Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 719620, 12 pages
http://dx.doi.org/10.1155/2015/719620

Research Article

Fuzzy Counter Propagation Neural Network Control for a Class of Nonlinear Dynamical Systems

Vandana Sakhre,[1] Sanjeev Jain,[1] Vilas S. Sapkal,[2] and Dev P. Agarwal[3]

[1]*Madhav Institute of Technology & Science, Gwalior 474005, India*
[2]*SGB Amravati University, Amravati 444062, India*
[3]*IIT, Delhi 110001, India*

Correspondence should be addressed to Vandana Sakhre; vssakhre@gmail.com

Received 3 May 2015; Revised 1 August 2015; Accepted 3 August 2015

Academic Editor: Ezequiel López-Rubio

Fuzzy Counter Propagation Neural Network (FCPN) controller design is developed, for a class of nonlinear dynamical systems. In this process, the weight connecting between the instar and outstar, that is, input-hidden and hidden-output layer, respectively, is adjusted by using Fuzzy Competitive Learning (FCL). FCL paradigm adopts the principle of learning, which is used to calculate Best Matched Node (BMN) which is proposed. This strategy offers a robust control of nonlinear dynamical systems. FCPN is compared with the existing network like Dynamic Network (DN) and Back Propagation Network (BPN) on the basis of Mean Absolute Error (MAE), Mean Square Error (MSE), Best Fit Rate (BFR), and so forth. It envisages that the proposed FCPN gives better results than DN and BPN. The effectiveness of the proposed FCPN algorithms is demonstrated through simulations of four nonlinear dynamical systems and multiple input and single output (MISO) and a single input and single output (SISO) gas furnace Box-Jenkins time series data.

1. Introduction

With the rapid advance of computers, the digitization has swept in all fields of science and technology with special emphasize on modeling and identification. Most of the physical processes are continuous in nature but they are generally modeled using discrete time with the use of derivative and integral as they are easily realizable. Taking these advantages, an enormous research has been focused on discrete time methods and many techniques based on linear and nonlinear discrete systems have been developed [1, 2]. Neural network has been widely used for nonlinear dynamical systems [3–5]. Various control systems like back stepping control [6, 7], sliding mode control [8, 9], and control using soft computing [10] are continuous source of interest. The nonlinear dynamical system is a generic problem which finds its application in every field of engineering. The technique based on soft computing, fuzzy logic, and neural network has also found its application in modeling of systems in various application domains. Feedforward Neural Network (FNN) is one of the most commonly used networks for this purpose; FNN with Back Propagation algorithm is another powerful network. Neural network is most popular approach due to its capability of modeling most of the nonlinear functions approximately to any arbitrary degree of accuracy [11]. Most of the systems are designed using feedforward network with gradient descent learning but the gradient descent method encounters the problem of slow convergence. This problem of convergence can be overcome by use of some Cauchy-Newton [12] Quasi-Newtonian method and Levenberg-Marquardt algorithms [13].

Optimal controllers for hybrid dynamical systems (HDS) developed using Hamilton-Jacobi-Bellman (HJB) solution method [14, 15]. Novel Lyapunov-Krasovskii functional (LKF) with triple integral time constructed for exponential synchronization of complex dynamical networks will control packet loss and additive time varying delay [16]. Control of discrete time varying system using dynamical model is difficult. To overcome this condition, new neural network approximation structure was developed to solve optimal tracking problem of nonlinear discrete time varying time system using reinforcement learning (RL) method [17]. To

enhance the performance of nonlinear multivariable system, fuzzy optimal controller based Takagi-Sugeno model was developed which was used to minimize a quadratic performance index [18]. An ultimate stable training algorithm inspired by adaptive observer for black box identification of nonlinear discrete systems developed using state space recurrent neural network. The algorithm developed was using only input and output measurement in discrete time [19]. Indirect adaptive controller that uses neural network was presented for the identification and control of experimental pilot distillation column. This system is multivariable with unknown dynamics and the neural network was trained using Levenberg-Marquardt algorithm [20]. Adaptive finite time stabilization of a class of switched nonlinear systems was investigated with unknown nonlinear terms using neural network. The finite law and adaptive law were constructed using radial basis function neural network (RBFNN) to approximate the unknown packaged function. Stability analysis of RBFNN was observed to evaluate the states of the closed loop systems [21]. An adaptive fuzzy output tracking control approach was proposed in [22] for single input single output (SISO) system which is uncertain and nonlinear under arbitrary switching. An iterative learning control scheme was proposed for a class of nonlinear dynamic systems which includes holonomic systems as its subsets with linear feedback mechanism and feedforward learning strategies [23].

In this proposed work we have used instar-outstar structure based CPN with Fuzzy Competitive Learning (FCL). We have designed Fuzzy Counter Propagation Network design to control some nonlinear dynamical systems. In the FCPN, CPN model is trained by FCL algorithms. The FCL learning is used for adjusting weights and update of Best Matched Node (BMN) in discrete time nonlinear dynamical system.

The main contributions of this research are as follows:

(1) This paper contributes the approximation for a class of nonlinear dynamical systems using Fuzzy Competitive Learning Based Counter Propagation Network (FCPN).

(2) FCPN is employed to optimize the Mean Absolute Error, Mean Square Error, Best Fit Rate, and so forth.

(3) Performance criteria of proposed FCPN for nonlinear dynamical systems are effectively improved by compensating reference signal and controller signal.

The paper is organized as follows. In Section 2, problem formulations for a class of nonlinear dynamical systems are given. Section 3 contains description of Feedforward Neural Network (FNN) with Back Propagation Network, Dynamic Network, and Fuzzy Learning Based Counter Propagation Network. Dynamical learning for CPN and optimal learning of dynamical system are presented in Section 4. The simulation results and comparison of different nonlinear dynamical models are presented in Section 5. Section 6 gives conclusion of the study.

2. Problem Formulation

Let us consider four models of discrete time nonlinear dynamical systems [24] for single input single output (SISO) and multiple input and single output (MISO) system considered in this paper and they are described by difference equations (1)–(4) and Box-Jenkins time series data [10]. Let $f : R^n \rightarrow R$ and $g : R^m \rightarrow R$ be the nonlinear continuous differentiable function of Model I–Model IV approximated by FCPN to the desired degree of accuracy. Once the system has been parameterized, the performance evaluation has been carried out by FCPN for (1) to (4).

Model I:

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + g[u(k), \ldots, u(k-m+1)]. \tag{1}$$

Model II:

$$y(k+1) = f[y(k), \ldots, y(k-n+1)] + \sum_{i=0}^{m-1} \beta_i u(k-i). \tag{2}$$

Model III:

$$y(k+1) = f[y(k), \ldots, y(k-n+1)] + g[u(k), \ldots, u(k-m+1)]. \tag{3}$$

Model IV:

$$y(k+1) = f[y(k), \ldots, y(k-n+1);\ u(k), \ldots, u(k-m+1)], \tag{4}$$

where $[u(k), y(k)]$ represent the input-output pair of the system at time k and their order is represented by (n, m). FCL is used to learn the system defined for (1) to (4) and the performance of the FCPN can be measured by error function given in

$$E(k) = \frac{1}{2}[y(k) - \overline{y}(k)]^2, \tag{5}$$

where $u(k)$ is the input, $y(k)$ is the system output, $\overline{y}(k)$ is neural network output, and (5) can be written for neural controller as

$$E^c = \frac{1}{P}\sum_{p=1}^{P}[y(k) - \overline{y}^c(k)]^2, \tag{6}$$

where $\overline{y}^c(k)$ is neural controller output and (6) is known as minimized error function and P is total number of input patterns.

