Hall, 1989). He is also the editor of four books. His research interests include stability theory, adaptive systems, learning automata, and the control of complex systems with neural networks.

Dr. Narendra is the member of Sigma Xi and a Fellow of the American Association for the Advancement of Science and IEE. He was the recipient of the 1972 Franklin V. Taylor Award of the IEEE Systems, Man, and Cybernetics Society, the George S. Axelby Best Paper Award of the IEEE Control Systems Society in 1988, the Education Award of the American Automatic Control Council in 1990, and the Outstanding Paper Award of the Neural Networks Council in 1991. In 1994, he received the Neural Networks Leadership Award of the International Neural Networks Society. He was also appointed Distinguished Visiting Scientist by the Jet Propulsion Laboratory for the year 1994–1995. He received the honorary D.Sc. degree from Anna University, Madras, India, and the Bode Prize of the Control Systems Society in 1995. He was elected member of the Connecticut Academy of Science and Engineering in 1995.

Snehasis Mukhopadhyay received the Ph.D. degree from Yale University, New Haven, CT, in 1994.

He is currently an Assistant Professor of Computer and Information Science at Indiana University/Purdue University, Indianapolis. During 1993–1994, he was with GM NAO Research and Development Center, Warren, MI. His research interests are in neural networks and intelligent control, distributed inteconnections of neural networks, multiagent learning, and intelligent information agents.

Dr. Mukhopadhyay is a recipient of NSF CAREER Grant in 1996.