Hindawi
Modelling and Simulation in Engineering
Volume 2020, Article ID 8642915, 13 pages
https://doi.org/10.1155/2020/8642915

Research Article

Adaptive Control Using Neural Networks and Approximate Models for Nonlinear Dynamic Systems

Khadija El Hamidi,[1] **Mostafa Mjahed**,[2] **Abdeljalil El Kari**,[1] **and Hassan Ayad**[1]

[1]*Laboratory of Electric Systems and Telecommunications, Cadi-Ayyad University, Marrakesh 4000, Morocco*
[2]*Mathematics and Systems Department, Royal School of Aeronautics, Marrakesh 40000, Morocco*

Correspondence should be addressed to Khadija El Hamidi; khadija.elhamidi91@gmail.com

Received 19 August 2019; Revised 22 June 2020; Accepted 13 August 2020; Published 26 August 2020

Academic Editor: Michele Calì

In this research, a comparative study of two recurrent neural networks, nonlinear autoregressive with exogenous input (NARX) neural network and nonlinear autoregressive moving average (NARMA-L2), and a feedforward neural network (FFNN) is performed for their ability to provide adaptive control of nonlinear systems. Three dynamical nonlinear systems of different complexity are considered. The aim of this work is to make the output of the plant follow the desired reference trajectory. The problem becomes more challenging when the dynamics of the plants are assumed to be unknown, and to tackle this problem, a multilayer neural network-based approximate model is set up which will work in parallel to the plant and the control scheme. The network parameters are updated using the dynamic backpropagation (BP) algorithm.

1. Introduction

Linear control methods are based on the existence of an analytical model of the system. However, most physical systems have nonlinearity, and their mathematical model is unknown or partially known and variable in time. So, conventional methods suffer some limitation in terms of stabilization and performance [1, 2]. With the considerable development of artificial intelligence, which is found to be more suitable in handling such complex processes, the artificial neural network (ANN) is one of the powerful tools that has been recognized for tracking highly nonlinear dynamic and complex systems [2–5].

In the field of control engineering for a class of nonlinear discrete-time systems, a neural network has recently emerged as adjustable approximators capable of reproducing the complex behavior of nonlinear systems. Artificial neural networks have been effectively used as tracking controllers for unknown linear and nonlinear dynamic plants [6, 7]. ANNs have been employed in various fields, like time series prediction, system identification and control, and function approximation [8]. It has been shown that ANNS can efficiently approximate dynamics without requiring detailed knowledge of the plant [8, 9]. Another advantage of ANNs is their possibility of learning, which can reduce the human effort during the design of the controllers and allows discovering more effective control structures than those already known.

There are different types of ANNs proposed in the literature: feedforward neural networks (FFNNs), Kohonen self-organizing network, radial basis function (RBF), and recurrent neural network [10–12]. In fact, two classes of neural networks are the most popular in practical applications and have received considerable attention in the area of artificial neural networks in recent years: multilayer feedforward neural networks and recurrent networks. Multilayer feedforward neural networks have an ability to model any nonlinear function and have proven extremely successful in pattern recognition tasks [13, 14] while recurrent networks have been used in associative memories as well as for the solution of optimization problems, and it constitutes also a powerful computational tool for sequence modeling and prediction. Both of these networks have common characteristics like parallelism in their operation, generalization, and learning capability, which make them suitable candidates for the control of nonlinear systems [2]; there are compelling reasons to view them in a unified fashion.

Various structures of neural networks have been proposed in the literature. In [2], the authors have applied a diagonal recurrent neural network (DRNN) as a controller for both single-input single-output (SISO) and multiple-input multiple-output (MIMO) plants. The responses of plants obtained with DRNN are compared with those obtained when a multilayer feedforward neural network is used as a controller. In [15], recurrent neural networks have been used for providing speed control to the nonlinear motor-drive system using a model-following control scheme. In [4], an adaptive controller for a class of unknown nonlinear discrete-time systems based on a multi-input fuzzy rules emulated network (MIFREN) is introduced; the neural network is assigned to identify the unknown system under control. The authors in [3] designed a nonlinear autoregressive moving average (NARMA-L2) controller, which is based on an adaptive neurofuzzy inference system. The NARMA-L2 controller for a single-link manipulator has also been used in [16, 17]. A Lyapunov function-based neural network tracking (LNT) strategy for SISO discrete-time nonlinear dynamic systems is proposed. The proposed scheme uses two Lyapunov function neural networks operating as the controller and estimator. In [18], both the feedforward and recurrent neural network approaches are proposed, tested, and compared.

The main contribution of this paper is to propose a strong nonlinear adaptive controller of unknown nonlinear dynamical systems based on the approximate models. Another objective is to present a prescriptive method for the dynamic adjustment of the parameters based on backpropagation. The big advantage of the proposed control system is that it does not require previous knowledge of the model. Our ultimate goal is to determine the control input using only the values of the input and output.

In our research study, the strong learning capability of the dynamic neural network in identification and control is combined with the functionality of the approximate model controller structure to propose a novel online NN-based controller for nonlinear single-input single-output (SISO) dynamical systems. The main issues in the field of ANN-based control are the choice of neural network structure to be used as a controller. In our paper, an attempt has been made to compare the three types of neural networks: two recurrent neural networks, nonlinear autoregressive with exogenous input (NARX) neural network and nonlinear autoregressive and moving average (NARMA-L2), and a feedforward neural network (FFNN). The control system configuration employed in our paper consists of an ANN-based controller in cascade with the plant, and the training is performed online.

This study differentiates from the studies in the literature in terms of mathematically obtaining nonlinear SISO controller parameters. Thus, the main novelty of this paper is that the parameters of the nonlinear SISO controller can be identified as neural network expressions for nonlinear SISO systems. All neural network simulations are performed in a Matlab environment. The results indicate that the online NARMA-L2, NARX, and FFNN controllers attain good modeling and control performances.

The remainder of the paper is organized as follows: Section 2 includes a brief introduction about NARX, NARMA-L2, and FFNN models and their mathematical formulation. Section 3 includes the discussion on the adaptive control of nonlinear systems. Section 4 contains the simulation study. In Section 5, the conclusion of the paper is given.

2. Neural Network Approximate Models and Their Mathematical Formulation

Generally, there are many different neural networks (NN) of nonlinear models. In this research study, the recurrent and feedforward neural networks are presented. Since a lot of literature exists on NARX, NARMA-L2, and FFNN, in this section, a brief introduction regarding their mathematical formulation is given. Figures 1–3 show the structures of FFNN, NARX, and NARMA-L2 models, respectively.

2.1. Feedforward Neural Network Model. In a feedforward neural network, the information moves only in one direction, from the input layer to the output layer, but not vice versa as presented in Figure 1 [19]. The input vector of FFNN at any k th time instant is denoted by $X(k) = \{x_1(k), x_2(k), \cdots x_p(k)\}$. Thus, there are p numbers of inputs. The weighted sums of inputs of hidden neurons are denoted by the vector $S(k)$ where $S(k) = \{S_1(k), S_2(k), \cdots S_q(k)\}$. The output of hidden neurons is denoted by a vector $a(k) = \{a_1(k), a_2(k), \cdots a_q(k)\}$. Further, a tangent hyperbolic function is used as an activation function for hidden neurons and a linear activation function for output neuron. The input weight vector $W^{(I)}(k)$ shows the connection weight between the external applied inputs and the neurons of a hidden layer. The adjustable output weight vector, $W^O(k) = \{W^{1O}(k), W^{2O}(k), \cdots W^{qO}(k)\}$, represents the hidden to the output layer connections, and $yFFNN(k+1)$ denotes the FFNN output. The mathematical model of FFNN is given by

$$\begin{aligned} S_j(k) &= \sum_{i=1}^{p} W_{ij}^{(I)} x_i(k) + b_j \quad j = 1 \cdots q, \\ a_j(k) &= f\left(S_j(k)\right), \\ y_{\text{FFNN}}(k+1) &= \sum_{j=1}^{q} a_j(k) W_{1j}^{(o)}(k) + b_j. \end{aligned} \tag{1}$$