3. Back Propagation Network (BPN)

A neural network is one of the most popular intelligent systems which has capability to approximate calculated and target value at an arbitrary degree of accuracy. BPN is one

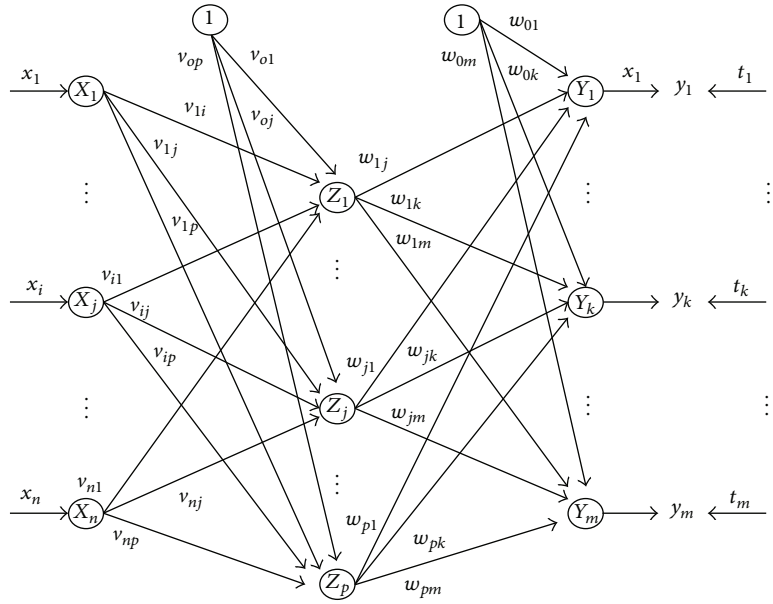


Figure 1: General architecture of BPN.

of the most commonly used networks for this purpose. The BPN consists of input layer, hidden layer, and output layer and possesses weighted interconnections. The BPN is based on supervised learning. This method is used where error is propagated back to the hidden layer. The aim of this neural network is to train the net to achieve a balance between the net's ability to respond (memorization) and its ability to give reasonable response to the input that is similar but not identical to the one that is used in training (generalization). For a given set of training input-output pairs, this network provides a procedure for changing weights to classify the given input patterns correctly. The weight update algorithm is based on gradient descent method [25]. The network architecture is shown in Figure 1.

3.1. Dynamic Network. Neural network can be classified into two categories: (i) Static and (ii) Dynamic. In Static Neural Network, output can be calculated directly from input through feedforward network. But in Dynamic Network output depends on current or previous inputs, outputs, or states of network. Where the current output is function of current input and previous output, it is known as recurrent (feedback) network. Training process of Static and Dynamic Network can be differentiated by use of gradient or Jacobian which is computed. Dynamic Network contains delays and operates on a sequence of inputs. Dynamic Networks can have purely feedforward connections or they can have some recurrent connections. They can be trained using Dynamic Back Propagation Network [26].

The general equation of the net input $n^m(t)$ for mth layer is given by

$$n^m(t) = \sum_{l \in L_m^f} \sum_{d \in \mathrm{DL}_{m,l}} \mathrm{IW}^{m,l}(d)\, a^l(t-d) + \sum_{l \in I_m} \sum_{d \in \mathrm{DI}_{m,l}} \mathrm{IW}^{m,l}(d)\, b^l(t-d) + b^m, \tag{7}$$

where $b^l(t)$ is the lth input vector at time t, $\mathrm{IW}^{m,l}$ is the input weight between lth and mth layer, $\mathrm{LW}^{m,l}$ is the layer weight between lth and mth layer, and b^m is the bias vector for layer m. $\mathrm{DL}_{m,l}$ is the set of all delays in the Tapped Delay Line (TDL) between lth and mth layer and $\mathrm{DI}_{m,l}$ is the set of all delays in the TDL between input l and mth layer, I_m is the set of indices of input vectors that connect to layer m, and L_m^f is the set of indices of layers that connect to layer m. The output of mth layer at time t is computed as

$$a^m(t) = f^m\left(n^m(t)\right). \tag{8}$$

Network has a TDL on the input with $\mathrm{DI}_{1,1} = \{0, 1, 2\}$ and its output is represented as

$$\begin{aligned} a(t) &= f(n(t)) = \sum_d \mathrm{IW}(d) * p(t-d), \\ a(t) &= f\left[\mathrm{iw}_{1,1}(0)\, p(t) + \mathrm{iw}_{1,1}(1)\, p(t-1) \right. \\ &\quad \left. + \mathrm{iw}_{1,1}(2)\, p(t-2)\right]. \end{aligned} \tag{9}$$

One simple architecture with delay $\mathrm{DI}_{1,1} = \{0, 1, 2\}$ for feedforward Dynamic Network is shown in Figure 2 known as Dynamic Adaptive Linear Neuron.

3.2. Counter Propagation Network (CPN). It is multilayer feedforward network based on the combination of the input, competitive, and output layers. Model of CPN is instar-outstar. It is three-layer neural network that performs input-output data mapping, that is, producing output in the response to an input vector on the basis of Fuzzy Competitive Learning. In CPN the connection between input layer and competitive layer is instar structure and connection between competitive and output layer is outstar structure. Counter Propagation Network involves two-stage training process. In first stage input vector is clustered on the basis of Euclidean

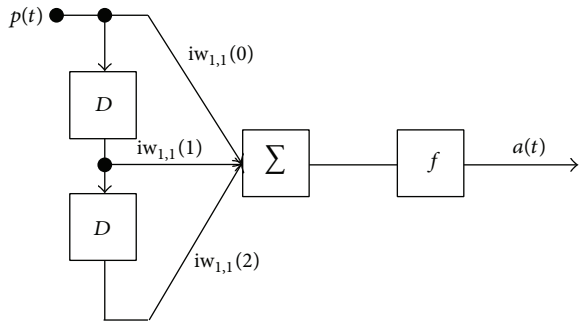


Figure 2: Simple architecture of Dynamic Adaptive Linear Neuron.

distance and the performance of network is improved using linear topology. Using Euclidean distance between input and weight vector, we evaluated BMN. In phase II, the desired response is obtained by adapting the weights from competitive layer to output layer [27]. Let $x = [x_1 \ x_2 \ \cdots \ x_n]$ and $y = [y_1^* \ y_2^* \ \cdots \ y_m^*]$ be input and desired vector, respectively, let v_{ij} be weight of input layer and competitive layer, where $1 \le i \le n$ and $1 \le j \le p$, and let w_{jk} be the weight between competitive layer and output layer, where $1 \le k \le m$. Euclidean distance between input vector x and weight vector v_{ij} is

$$D_j = \sum_{i=1}^{n} \left(x_i - v_{ij}\right)^2, \quad \text{where } j = 1, 2, \ldots, p. \tag{10}$$

The architecture of CPN is shown in Figure 3.

3.3. Fuzzy Competitive Learning (FCL). FCL is used for adjusting instar-outstar weights and update of BMN for discrete time nonlinear dynamical systems. Learning of CPN is divided into two phases: Fuzzy Competitive Learning phase (unsupervised) and Grossberg Learning phase (supervised) [28–31]. CPN is efficient network for mapping the input vector of size n placed in clusters of size m. Like traditional self-organizing map, the weight vector for every node is considered as a sample vector to the input pattern. The process is to determine which weight vector (say J) is more similar to the input pattern chosen as BMN. First phase is based on closeness between weight vector and input vector and second phase is weight update between competitive and output layer for desired response.