2.2. NARX Model. The NARX model represents a generic recurrent neural network having one step ahead output, $yNARX(k+1)$, depending upon the present and past values of the input which are called the exogenous inputs, namely, $\{u(k), u(k-1), \cdots u(k-du+1)\}$, as well as on its delayed values of the output, that is, $\{yNARX(k), yNARX(k-1), \cdots yNARX(k-dy+1)\}$ [10]. In the NARX neural network model, the internal architecture that performs this approximation is the multilayer perceptron (MLP). The dynamic

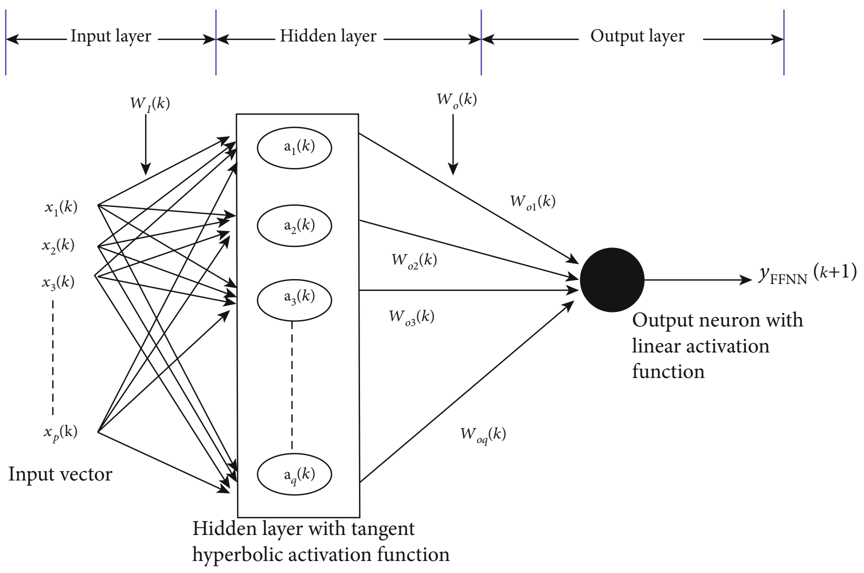

Figure 1: A feedforward neural network structure.

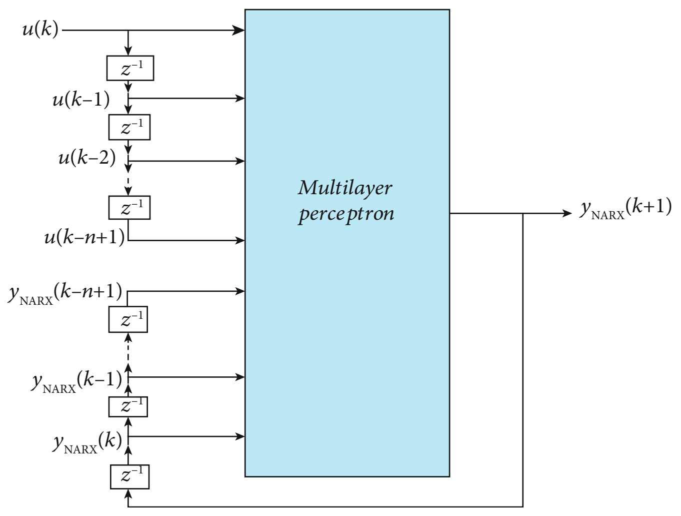

Figure 2: Nonlinear autoregressive with exogenous input (NARX) model.

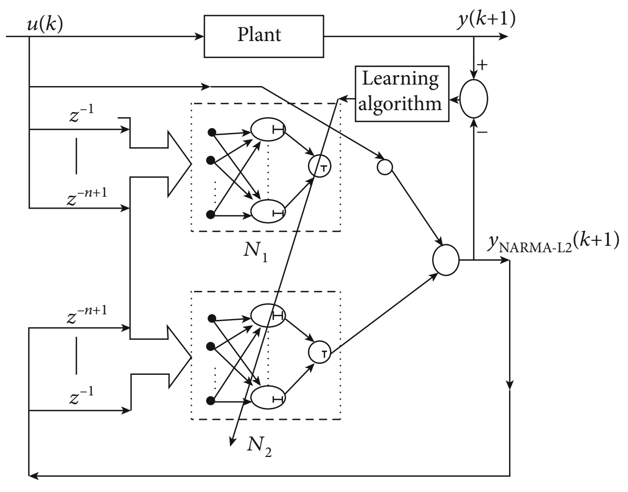

Figure 3: Nonlinear autoregressive and moving average.

behavior of the NARX model may be written in the following form:

$$y_{\text{NARX}}(k+1) = g\left[\begin{array}{l}\displaystyle\sum_{h=1}^{N} W_h^{(o)} f\left(\sum_{i=1}^{du} W_{ih}^{(I)} u(k-j+1) + \right.\\ \displaystyle\left.\sum_{j=1}^{dy} W_{jh}^{(I)} y_{\text{NARX}}(k-j+1) + b\right)\end{array}\right]. \qquad (2)$$

Note that in Figure 2, we have assumed that the two delay line memories du and dy are both equal to the order of the plant or different with $du \leq dy$, where f and g are a hyperbolic tangent and a linear activation function characterizing the hidden and output layers of the MLP, respectively, and N denotes the number of hidden layers.

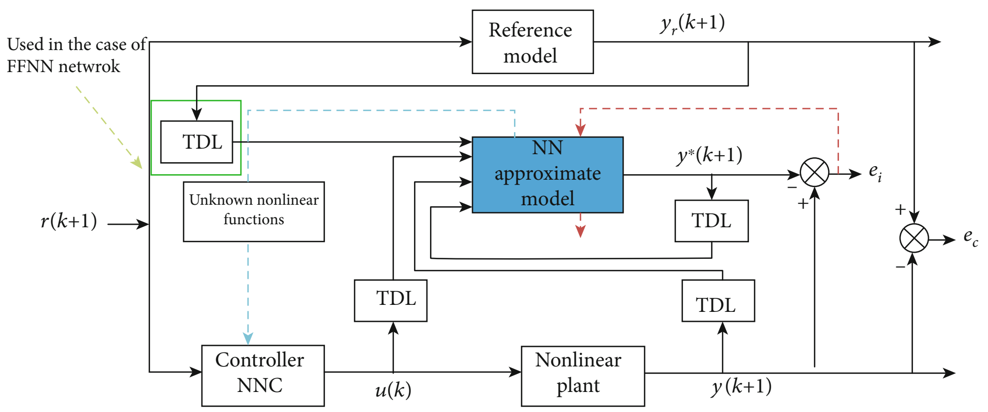

Figure 4: Adaptive control using neural network approximate models.

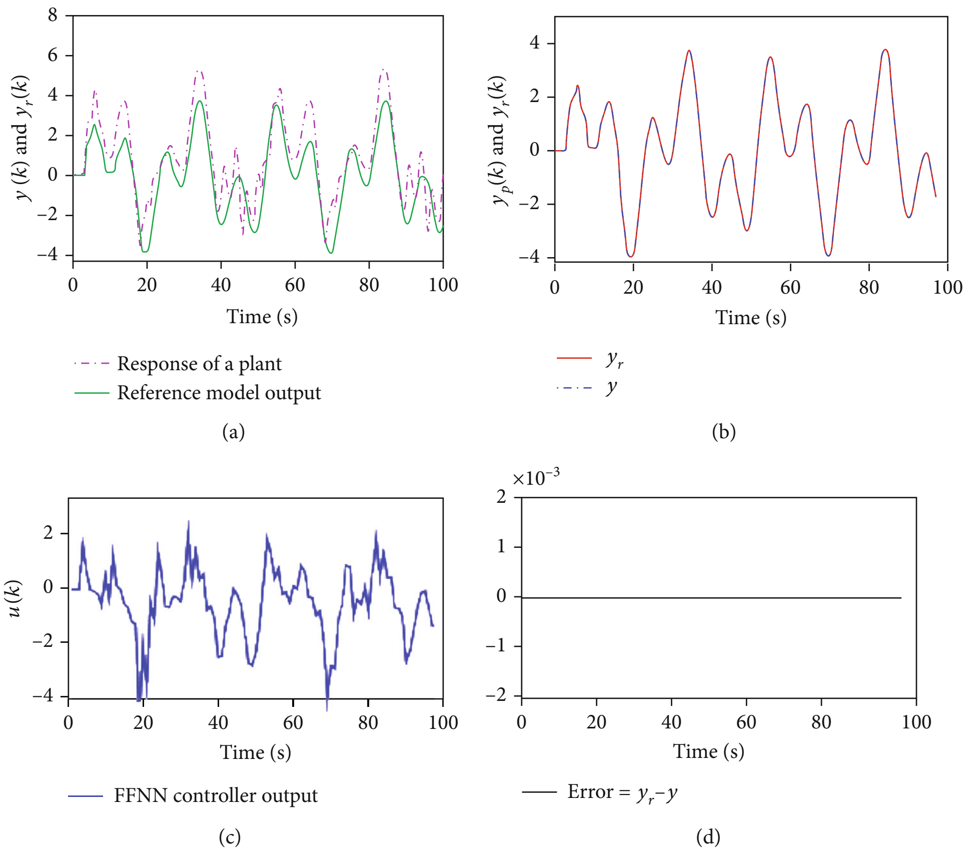

Figure 5: Example 1: (a) response for no control action (with $r = u = \sin(2\pi k/25) + \sin(2\pi k/10)$), (b) outputs of the reference model and the plant with control action, (c) FFNN controller output, and (d) control error.

2.3. NARMA-L2 Model.

2.3. NARMA-L2 Model. The nonlinear autoregressive moving average (NARMA) (Figure 3) is one of the most certified representations of general discrete-time nonlinear systems. This model representation is used in the form of the past, the current, and the future system parameters, as shown in [3]

$$y(k+d) = \Phi[y(k), y(k-1), \cdots y(k-n+1), u(k), u(k-1), \cdots u(k-n+1)], \quad (3)$$

where $y(k)$ and $u(k)$ are the input and output of the system, respectively, the relative degree d represents the delay of the system from control effort u to the output y, and $\Phi[\cdot]$ is a nonlinear function. This model is not convenient for finding a control signal $u(k)$. To overcome this problem, an efficient method is proposed by Narendra and Mukhopadhyay by introducing approximation models. Two classes of the NARMA model have been proposed, namely, NARMA-L1 and NARMA-L2. It was found that the second class involving two subapproximation functions is more efficient and

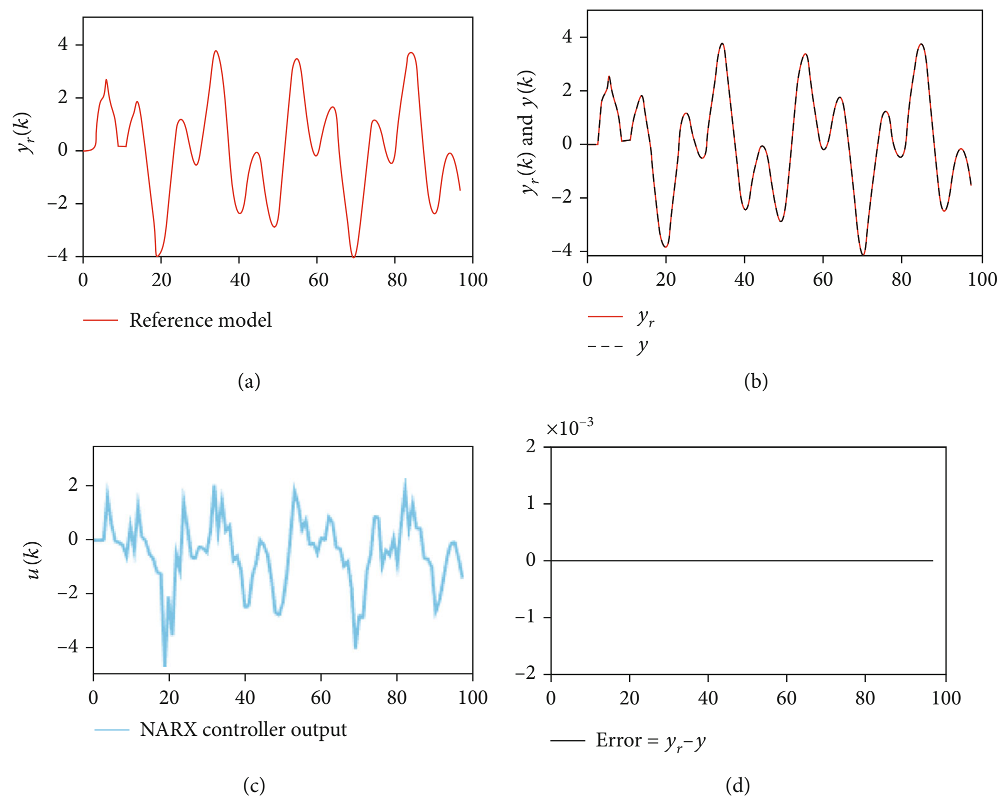

FIGURE 6: Example 1: (a) response of the reference model, (b) outputs of the reference model and the plant with control action, (c) NARX controller output, and (d) control error.

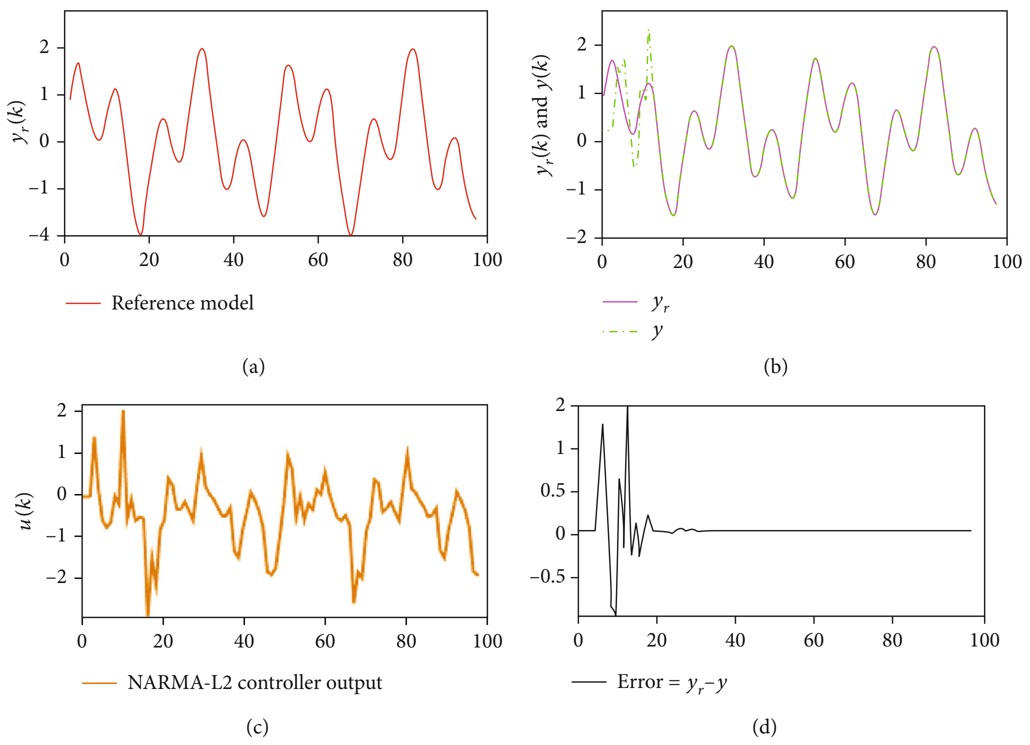

FIGURE 7: Example 1: (a) response of the reference model, (b) outputs of the reference model and the plant with control action, (c) NARMA-L2 controller output, and (d) control error.