Let v_{ij} be weight of input node i to neuron j and let w_{jk} be the weight between competitive node j and neuron k. We propose a new learning rate calculation method for use of weight update:

$$\propto (J) = \frac{\sum_{i=1}^{n} \left(v_{iJ} - x_i\right)^2}{\sum_{j=1}^{m} \sum_{i=1}^{n} \left(v_{iJ} - x_i\right)^2}, \tag{11}$$

where J denotes the index of BMN and $\propto$ is known as Fuzzy Competitive Learning rate.

Fuzzy Competitive Learning phase is described as follows.

3.3.1. Training Algorithm of FCL

Phase I (for determination of BMN). There are steps for training of CPN as follows.

Step 1. Initialize instar-outstar weights.

Step 2. Perform Steps 3–8 until stopping criteria for Phase I training fail. The stopping condition may be fixed number of epochs or learning rate has reduced to negligible value.

Step 3. Perform Steps 4–6 for all input training vector X.

Step 4. Set the X-input layer activation to vector X.

Step 5. Compute the Best Matched Node (BMN) (J) using Euclidean distance; find the node Z_J whose distance from the input pattern is minimum. Euclidean distance is calculated as follows:

$$D_j = \sum_{i=1}^{n} \left(x_i - v_{ij}\right)^2, \quad \text{where } j = 1, 2, \ldots, p. \tag{12}$$

Minimum net input:

$$z_{inJ} = \sum_{i=1}^{n} x_i v_{iJ}. \tag{13}$$

Step 6. Calculate Fuzzy Competitive Learning (14) rate for weight update using BMN.

Step 7. We update weight for unit Z_J:

$$v_{iJ}\,(\text{new}) = v_{iJ}\,(\text{old}) + \propto h\left(J; i, j\right)\left(x_i - v_{iJ}\right), \tag{14}$$

where $i = 1, 2, 3, \ldots, n$ and $h(\cdot;\cdot)$ is neighborhood function around BMN. Consider

$$h\left(J; i, j\right) = \exp\left(\frac{-\left\|v_{ij} - v_{iJ}\right\|}{2\overline{\sigma}^2\,(t)}\right), \quad \text{where } h\left(J; i, j\right) \in [0, 1], \tag{15}$$

$$\overline{\sigma}\,(t) = \sigma_0 \exp\left(\frac{-t}{T}\right). \tag{16}$$

Step 8. Test the stopping criteria.

Phase II (to obtain desired response)

Step 1. Set activation function to (x, y) input and output layer, respectively.

Step 2. Update the BMN (J) (Step 5 from Phase I). Also update the weights into unit Z_J as

$$v_{iJ}\,(\text{new}) = v_{iJ}\,(\text{old}) + \alpha h\left(J; i, j\right)\left[x_i - v_{iJ}\,(\text{old})\right]. \tag{17}$$

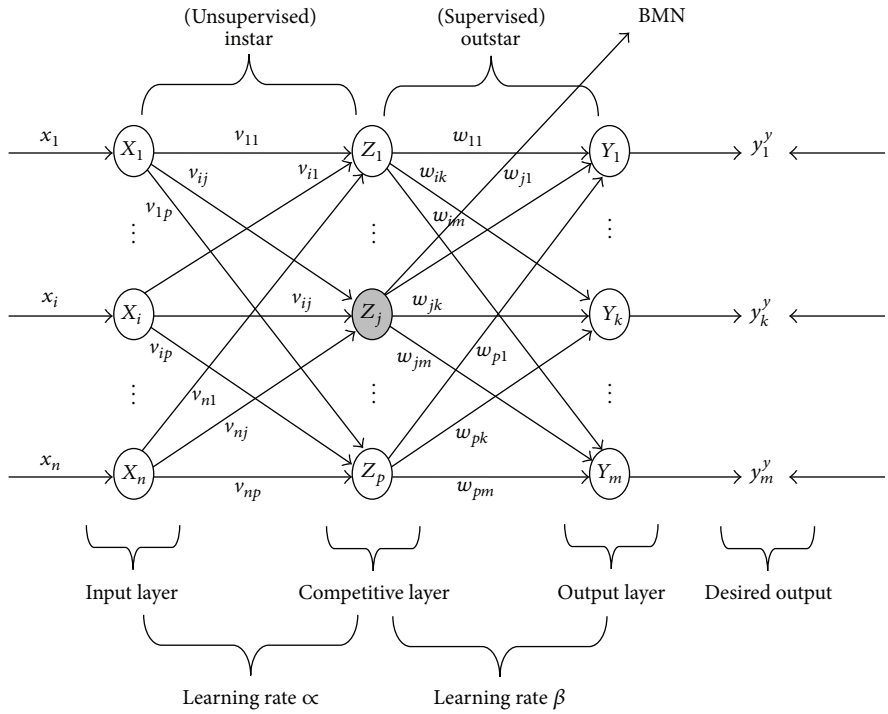


Figure 3: Architecture of Counter Propagation Network.

Step 3. Update the weights from node Z_J to the output unit as

$$w_{Jk}\,(\text{new}) = w_{Jk}\,(\text{old}) + \beta h\,(J; j, k)\left[y_j - w_{Jk}\,(\text{old})\right] \quad \text{for } 1 \le k \le m. \tag{18}$$

Step 4. Learning rate β is calculated as

$$\beta\,(J) = \frac{\sum_{k=1}^{m}\left(y_k - w_{Jk}\right)^2}{\sum_{j=1}^{p}\sum_{k=1}^{m}\left(y_k - w_{jk}\right)^2}. \tag{19}$$

Step 5. Decrease the rate of learning s.t. $\beta = \beta - \varepsilon$, where ε is small positive number.

Step 6. Test the stopping condition for Phase II (i.e., fixed number of epochs or its learning rate has reduced to negligible value).

3.3.2. Testing Phase (to Test FCPN)

Step 1. Set the initial weights, that is, the weights obtained during training.

Step 2. Apply FCPN to the input vector X.

Step 3. Find unit J that is closest to vector X.

Step 4. Set activations of output units.

Step 5. Apply activation function at y_k, where $y_k = \sum_j z_j w_{jk}$.

4. Dynamical Learning for CPN

Learning stabilities are fundamental issues for CPN; however there are few studies on the learning issue of FNN. BPN for learning is not always successful because of its sensitivity to learning parameters. Optimal learning rate always changes during the training process. Dynamical learning of CPN is carried out using Lemmas 1, 2, and 3.

Assumption. Let us assume $\phi(x)$ is a sigmoid function if it is bounded continuous and increasing function. Since input to the neural network in this model is bounded, we consider Lemmas 1 and 2 given below.

Lemma 1. *Let $\phi(x)$ be a sigmoid function and let Ω be a compact set in $\mathbb{R}^n$, and $f : \mathbb{R}^n \to \mathbb{R}$ on Ω is a continuous function and, for arbitrary $\in > 0$, $\exists$ integer N and real constants c_i, θ_i, and w_{ij}, $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, n$, such that*

$$\overline{f}\,(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{N} c_i \emptyset\left(\sum_{j=1}^{n} w_{ij} x_i - \theta_i\right) \tag{20}$$

satisfies

$$\max_{x\in\Omega}\left\|f\,(x_1, x_2, \ldots, x_n) - \overline{f}\,(x_1, x_2, \ldots, x_n)\right\| < \in . \tag{21}$$

Using Lemma 1, dynamical learning for a three-layer CPN can be formulated where the hidden layer transfer functions are $\phi(x)$ and transfer function at output layer is linear.

Let all the vectors be column vectors and superscript d_p^k refers to specific output vectors component.