adequate in the identification and adaptation of control contexts [20].

$$\begin{aligned} y_{\text{NARMA-L2}}(k+d) &= f_0[y(k), y(k-1), \cdots y(k-n+1), \\ &\quad u(k-1), \cdots u(k-n+1)] \\ &\quad + g_0[y(k), y(k-1), \cdots y(k-n+1), \\ &\quad u(k-1), \cdots u(k-n+1)]u(k), \end{aligned} \tag{4}$$

$$\begin{aligned} y_{\text{NARMA-L2}}(k+d) &= N_1[[y(k), y(k-1), \cdots y(k-n+1), \\ &\quad u(k-1), \cdots u(k-n+1)] \\ &\quad + N_2[[y(k), y(k-1), \cdots y(k-n+1), \\ &\quad u(k-1), \cdots u(k-n+1)]u(k). \end{aligned} \tag{5}$$

The NARMA-L2 is obtained by

$$\begin{aligned} y_{\text{NARMA-L2}}(k+d) &= g\left[\sum_{h=1}^{h1} W_h^{(1o)} f\left(\sum_{i=1}^{du} W_{ih}^{(1I)} u(k-j+1) \right.\right. \\ &\quad \left.\left. + \sum_{j=1}^{dy} W_{jh}^{(1I)} y_p(k-j+1) + b_1\right)\right] \\ &\quad + g\left[\sum_{h=1}^{h2} W_h^{(2o)} f\left(\sum_{i=1}^{du} W_{ih}^{(2I)} u(k-j+1) \right.\right. \\ &\quad \left.\left. + \sum_{j=1}^{dy} W_{jh}^{(2I)} y_p(k-j+1) + b_2\right)\right], \end{aligned} \tag{6}$$

where f and g are, respectively, the activation functions of the hidden layers and the linear layers of the networks N_1 and N_2.

Equations (5) to (7) represent single-input single-output (SISO) systems. As it can be seen from equation (4), NARMA-L2 consists of two nonlinear functions, f_0 and g_0, which can be approximated using two subnetworks (N_1 and N_2) as presented in equation (5). To design the neuro-controller, the number of delayed plant inputs and outputs is chosen based on a structural model. The size of the hidden layer is chosen such that the network can accurately approximate the nonlinearity of the system. There are two steps to construct the NARMA-L2 controller: identification and control.

The two subfunctions, N_1 and N_2, are used in the identification phase as well as to compute a signal as follows:

Table 1: Comparison of the performance index of the proposed controller and other controllers (example 1).

Controller	MSE	Nbr. of iterations	Learning rate
RBFN [9]	0.012	2000	0.005
DRNN [2]	0.0253	2000	0.0154
FFNN	1.6366×10^{-31}	100	0.1
NARX	5.1093×10^{-19}	100	0.1
NARMA-L2	0.25	100	0.1

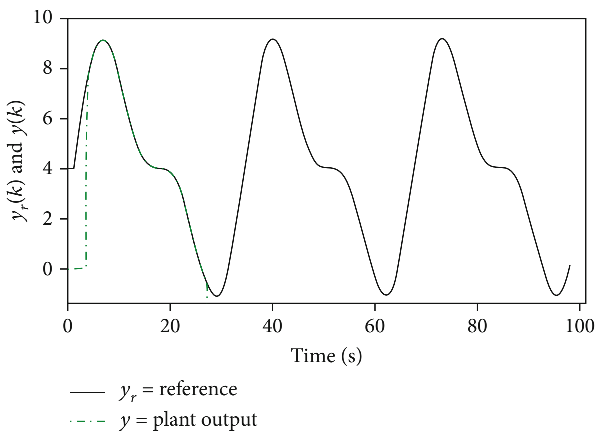

Figure 8: Plant output and reference model response for $n = m = 1$.

$$u(k) = \frac{[y_r(k+d) - N1(y(k), \cdots y(k-n+1), u(k-1), \cdots, u(k-n+1))]}{N2(y(k), \cdots, y(k-n+1), u(k-1), \cdots, u(k-n+1))}, \tag{7}$$

where $y_r\,(k+d)$ is the reference signal to follow. This controller can be implemented using the model of the NARMA-L2 system previously identified. If the system is precisely approximated, the output of the system will be equal to the output of the reference model.

3. Controller Design

The control configuration based on the NN approximate model is shown in Figure 4. In this control scheme, a NN model is utilized to approximate the nonlinear function of the unmodeled dynamics. The system model is considered unknown. The use of neural networks is justified by this absence of a model. In this case, three approaches predominate. In the two approaches based on the FFNN and NARX models, a single network is used for system control. The third approach based on the NARMA-L2 model uses two networks to deduce the controller adaptation law.

3.1. Learning Algorithm. The process of training consists in modifying the weights in an organized way using an appropriate algorithm. During the training process, a specified

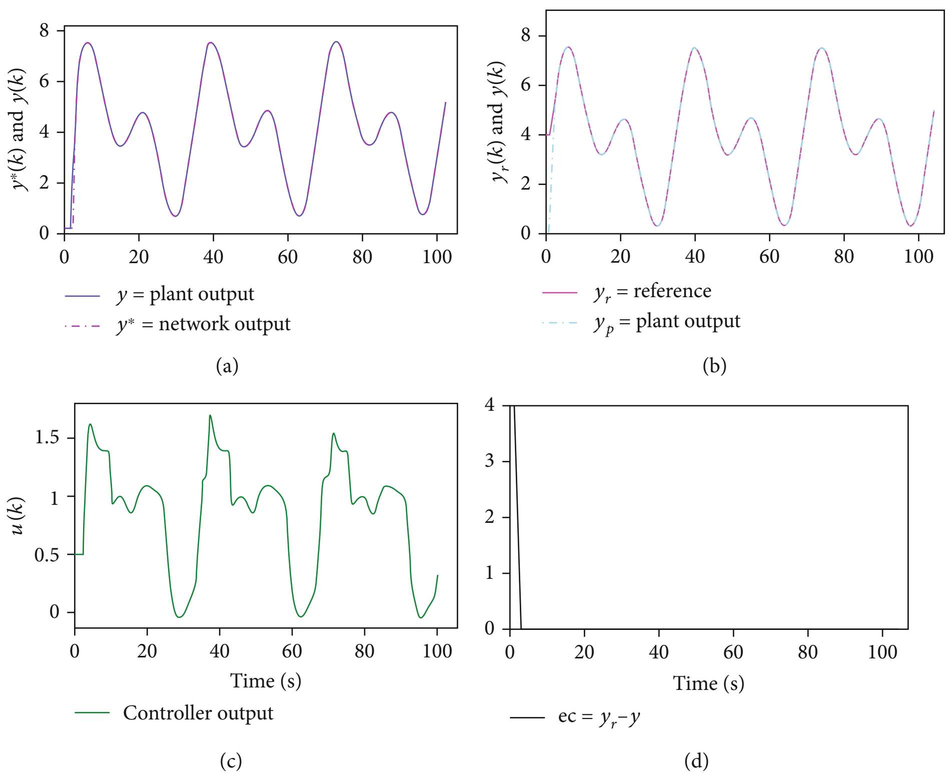

FIGURE 9: Example 2: (a) response of an ANN-NARX identification model, (b) response of the plant under the NARX controller, (c) NARX controller output, and (d) control error with the NARX model (using $n = 3$ and $m = 2$).

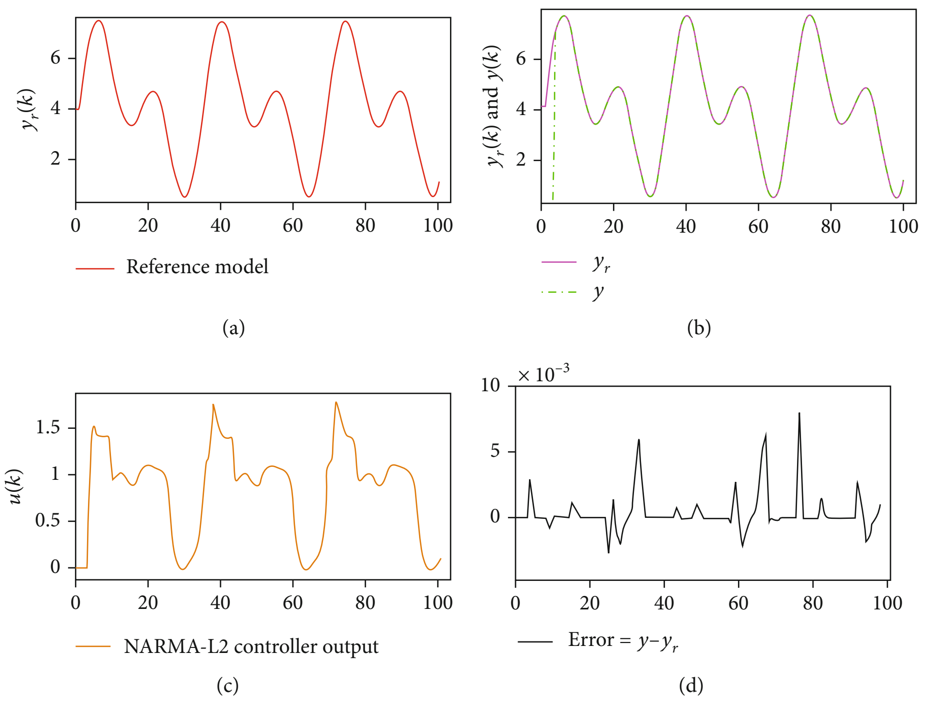

FIGURE 10: Example 2: (a) response of the reference model, (b) response of the plant under the NARMA-L2 controller, (c) NARMA-L2 controller output, and (d) control error with the NARMA-L2 model (using $n = 3$ and $m = 2$).

number of inputs and their desired output are introduced in the network. Then, the weights are tuned so that the neural network produces an output close to the target values. The fundamental training algorithm for multilayer networks is the backpropagation (BP) algorithm [20]. This algorithm is of iterative type, and it is based on the minimization of a sum-squared error (MSE) utilizing the optimization gradient descent method. The MSE is used as the cost function which is a function of error, which is defined as follows:

$$E(k) = \frac{1}{2}\sum_{k=1}^{N} (y_r(k) - y(k))^2, \tag{8}$$

where $y_r(k)$ and $y(k)$ are the desired output of the network and the actual response of the network on the given input pattern $u(k)$, respectively, and N is the dimension of the training set. The modification of the weights of the jth neuron in the μth layer is performed according to the formula:

$$\begin{cases} W_{ij}^{\mu}(k+1) = W_{ij}^{\mu}(k) - \eta \dfrac{\partial E}{\partial W_{ij}^{\mu}}, \\ b_i^{\mu}(k+1) = b_i^{\mu}(k) - \eta \dfrac{\partial E}{\partial b_i^{\mu}}, \end{cases} \tag{9}$$

where

$$\begin{cases} \dfrac{\partial E}{\partial W_{ij}^{\mu}} = \dfrac{\partial E}{\partial S_{ij}^{\mu}} \times \dfrac{\partial S_{ij}^{\mu}}{\partial W_{ij}^{\mu}}, \\ \dfrac{\partial E}{\partial b_i^{\mu}} = \dfrac{\partial E}{\partial S_{ij}^{\mu}} \times \dfrac{\partial S_{ij}^{\mu}}{\partial b_i^{\mu}}, \\ S_{ij}^{\mu} = \displaystyle\sum_{j=1}^{\mu-1} W_{ij}^{\mu} a_j^{\mu-1} + b_i^{\mu}. \end{cases} \tag{10}$$

Therefore, $a^{\mu} = f^{\mu}(S_{ij}^{\mu})$, $\partial S_i^{\mu}/\partial W_{ij}^{\mu} = a_j^{\mu-1}$, and $\partial S_i^{\mu}/\partial b_i^{\mu} = 1$. If we define $\partial E/\partial S_{ij}^{\mu} = \delta_i^{\mu}$, then, $\partial E/\partial W_{ij}^{\mu} = \delta_i^{\mu} a_j^{\mu-1}$.

Thus, each element in $W^{\mu}(k)$ and $b^{\mu}(k)$ can be updated as

$$\begin{cases} W_{ij}^{\mu}(k+1) = W_{ij}^{\mu}(k) - \eta\delta_i^{\mu} a_j^{\mu-1}, \\ b_i^{\mu}(k+1) = b_i^{\mu}(k) - \eta\delta_i^{\mu}. \end{cases} \tag{11}$$

The error δ is defined as follows:

$$\begin{gathered} \delta_j^{\mu}(k) = \begin{cases} -f'\left(S_j^{\mu}(k)\right)e(k), & \text{for } \mu = M \text{ (OL)}, \\ f'\left(S_j^{\mu}(k)\right)\displaystyle\sum_{p=1}^{\mu+1}\left(\delta_p^{\mu+1}(k)W_{pj}^{\mu+1}\right), & \text{for } \mu = 1, \cdots, M-1 \text{ (HL)}, \end{cases} \\ \delta_j^{\mu}(k) = \begin{cases} -e(k), & \text{for } \mu = M \text{ (OL)}, \\ \left(1 - a_j^2(k)\right)\left(\delta^{\mu+1}(k)W_{ij}^{\mu+1}\right), & \text{for } \mu = 1, \cdots, M-1 \text{ (HL)}, \end{cases} \end{gathered} \tag{12}$$

where $f^{(\mu)}(x) = \tan h(x) = (e^x - e^{-x})/(e^x + e^{-x})$ and $f^{\mu+1}(x) = x$.

4. Results and Discussion

Simulation studies have been carried out using the Matlab software environment to verify the performance of the proposed methods. Three different nonlinear dynamical systems are presented to evaluate the performance of the NN controllers. Two hidden layers are used with $H1NN = 20$ and $H2NN = 10$ (thus, 20 neurons are present in the first and 10 neurons in the second hidden layers).

4.1. Example 1. Consider the nonlinear discrete time which is described by the third-order difference equation [19]:

$$y(k+1) = \frac{5y_p(k)y_p(k-1)}{1 + y_p^2(k) + y_p^2(k-1) + y_p^2(k-2)} + u(k) + 0.8u(k-1). \tag{13}$$

The system can be reformulated as

$$y(k+1) = h(y(k), y(k-1), y(k-2)) + u(k) + 0.8u(k-1). \tag{14}$$

A reference model is described by the third-order difference equation:

$$y_r(k+1) = 0.32y_r(k) + 0.64y_r(k-1) - 0.5y_r(k-2)] + r(k), \tag{15}$$

where $r(k)$ is a bounded reference input. The function h is considered to be unknown; it can be estimated using neural network N_h. The control input to the plant at any instant k is computed using $Nh(-)$ in place of h.