Let $X = [x_1, x_2, \ldots, x_p] \in \mathbb{R}^{L\times P}$, $Y = [y_1, y_2, \ldots, y_p] \in \mathbb{R}^{H\times P}$, $O = [o_1, o_2, \ldots, o_p] \in \mathbb{R}^{K\times P}$, $D = [d_1, d_2, \ldots, d_p] \in \mathbb{R}^{K\times P}$ be the input, hidden, output, and desired vector, respectively, where L, H, and K denote the number of input, hidden, and output layer neurons. Let V and W represent the input-hidden and hidden-output layer weight matrix, respectively. The objectives of the network training minimize error function J, where J is

$$J = \frac{1}{2PK}\sum_{p=1}^{P}\sum_{k=1}^{K}\left(o_p^k - d_p^k\right)^2. \tag{22}$$

4.1. Optimal Learning of Dynamical System. Consider a three-layer CPN; the network error matrix is defined by the error between differences of desired and FCPN output at any iteration and given as

$$E_t = O_t - D = W_tY_t - D = W_tY_tX - D. \tag{23}$$

The objective of the network for minimization of error given in (24) is defined as follows:

$$\text{s.t.} \quad J = \frac{1}{2PK}T_r\left(E_tE_t^T\right), \tag{24}$$

where T_r represent the trace of matrix. Using gradient descent method updated weight is given by

$$\begin{aligned} W_{t+1} &= W_t - \beta_t\frac{\partial J}{\partial W_t}, \\ V_{t+1} &= V_t - \beta_t\frac{\partial J}{\partial V_t}, \end{aligned} \tag{25}$$

or

$$\begin{aligned} W_{t+1} &= W_t - \frac{\beta_t}{PK}E_tY_t^T, \\ V_{t+1} &= V_t - \frac{\beta_t}{PK}W_t^TE_tX^T. \end{aligned} \tag{26}$$

Using (23)–(26), we have

$$E_{t+1}E_{t+1}^T = \left(W_{t+1}Y_{t+1} - D\right)\left(W_{t+1}Y_{t+1} - D\right)^T. \tag{27}$$

To obtain minimum error for multi-layer FNN in above equation (28) after simplification, we have

$$\begin{aligned} J_{t+1} - J_t &= \frac{1}{2PK}g\left(\beta\right), \\ J &= \frac{1}{2PK}\sum_{p=1}^{P}\sum_{k=1}^{K}\left(o_p^k - d_p^k\right)^2, \\ E_{t+1}E_{t+1}^T &= \left(W_{t+1}V_{t+1}X - D\right)\left(W_{t+1}V_{t+1}X - D\right)^T \\ &= \left[\left(W_t - \frac{\beta_t}{PK}E_tY_t^T\right)\left(V_t - \frac{\beta_t}{PK}W_t^TE_tX^T\right)X\right. \\ &\quad \left. - D\right]\left[\left(W_t - \frac{\beta_t}{PK}E_tY_t^T\right)\left(V_t - \frac{\beta_t}{PK}W_t^TE_tX^T\right)X\right. \\ &\quad \left. - D\right]^{\mathbf{T}} = \left[E_t - \frac{\beta_t}{PK}\left(W_tW_t^TE_tX^TX + E_tY_t^TV_tX\right)\right. \\ &\quad \left. + \frac{\beta_t^2}{(PK)^2}E_tY_t^TW_t^TE_tX^TX\right] * \left[E_t\right. \\ &\quad \left. - \frac{\beta_t}{PK}\left(W_tW_t^TE_tX^TX + E_tY_t^TV_tX\right)\right. \\ &\quad \left. + \frac{\beta_t^2}{(PK)^2}\left(E_tY_t^TW_t^TE_tX^TX\right)\right]^T = E_tE_t^T \\ &\quad - \frac{\beta_t}{PK}\left[E_t\left(W_tW_t^TE_tX^TX\right)^T + E_t\left(E_tY_t^TV_tX\right)^T\right. \\ &\quad \left. + \left(W_tW_t^TE_tX^TXE_t^T\right) + \left(E_tY_t^TV_tXE_t^T\right)\right] \\ &\quad + \frac{\beta_t^2}{(PK)^2}\left[E_t\left(E_tY_t^TW_t^TE_tX^TX\right)^T\right. \\ &\quad + E_tY_t^TW_t^TE_tX^TXE_t^T \\ &\quad + E_tY_t^TV_tX\left(W_tW_t^TE_tX^TX\right)^T \\ &\quad + W_tW_t^TE_tX^TX\left(E_tY_t^TV_tX\right)^T \\ &\quad + E_tY_t^TV_tX\left(E_tY_t^TV_tX\right)^T \\ &\quad \left. + W_tW_t^TE_tX^TX\left(W_tW_t^TE_tX^TX\right)^T\right] \\ &\quad - \frac{\beta_t^3}{(PK)^3}\left[W_tW_t^TE_tX^TX\left(E_tY_t^TW_t^TE_tX^TX\right)\right. \\ &\quad + E_tY_t^TV_tX\left(E_tY_t^TW_t^TE_tX^TX\right)^T \\ &\quad + E_tY_t^TW_t^TE_tX^TX\left(W_tW_f^TE_tX^TX\right)^T \\ &\quad \left. + E_tY_t^TW_t^TE_tX^TX\left(E_tY_t^TV_tX\right)^T\right] \\ &\quad + \frac{\beta_t^4}{(PK)^4}\left[E_tY_t^TW_t^TE_tX^TX\left(E_tY_t^TW_t^TE_tX^TX\right)^T\right]. \end{aligned} \tag{28}$$

For simplification, omit subscript t, we have; that is,

$$J_{t+1} - J_t = \frac{1}{2PK}\left(A\beta^4 + B\beta^4 + C\beta^4 + M\beta\right) \tag{29}$$

where

$$\begin{aligned} A &= \frac{1}{(PK)^4}T_r\left[EY^TWEX^TX\left(EY^TW^TEX^TX\right)^T\right], \\ B &= \frac{1}{(PK)^3}T_r\left[WW^TEX^TX\left(EY^TW^TEX^TX\right)^T\right. \end{aligned}$$

$$
\begin{aligned}
&+ EY^T VX\left(EY^T W^T EX^T X\right)^T \\
&+ EY^T W^T EX^T X\left(WW^T EX^T X\right)^T \\
&+ EY^T W^T EX^T X\left(EY^T VX\right)\Big], \\
C = \frac{1}{(PK)^2} T_r &\Big[\left(EY^T W^T EX^T X\right)^T \\
&+ EY^T WEX^T XE^T + EY^T VX\left(WW^T EX^T X\right)^T \\
&+ WW^T EX^T X + \left(EY^T VX\right)^T \\
&+ EY^T VX\left(EY^T VX\right)^T \\
&+ WW^T EX^T X\left(WW^T EX^T X\right)^T\Big], \\
M = \frac{1}{PK} T_r &\Big[E\left(WW^T EX^T X\right)^T + E\left(EY^T VX\right)^T \\
&+ WW^T EX^T X + EY^T VXE^T\Big],
\end{aligned} \tag{30}
$$

where

$$
g(\beta) = \left(A\beta^4 + B\beta^3 + C\beta^2 + M\beta\right), \tag{31}
$$

$$
\frac{\partial g}{\partial \beta} = 4A\left(\beta^3 + a\beta^2 + b\beta + c\right), \tag{32}
$$

where $a = 3B/4A$, $b = 2C/4A$, and $c = M/4A$.