If $y(k+1) = yr(k+1)$, then,

$$\begin{aligned} u(k) = &-N_h[y(k), y(k-1), y(k-2)] - 0.8u(k-1) \\ &+ 0.32y_r(k) + 0.64y_r(k-1) - 0.5y_r(k-2) + r(k), \end{aligned} \tag{16}$$

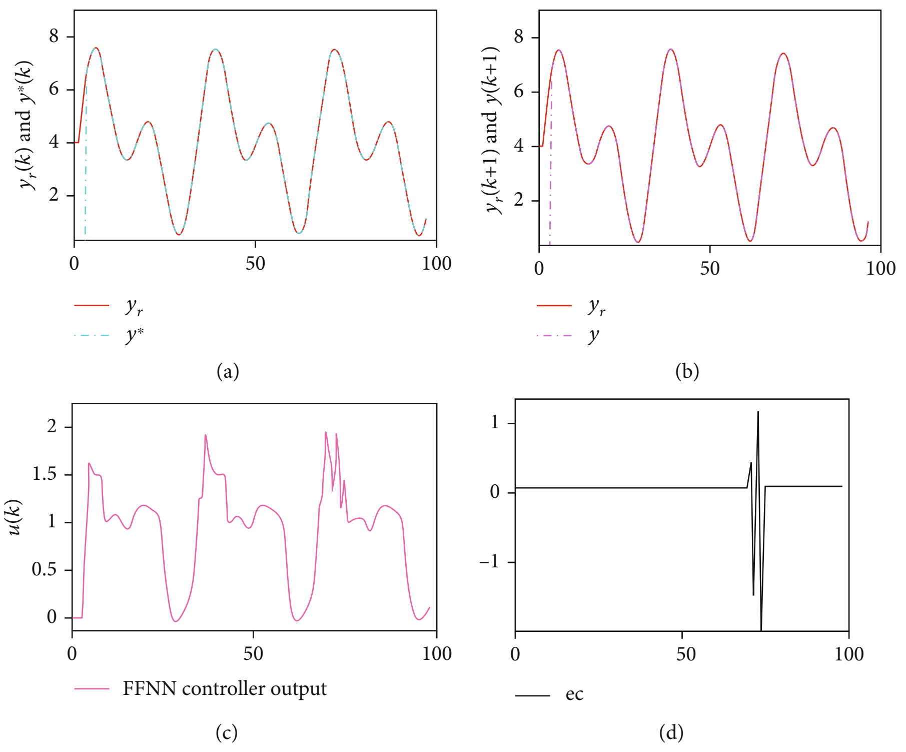

Figure 11: Example 2: (a) response of an ANN-FFNN identification model, (b) response of the plant under the FFNN controller, (c) FFNN controller output, and (d) control error with the FFNN model (using $n = 3$ and $m = 2$).

Table 2: Comparison of the performance index of the proposed controller (example 2).

Controller	MSE	Nbr. of iterations	Learning rate
FFRN	0.0698	100	0.01
NARX	1.0027×10^{-13}	300	0.01
NARMA-L2	2.99×10^{-6}	100	0.01

whereas for NARMA-L2 the control signal is generated based on equation (7):

$$u(k) = \frac{[y_r(k+d) - f_0(y(k), y(k-1), y(k-2), u(k-1))]}{g_0(y(k), y(k-1), y(k-2), u(k-1))}. \tag{17}$$

N_1 and N_2 are used to learn $f_0(\cdot)$ and $g_0(-)$; then,

$$u(k) = \frac{[y_r(k+d) - N_1(y(k), y(k-1), y(k-2), u(k-1))]}{N_2(y(k), y(k-1), y(k-2), u(k-1))}. \tag{18}$$

The simulation reported in Figures 5–7 indicates stable and efficient online control. The improvements in the responses, when the neural networks in the approximate model are used to generate the control input to the plant, are evident from the figure. The outputs of the controlled plant and the reference model are shown and indicate that the output error is almost zero. All three neural networks FFNN-, NARX-, and NARMA-L2-based controllers provided satisfactory results.

The performance index of the different controllers with various neural network models is given in Table 1. We can notice that all the designed controllers give better performance. Nevertheless, the performance of the FFNN and NARX indicated better attainment regarding the performance indices (MSE).

4.2. Example 2. Consider the nonlinear discrete-time system [1, 17]:

$$y(k+1) = \sin[y(k)] + u(k)(5 + \cos[y(k)u(k)]), \tag{19}$$

where $y(k)$ stands for the system output and $u(k)$ is the control input at time index k. The output $y(k+1)$ is required to track the desired trajectory $y_r(k+1)$ given in

$$y_r(k) = \left(\sin\left(\frac{2\pi k}{10}\right) + \sin\left(\frac{2\pi k}{25}\right)\right) \times r(k). \tag{20}$$

The order of the plant is $n = 1$, so the inputs for the FFNN controller are $y_r(k)$ and $y(k)$) and the inputs for the NARMA-L2 controller are $y(k)$ and $u(k)$), and NARX models are $y^*(k)$, $y(k)$,and $u(k)$).

The control results for different epochs and learning rates along with control error are shown in Figure 8. As we can see from the results, the network is unstable using $n = m = 1$ in the delayed input/output (not enough information).

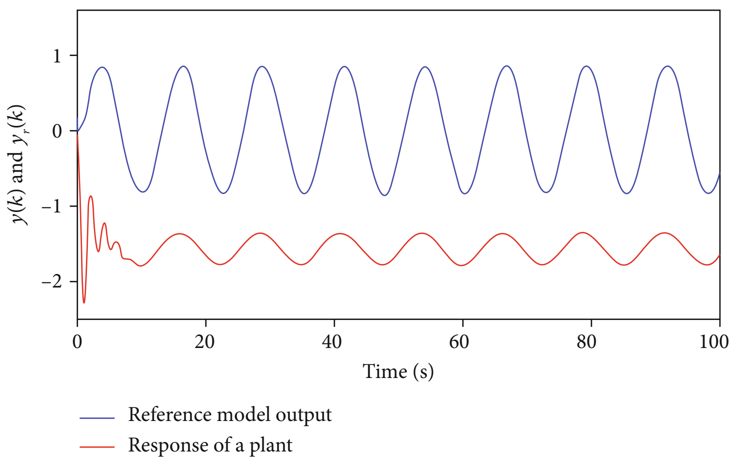

FIGURE 12: Reference model and open-loop response of the plant without a controller (example 3).

TABLE 3: Comparison of the performance index of the proposed controller and other controllers (example 3).

Controller	MSE	Nbr. of iterations	Learning rate
RBFN [9]	0.1427	2000	0.027
DRNN [2]	0.0288	800	0.0354
FFNN	9.644×10^{-32}	1000	0.05
NARX	2.1077×10^{-31}	500	0.1
NARMA-L2	1.802×10^{-31}	100	0.01

The model of discrete-time plants introduced here can be also described by the following nonlinear difference equations:

$$y(k+1) = h[y(k), y(k-1), \cdots, y(k-n+1)] + \sum_{i=0}^{m-1} \beta u(k-i). \tag{21}$$

The same proposed scheme in Figure 4 is considered. To approximate the unknown h, an FNN-based identification model is used:

$$y^*(k+1) = N_h[y(k), y(k-1)] + u(k). \tag{22}$$

The control action is required to get $y_r(k) = y(k)$.

In this example, performance comparisons of the BP-NNs are carried out with the three approximate models, FFNN, NARX, and NARMA-L2. The results are shown in Figures 9–11. Also, notice that after sufficient online training of the plant, it has also converged to the desired response. The result during the training is presented in the figures. It can be seen from these figures that the control error (ec) reduces quickly to zero in the case of the NARX and NARMA-L2 controllers as compared to the control error obtained with FFNN. This suggests the strong learning ability of recurrent NN models over FFNN in this example.

The numerical parameters used in the second system are presented in Table 2, as well as the mean square error (MSE) obtained with different controllers.

4.3. Example 3. The second-order differential equation which describes the dynamics of a single-link robotic manipulator is given as

$$ml^2 \frac{d^2y(t)}{dt^2} + D\frac{dy(t)}{dt} + mgl \cos(y(t)) = u(t). \tag{23}$$

The angular position of the robotic manipulator arm is represented by $y(t)$. The other parameters include link length which is denoted by l. Also, the term $D(dy(t)/dt)$ quantifies the viscous friction torque, and $g = 9.8\, m/s^2$ is the acceleration due to gravity. For simplicity, the values of $m = l = D = 1$ are used in this paper. Further, $u(t)$ is the control torque exerted on the robotic arm. The corresponding state space representation of the above differential equation is given by

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \dfrac{-g\cos(y)}{l} & \dfrac{-D}{ml^2} \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{ml^2} \end{bmatrix} [u]. \tag{24}$$

The difference equation of the one-link robotic manipulator with a sampling period is given by

$$y(k+1) = h[y(k), y(k-1)] + T^2 u(k-1), \tag{25}$$

where the nonlinear function h is

$$\begin{aligned} h[y(k), y(k-1)] = {} & (2-T)y(k) + y(k-1)(T-1) \\ & - 9.8T^2 \cos((y(k-1)). \end{aligned} \tag{26}$$

The desired reference model dynamics is given by the following difference equation:

$$\begin{aligned} y_r(k+1) = {} & y_r(k)(2-1.5T) + y_r(k-1) \\ & \cdot \left(-1 - 2.5T^2 + 1.5T\right) + T^2 r(k-1), \end{aligned} \tag{27}$$

where $r(k) = \sin(kT)$ is the externally applied input.

Before control action, the sampling period T should be chosen with care. It is taken to be $T = 0.4$. In this part, the

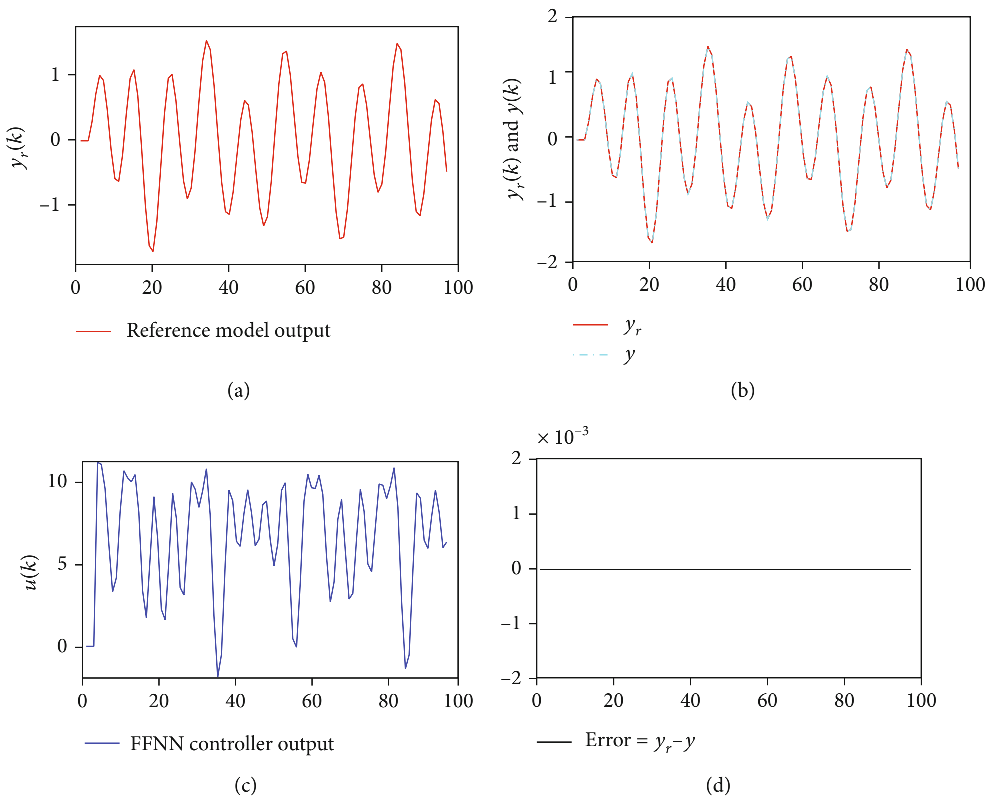

FIGURE 13: Example 3: (a) response of the reference model, (b) response of the plant under the adaptive controller, (c) FFNN controller output, and (d) control error (using $n = 2$ and $m = 2$) (online learning).

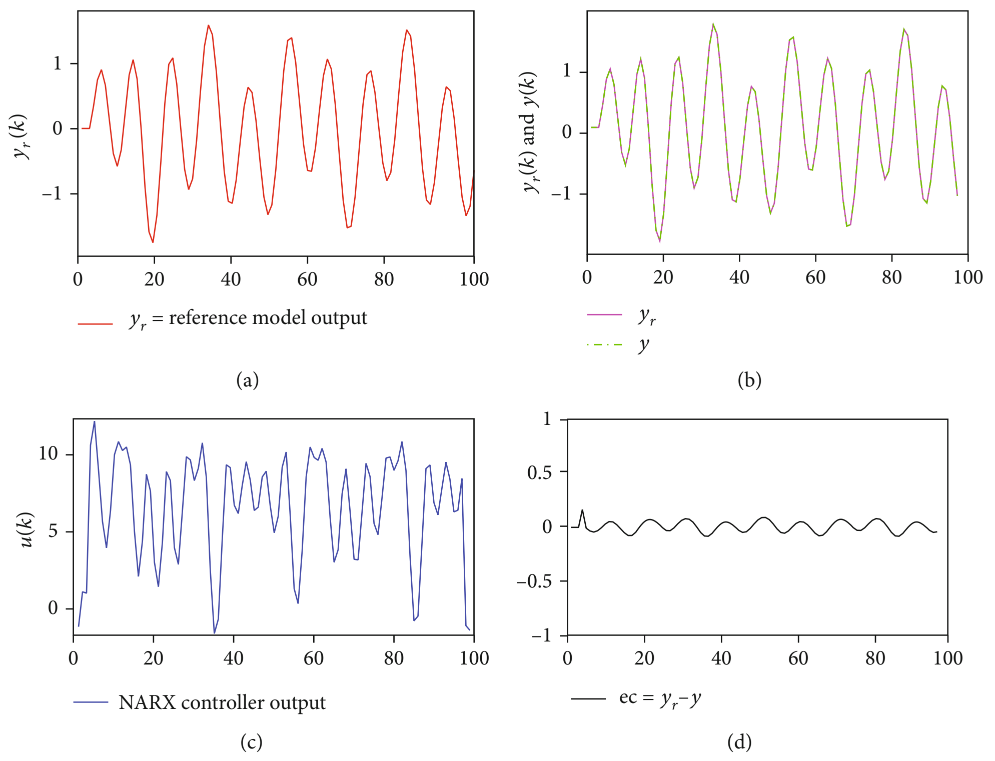

FIGURE 14: Example 3: (a) response of the reference model, (b) response of the plant under the adaptive controller, (c) NARX controller output, and (d) control error (using $n = 2$ and $m = 2$).