Lemma 2. *For solution of general real cubic equation, one uses the following lemma:*

$$
\begin{aligned}
&f(x) = x^3 + ax^2 + bx + c, \\
&Let\ D = -27c^2 + 18cab + a^2b^2 - 4a^3b^3 - 4b^3,
\end{aligned} \tag{33}
$$

where D is discriminant of $f(x)$.

Then we have the following:

(1) *If $D < 0$, $f(x)$ has one real root.*

(2) *If $D \geq 0$, $f(x)$ has three real roots:*

   (a) *$D > 0$; $f(x)$ has three different real roots,*

   (b) *$D = 0$, $6b - 2a^2 \# 0$; $f(x)$ has one single root and one multiple root,*

   (c) *$D = 0$, $66 - 2a^2 = 0$; $f(x)$ has one root of three multiplicities.*

Lemma 3. *For given polynomial $g(\beta)$ given (31) if optimum $\beta = \{\beta_i \mid g(\beta_i) = \min(g(\beta_1), g(\beta_2), g(\beta_3)),\ i \in \{1, 2, 3\}\}$, where β_i is real root of $\partial g/\partial \beta$, the optimum (β) is the optimal learning rate and this learning process is stable.*

Proof. To find stable learning range of β, consider that Lyapunov function is

$$
\begin{aligned}
V_t &= J_t^2, \\
\Delta V_t &= J_{t+1}^2 - J_t^2 \quad \text{if } \Delta V_t < 0,
\end{aligned} \tag{34}
$$

and then dynamical system is guaranteed to be stable if $\Delta V_t < 0$; that is, $J_{t+1} - J_t = (1/2PK)(A\beta^4 + B\beta^4 + C\beta^4 + M\beta) < 0$ (29), where β is learning rate, since in the training process input matrix remains the same during the whole training process. To find the range of β, which satisfy $g(\beta) = (A\beta^4 + B\beta^3 + C\beta^2 + M\beta) < 0$. Since $\partial g/\partial \beta$, where has at least one real root (Lemma 2) and one of them must give optimum $g(\beta)$. Obviously minimum value of $g(\beta)$ gives the largest reduction in J_t at each step of learning process. Equation (31) shows that $g(\beta)$ has two or four real roots, one including $\beta = 0$ (Lemma 2), such that minimum value of β shows largest reduction in error at two successive times and minimum value is obtained by differentiating (31) with respect to β we have from (32):

$$
\frac{\partial g}{\partial \beta} = 4A\left(\beta^3 + a\beta^2 + b\beta + C\right), \tag{35}
$$

where $a = 3B/4A$, $b = 2C/4A$, and $c = M/4A$.

Solving $\partial g/\partial \beta = 0$ gives β which minimizes error in (5). □

5. Simulation Results

To demonstrate the effectiveness and merit of proposed FCPN, simulation results are presented and discussed. Four different nonlinear dynamical systems and one general benchmark problem known as Box-Jenkins model with time series data are considered.

5.1. Performance Criteria. For accessing the performance criteria of FCPN and comparing with Dynamic and Back Propagation Network, we have evaluated various errors as given below. In case of four dynamical models, it is recommended to access criterion such as subset of the following.

Given N pair of data points $(y(k), \overline{y}(k))$, where $y(k)$ is output of system and $\overline{y}(k)$ is output of controller, the Maximum Absolute Error (MAE) is

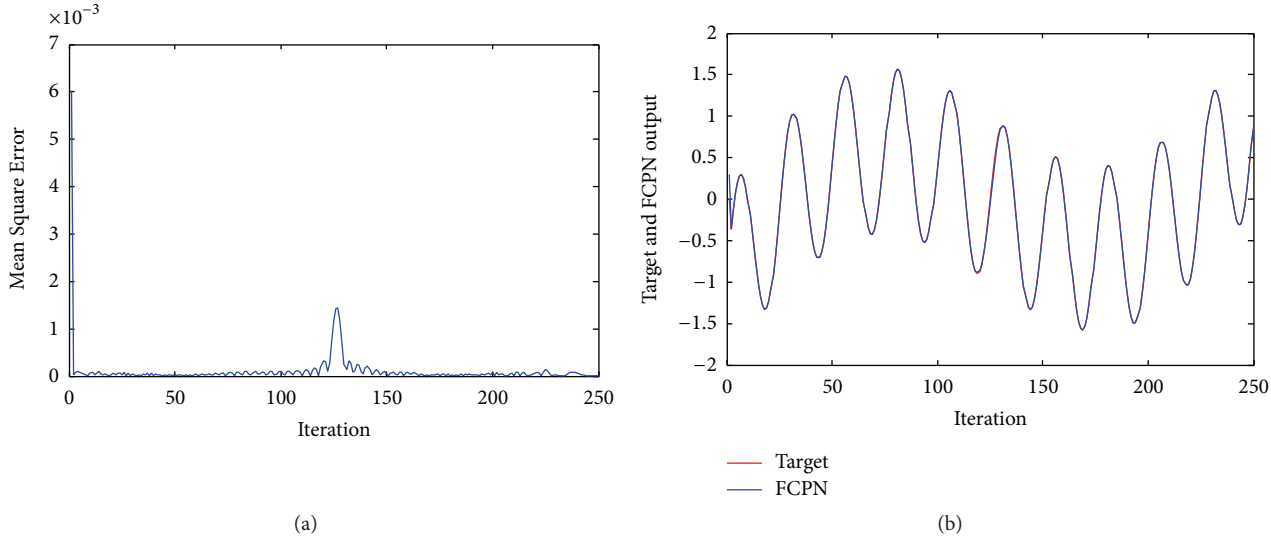
$$
J_{\mathrm{MAE}} = J_{\max} = \max_{1 \leq k \leq N} \left|y(k) - \overline{y}(k)\right|, \tag{36}
$$

the Sum of Squared Errors (SSE) is

$$
J_{\mathrm{SSE}} = \sum_{k=1}^{N} \left(y(k) - \overline{y}(k)\right)^2, \tag{37}
$$

the Mean Squared Error (MSE) is

$$
J_{\mathrm{MSE}} = \frac{1}{N} \sum_{k=1}^{N} \left(y(k) - \overline{y}(k)\right)^2, \tag{38}
$$



(a) (b)

Figure 4: (a) Mean Square Error of system (Example 1) using FCPN. (b) Performance of the controller using FCPN algorithm for Example 1.

and/or the Root Mean Squared Error (RMSE) is

$$J_{\mathrm{RMSE}} = \sqrt{J_{\mathrm{MSE}}}. \tag{39}$$

The measure Variance Accounting Factor (VAF) is

$$J_{\mathrm{VAF}} = \left(1 - \frac{\mathrm{Var}\left(y\left(k\right) - \overline{y}\left(k\right)\right)}{\mathrm{Var}\left(y\left(k\right)\right)}\right) 100\%. \tag{40}$$

A related measure is the Normalized Mean Squared Error (NMSE):

$$J_{\mathrm{NMSE}} = \frac{\sum_{k=1}^{N}\left(y\left(k\right) - \overline{y}\left(k\right)\right)^2}{\sum_{k=1}^{N}\left(y\left(k\right) - \overline{y}\right)^2}, \tag{41}$$

and the Best Fit Rate (BFR) is

$$J_{\mathrm{BFR}} = \left(1 - \frac{\sqrt{\sum_{k=1}^{N}\left(y\left(k\right) - \overline{y}\left(k\right)\right)^2}}{\sum_{k=1}^{N}\left(y\left(k\right) - \widehat{y}\right)^2}\right) 100\%. \tag{42}$$

Example 1. The nonlinear dynamical system [24] is governed by the following difference equation:

$$y\left(k+1\right) = 0.3y\left(k\right) + 0.6y\left(k-1\right) + f\left[u\left(k\right)\right]. \tag{43}$$

The nonlinear function in the system in this case is $f(u) = u^3 + 0.3u^2 - 0.4u$, where y and u are uniformly distributed in $[-2, 2]$. The objective of Example 1 is to control the system to track reference output given as 250 sample data points. The model has two inputs $y(k)$ and $u(k)$ and single output $y(k+1)$ and the system identification was initially performed with the system input being uniformly distributed over $[-2, 2]$. The FCPN controller uses two inputs $y(k)$ and $u(k)$ to produce output $\overline{y}(k+1)$. FCPN model governed by the difference equation $\overline{y}(k+1) = 0.3y(k) + 0.6y(k-1) + N[u(k)]$ was used. Figures 4(a) and 4(b) show the error $e(k+1) = y(k+1) - \overline{y}(k+1)$ and outputs of the dynamical system and the FCPN.

Table 1: Calculated various errors for Example 1.

NN model	Different error calculation				
	MAE	SSE	MSE	RMSE	NMSE
FCPN	1.3364	0.1143	$4.5709e-004$	0.0214	$7.1627e-004$
Dynamic	5.8231	3.1918	0.0128	0.1130	0.0028
BPN	38.0666	48.1017	0.4810	0.6936	0.1035

As can be seen from the figure, the identification error is small even when the input is changed to a sum of two sinusoids $u(k) = \sin(2\pi k/250) + \sin(2\pi k/25)$ at $k = 250$.

Table 1 includes various calculated errors of different NN models for Example 1. It can be seen from the table that various errors calculated for FCPN are minimum as compared to the DN and BPN.

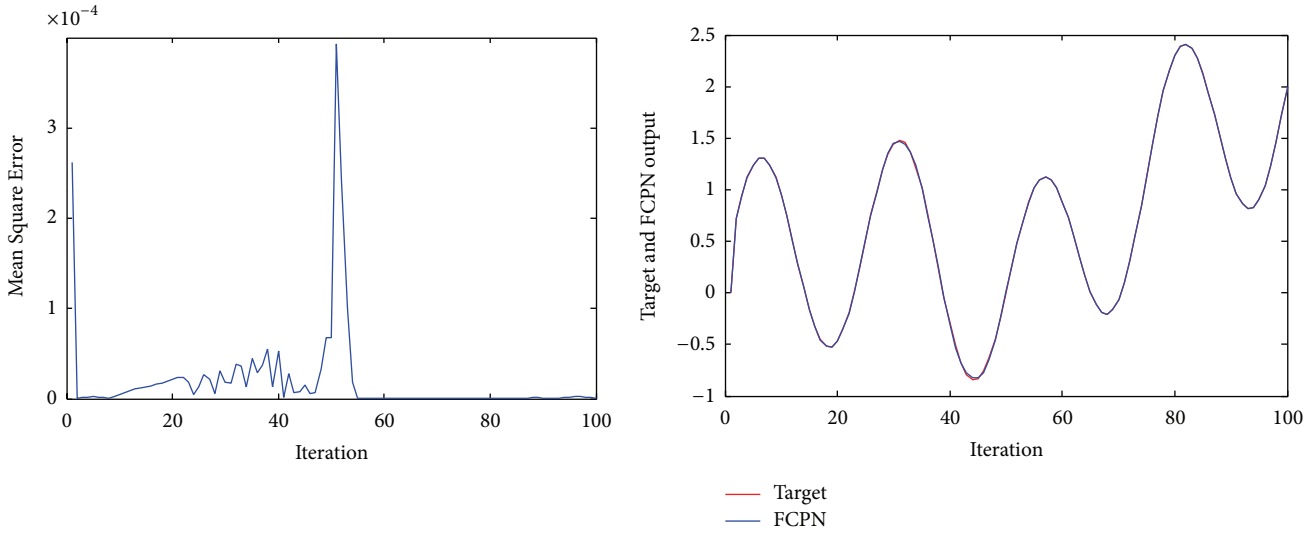
Example 2. The nonlinear dynamical system [24] is governed by the following difference equation:

$$y\left(k+1\right) = f\left[y_p\left(k\right), y_p\left(k-1\right)\right] + u\left(k\right), \tag{44}$$

where

$$f\left[y_p\left(k\right), y_p\left(k-1\right)\right] = \left[\frac{y_p\left(k\right) y_p\left(k-1\right)\left[y_p\left(k\right) + 2.5\right]}{1 + y_p^2\left(k\right) + y_p^2\left(k-1\right)}\right]. \tag{45}$$

The objective of (44) is to control the system to track reference output given 100 sample data points. The model has two inputs $y(k)$ and $u(k)$ and single output $y(k+1)$ and the system identification was initially performed with the system input being uniformly distributed over $[-2, 2]$. Training and testing samples contain 100 sample data points. The FCPN controller uses two inputs $y(k)$ and $u(k)$ to output $\overline{y}(k+1)$. FCPN



(a) (b)

Figure 5: (a) Mean Square Error of system (Example 2) using FCPN. (b) Performance of the controller using FCPN algorithm for Example 2.

Table 2: Calculated various errors for Example 2.

NN models	Different error calculation				
	MAE	SSE	MSE	RMSE	NMSE
FCPN	0.1956	0.0032	$3.2328e-005$	0.0057	$2.7164e-005$
Dynamic	9.5152	2.9856	0.0299	0.1728	0.0308
BP	10.1494	4.5163	0.0452	0.2125	0.0323

Table 3: Calculated various errors and BFR for Example 3.

NN model	MAE	SSE	MSE	RMSE	NMSE
FCPN	0.3519	0.0032	$3.1709e-005$	0.0056	$5.6371e-006$
Dynamic	5.5652	3.6158	0.0362	0.1902	0.0223
BP	42.6524	46.9937	0.4699	0.6855	0.2892

network discussed earlier is used to identify the system from input-output data and is described by the equation

$$\overline{y}(k+1) = N\left[y(k), y(k-1)\right] + u(k). \quad (46)$$

For FCPN, the identification process involves the adjustment of the weights of N using FCL. FCPN identifier needs some prior information concerning the input-output behavior of the system before identification can be undertaken. FCL is used to adjust the weights of the neural network so that the error $e(k+1) = y(k+1) - \overline{y}(k+1)$ is minimized as shown in Figure 5(a). The behavior of the FCPN model for (16) is shown in Figure 5(b). The input $u(k)$ was assumed to be a random signal uniformly distributed in the interval $[-2, 2]$. The weights in the neural network were adjusted.

Table 2 includes various calculated errors of different NN models for Example 2. It can be seen from the table that various errors calculated for FCPN are minimum as compared to the DN and BPN.

Example 3. The nonlinear dynamical system [23] is governed by the following difference equation:

$$y(k+1) = \frac{y(k)}{1+y(k)^2} + u^3(k), \quad (47)$$

which corresponds to $f[y(k)] = y(k)/(1+y(k)^2)$ and $g[u(k)] = u^3(k)$. FCPN network equation (47) is used to identify the system from input-output data. Weights in the neural networks were adjusted for every instant of discrete time steps.

The objective of (47) is to control the system to track reference output given 100 sample data points. The model has two inputs $y(k)$ and $u(k)$ and single output $y(k+1)$ and the system identification was initially performed with the system input being uniformly distributed over $[-2, 2]$. Training and testing samples contain 250 sample data points. The FCPN controller uses two inputs $y(k)$ and $u(k)$ to output $\overline{y}(k+1)$. The input data is generated using uniform distribution in interval $[-2, 2]$. The function $\overline{f}$ obtained by FCPN is used to adjust the weights of the neural network so that the error $e(k+1) = y(k+1) - \overline{y}(k+1)$ is minimized as shown in Figure 6(a). In Figure 6(b), the desired outputs of the system as well as the FCPN model are shown and are seen to be indistinguishable.