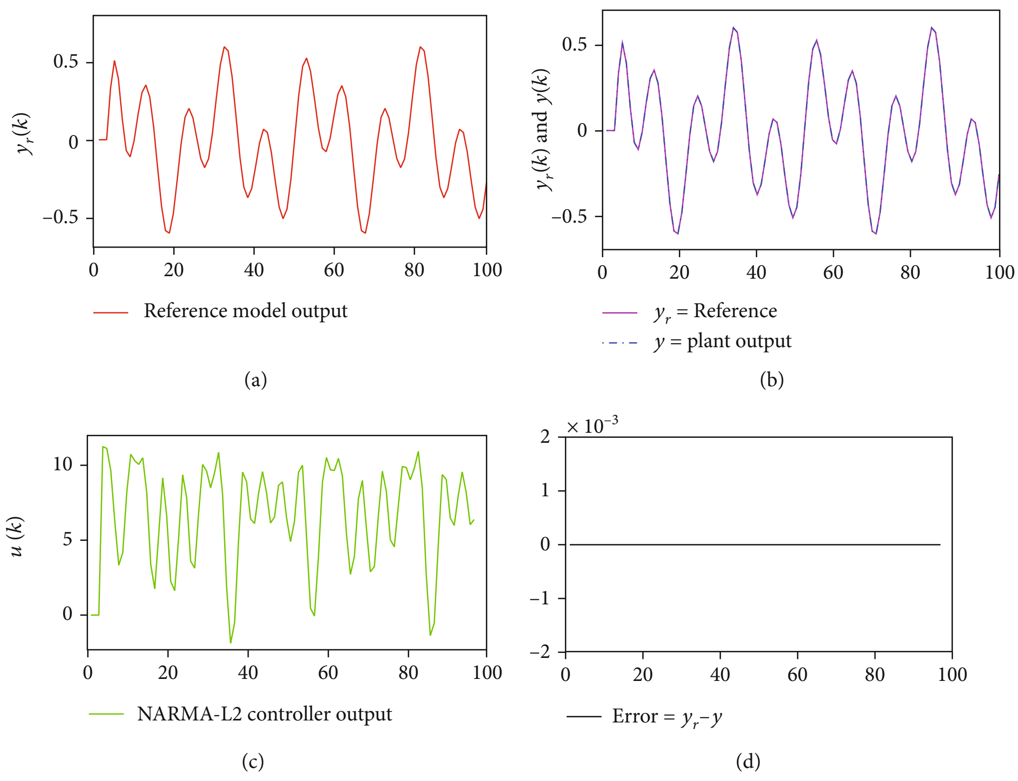

FIGURE 15: Example 3: (a) response of the reference model, (b) response of the plant under the adaptive NARMA-L2 controller, (c) controller output, (d) control error (using $n = 2$ and $m = 2$).

control is to make the robotic link output follow the reference trajectory signal. The response of the plant without control is shown in Figure 12. Clearly, the plant's output is not following the reference input. So, the control scheme is set up and the training is continued for 100 epochs, and each epoch contains 100 training samples.

In the simulation studies, N_1 and N_2 were multilayer neural networks chosen with three-layer networks with p inputs, 20 neurons in the first hidden layer, 10 neurons in the second hidden layer, and one output neuron.

From equation (25), the control input $u(k)$ can be computed from acknowledging $y(k+1)$ and its past values as

$$\begin{aligned}
u(k) &= [-f[y(k), y(k-1)] + y(k+1)] \times \frac{1}{T^2}, \\
u(k) &= [-f[y(k), y(k-1)] + y(k)(2 - 1.5T) + y(k-1) \\
&\quad \cdot \left(-1 - 2.5T^2 + 1.5T\right) + T^2 r(k-1)\big] \times \frac{1}{T^2}, \\
r(k) &= \sin\left(\frac{2\pi k}{25}\right) + \sin\left(\frac{2\pi k}{10}\right).
\end{aligned} \tag{28}$$

The mean square error (MSE) obtained with different controllers for example 3 is presented in Table 3. From the table, it can be seen that the MSE obtained with FFNN, NARX, and NARMA-L2 is minimum as compared to the MSE obtained with the RBFN- and DRNN-based controllers.

In this study, all the three methods provide simultaneous structure and parameter learning within the approximate models. The learning rate, η, is taken to be 0.01. From Figures 13–15, it can be seen that the response of the robotic link is successfully controlled.

5. Conclusions

This paper describes the application of NN models as a controller. Both feedforward and two recurrent NARX and NARMA-L2 predictors are proposed, tested, and compared. The method for the adjustment of parameters in generalized neural networks is treated, and the concept of dynamic back propagation is introduced. The training of all the three neural networks is done online. Simulation and comparative studies demonstrate the superior performance of the proposed approaches. It can be concluded that the ANNs can be successfully utilized for controlling different nonlinear dynamic systems. In addition, future works will explore the adaptive neural network control for nonlinear MIMO systems under disturbances using the hybrid learning method.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers and Structures*, vol. 169, pp. 1–12, 2016.

[2] R. Kumar, S. Srivastava, and J. R. P. Gupta, "Diagonal recurrent neural network based adaptive control of nonlinear dynamical systems using Lyapunov stability criterion," *ISA Transactions*, vol. 67, pp. 407–427, 2017.

[3] Y. Al-dunainawi, M. F. Abbod, and A. Jizany, "A new MIMO ANFIS-PSO based NARMA-L2 controller for nonlinear dynamic systems," *Engineering Applications of Artificial Intelligence*, vol. 62, pp. 265–275, 2017.

[4] C. Treesatayapun, "Nonlinear discrete-time controller with unknown systems identification based on fuzzy rules emulated network," *Applied Soft Computing*, vol. 10, no. 2, pp. 390–397, 2010.

[5] K. El Hamidi, M. Mjahed, A. El Kari, and H. Ayad, "Neural and fuzzy based nonlinear flight control for an unmanned quadcopter," *International Review of Automatic Control*, vol. 11, no. 3, pp. 98–106, 2018.

[6] X. L. Li, C. Jia, D. X. Liu, and D. W. Ding, "Adaptive control of nonlinear discrete-time systems by using OS-ELM neural networks," *Abstract and Applied Analysis*, vol. 2014, Article ID 267609, 11 pages, 2014.

[7] K. El Hamidi, M. Mjahed, A. El Kari, and H. Ayad, "Quadcopter attitude and altitude tracking by using improved PD controllers," *International Journal of Nonlinear Dynamics and Control*, vol. 1, no. 3, pp. 287–303, 2019.

[8] Z. Cömert and A. F. Kocamaz, "A study of artificial neural network training algorithms for classification of cardiotocography signals," *Bitlis Eren University Journal of Science and Technology*, vol. 7, no. 2, pp. 93–103, 2017.

[9] R. Kumar, S. Srivastava, and J. R. P. Gupta, "Comparative study of neural networks for control of nonlinear dynamical systems with lyapunov stability-based adaptive learning rates," *Arabian Journal for Science and Engineering*, vol. 43, no. 6, pp. 2971–2993, 2018.

[10] S. Haykin and N. Network, "A comprehensive foundation," *Neural Networks*, vol. 2, no. 2004, p. 41, 2004.

[11] K. Patan, *Artificial Neural Networks for the Modelling and Fault Diagnosis of Technical Processes*, Springer, 2008.

[12] H. Wu, Y. Zhou, Q. Luo, and M. A. Basset, "Training feedforward neural networks using symbiotic organisms search algorithm," *Computational intelligence and neuroscience*, vol. 2016, Article ID 9063065, 14 pages, 2016.

[13] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.

[14] E. Diaconescu, "The use of NARX neural networks to predict chaotic time series," *Wseas Transactions on computer research*, vol. 3, no. 3, pp. 182–191, 2008.

[15] K. Nouri, R. Dhaouadi, and N. B. Braiek, "Adaptive control of a nonlinear dc motor drive using recurrent neural networks," *Applied Soft Computing*, vol. 8, no. 1, pp. 371–382, 2008.

[16] R. Celikel, "NARMA-L2 controller for single link manipulator," in *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, pp. 1–6, Malatya, Turkey, 2018.

[17] M. S. Aftab and M. Shafiq, "Neural networks for tracking of unknown SISO discrete-time nonlinear dynamic systems," *ISA Transactions*, vol. 59, pp. 363–374, 2015.

[18] Y. Gao and M. J. Er, "NARMAX time series model prediction: feedforward and recurrent fuzzy neural network approaches," *Fuzzy sets and systems*, vol. 150, no. 2, pp. 331–350, 2005.

[19] R. G. S. Asthana, "Evolutionary algorithms and neural networks," in *Soft Computing and Intelligent Systems*, pp. 111–136, Elsevier Inc., 2000.

[20] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jess, *Neural Network Design*, vol. 20, Pws Pub, Boston, 1996.