Table 3 includes various calculated errors of different NN models for Example 3. It can be seen from the table that various errors calculated for FCPN are minimum as compared to the DN and BPN.

Example 4. The nonlinear dynamical system [24] is governed by the difference equation (48). Generalized form of this equation is Model IV. In this example, the system is assumed to be of the form

$$y_p(k+1) = f\left[y_p(k), y_p(k-1), y_p(k-2), u(k), u(k-1)\right], \quad (48)$$

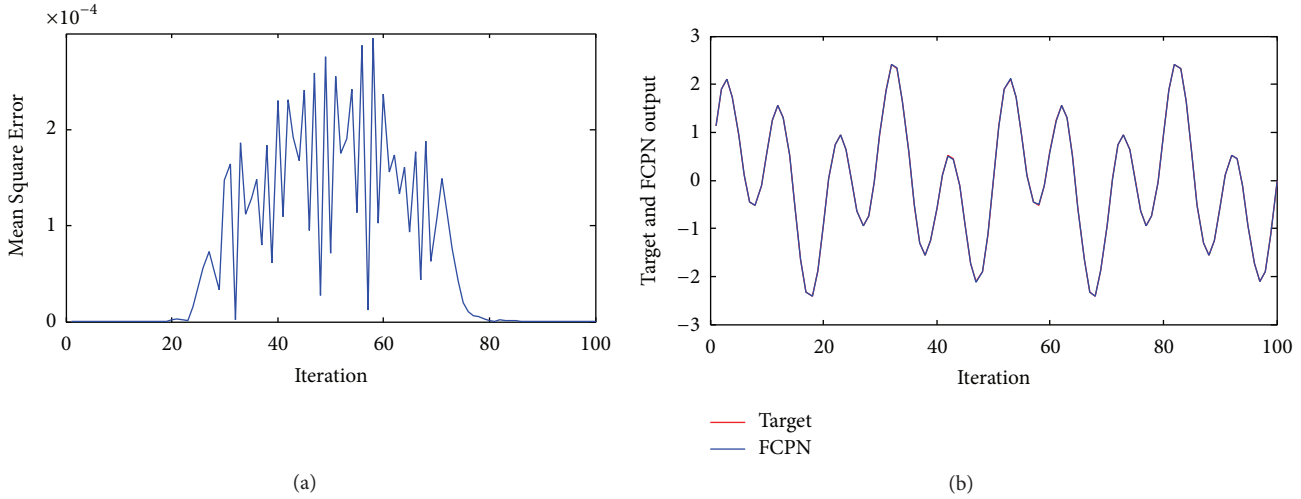


Figure 6: (a) Mean Square Error of system (Example 3) using FCPN. (b) Performance of the controller using FCPN algorithm for Example 3.

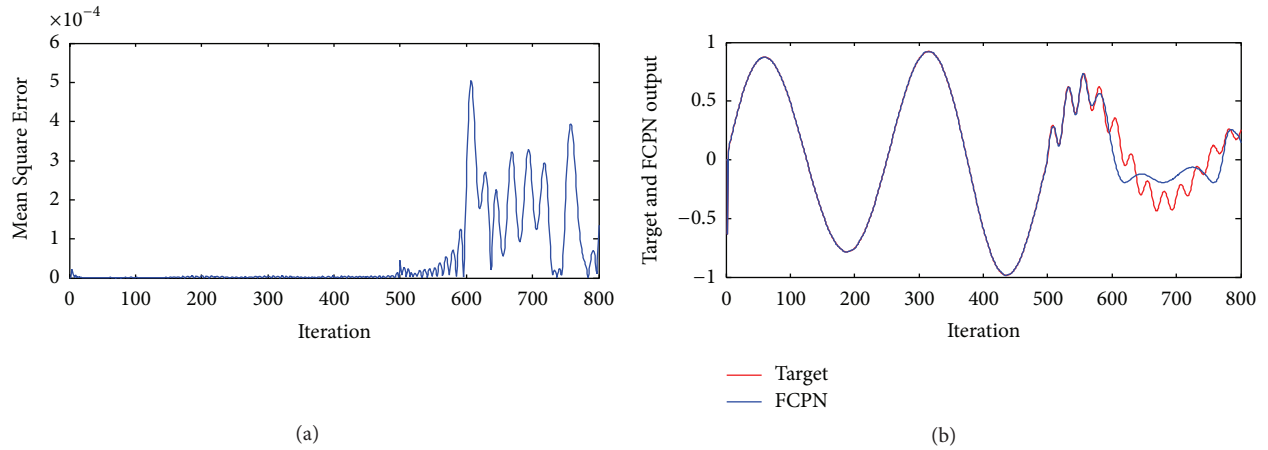


Figure 7: (a) Mean Square Error of system (Example 4) using FCPN. (b) Performance of the controller using FCPN algorithm for Example 4.

where the unknown function f has the form

$$f[x_1, x_2, x_3, x_4, x_5] = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_3^2 + x_2^2}. \quad (49)$$

In the identification model, which is approximated by FCPN, Figure 7(b) shows the output of the system and the model when the identification procedure carried random input signal uniformly distributed in the interval $[-1, 1]$. The performance of the model is studied and the error $e(k+1) = N[y_p(k+1)]$ is minimized as shown in Figure 7(a). In Figure 7(b), the outputs of the system as well as output of FCPN model are shown. Input to the system and the identified model is given by $u(k) = \sin(2\pi k/250)$ for $k \leq 500$ and $u(k) = 0.8\sin(2\pi k/250) + 0.2\sin(2\pi k/25)$ for $k > 500$.

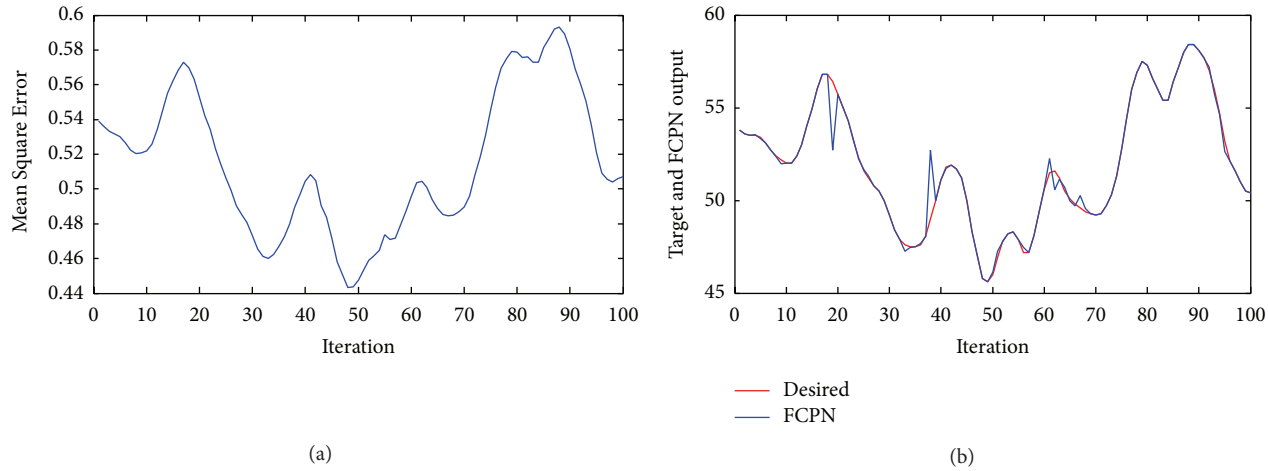
Table 4 includes various calculated errors of different NN models for Example 4. It can be seen from the table that various errors calculated for FCPN are minimum as compared to the DN and BPN.

Table 4: Calculated various errors and BFR for Example 4.

NN models	MAE	SSE	MSE	RMSE	NMSE
FCPN	30.4348	2.1931	0.0027	0.0524	0.0186
Dynamic	32.8952	6.4059	0.0080	0.0895	0.0276
BP	53.1988	13.4174	0.0168	0.1295	0.0844

Example 5. In this example we have used Box-Jenkins time series [32], of 296 pairs of data measured from a gas furnace system with single input $u(t)$ being gas flow rate and single output $y(t)$ being CO_2 concentration in outlet gas. Training samples and testing samples contained 196 and 100 data points, respectively. The FCPN uses the two inputs, the current state $y(k)$ and the desired state $y^d(k)$, to produce an output which is $\overline{y}(k)$ [33].

Figure 8(a) shows the mean squared control errors of FCPN methods. Figure 8(b) shows the performance of the controller with FCPN algorithm. Result shows that FCPN algorithm enable us to appropriate the approximation using



(a) (b)

Figure 8: (a) Mean Square Error of system (Example 5) using FCPN. (b) Performance of the controller using FCPN algorithm for Example 5.

Table 5: BFR (%) values of NN models for various examples.

NN models	BFR (%)			
	Example 1	Example 2	Example 3	Example 4
FCPN	97.32	99.48	99.76	86.35
DN	94.69	79.32	85.08	83.39
BPN	67.83	79.28	46.2234	70.94

fuzzy learning as given in equation (11) of Box Jenkins time series data, based on calculation of BMN.

Table 5 shows BFR (%) various NN models for all Examples 1–5, respectively. It can be observed from Table 5 that the Best Fit Rate found for FCPN is maximum as compared to DN and BPN which shows better performance of the FCPN network for nonlinear system.

6. Conclusions

FCPN is a neural network control method which was developed and presented. It is based on the concept of combining FCL algorithm and CPN. The key ideas explored are the use of the FCL algorithm for training the weights of the instar-outstar. The performances of the FCPN controller based training are tested using four nonlinear dynamical systems and on time series (Box-Jenkins) data and compared with the Dynamic Network and standard Back Propagation algorithm. The comparative performances of the FCPN algorithm and Dynamic Network, such as the number of iterations and performance functions like MAE, MSE, SSE, NMSE, BFR, and so forth, of FCPN and Dynamic Network error are summarized in Tables 1–4. It can be seen that the FCPN algorithm gives minimum errors as compared to the Dynamic Network and standard Back Propagation Network for all four models of nonlinear dynamical systems and Box-Jenkins time series data. Results obtained from FCPN were compared for various errors and it is clearly shown that FCPN works much better for the control of nonlinear dynamical systems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors gratefully acknowledged the financial assistance provided by the All India Council of Technical Education (AICTE) in the form of Research Promotion Scheme (RPS) project in 2012.

References

[1] T. Soderstrom and P. Stoica, *System Identification*, Prentice Hall, New York, NY, USA, 1989.

[2] S. A. Billings, *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*, Wiley, Chichester, UK, 2013.

[3] M. Liu, "Decentralized control of robot manipulators: nonlinear and adaptive approaches," *IEEE Transactions on Automatic Control*, vol. 44, no. 2, pp. 357–363, 1999.

[4] C.-M. Lin, A.-B. Ting, M.-C. Li, and T.-Y. Chen, "Neural-network-based robust adaptive control for a class of nonlinear systems," *Neural Computing and Applications*, vol. 20, no. 4, pp. 557–563, 2011.

[5] I. Rivals and L. Personnaz, "Nonlinear internal model control using neural networks: application to processes with delay and design issues," *IEEE Transactions on Neural Networks*, vol. 11, no. 1, pp. 80–90, 2000.

[6] I. Kanellakopoulos, P. V. Kokotovic, and A. S. Morse, "Systematic design of adaptive controllers for feedback linearizable systems," *IEEE Transactions on Automatic Control*, vol. 36, no. 11, pp. 1241–1253, 1991.

[7] P. Kokotovic, "The joy of feedback: nonlinear and adaptive," *IEEE Control Systems*, vol. 12, no. 3, pp. 7–17, 1992.

[8] H. Elmali and N. Olgac, "Robust output tracking control of nonlinear MIMO systems via sliding mode technique," *Automatica*, vol. 28, no. 1, pp. 145–151, 1992.

[9] N. Sadati and R. Ghadami, "Adaptive multi-model sliding mode control of robotic manipulators using soft computing," *Neurocomputing*, vol. 71, no. 13–15, pp. 2702–2710, 2008.

[10] A. Kroll and H. Schulte, "Benchmark problems for nonlinear system identification and control using Soft Computing methods: need and overview," *Applied Soft Computing Journal*, vol. 25, pp. 496–513, 2014.

[11] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[12] A. Bortoletti, C. Di Fiore, S. Fanelli, and P. Zellini, "A new class of Quasi-Newtonian methods for optimal learning in MLP-networks," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 263–273, 2003.

[13] G. Lera and M. Pinzolas, "Neighborhood based Levenberg-Marquardt algorithm for neural network training," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1200–1203, 2002.

[14] A. K. Shah and D. M. Adhyaru, "Clustering based multiple model control of hybrid dynamical systems using HJB solution," *Applied Soft Computing*, vol. 31, pp. 103–117, 2015.

[15] Y. Cui, H. Zhang, Y. Wang, and Z. Zhang, "Adaptive neural dynamic surface control for a class of uncertain nonlinear systems with disturbances," *Neurocomputing*, vol. 165, pp. 152–158, 2015.

[16] R. Rakkiyappan, N. Sakthivel, and J. Cao, "Stochastic sampled-data control for synchronization of complex dynamical networks with control packet loss and additive time-varying delays," *Neural Networks*, vol. 66, pp. 46–63, 2015.

[17] B. Kiumarsi, F. L. Lewis, and D. S. Levine, "Optimal control of nonlinear discrete time-varying systems using a new neural network approximation structure," *Neurocomputing*, vol. 156, pp. 157–165, 2015.

[18] B. M. Al-Hadithi, A. Jiménez, and R. G. López, "Fuzzy optimal control using generalized Takagi-Sugeno model for multivariable nonlinear systems," *Applied Soft Computing Journal*, vol. 30, pp. 205–213, 2015.

[19] M. A. González-Olvera and Y. Tang, "Identification of nonlinear discrete systems by a state-space recurrent neurofuzzy network with a convergent algorithm," *Neurocomputing*, vol. 148, pp. 318–325, 2015.

[20] J. F. de Canete, P. Del Saz-Orozco, I. Garcia-Moral, and S. Gonzalez-Perez, "Indirect adaptive structure for multivariable neural identification and control of a pilot distillation plant," *Applied Soft Computing*, vol. 12, no. 9, pp. 2728–2739, 2012.

[21] M. Cai and Z. Xiang, "Adaptive neural finite-time control for a class of switched nonlinear systems," *Neurocomputing*, vol. 155, pp. 177–185, 2015.

[22] Y. Li, S. Tong, and T. Li, "Adaptive fuzzy backstepping control design for a class of pure-feedback switched nonlinear systems," *Nonlinear Analysis: Hybrid Systems*, vol. 16, pp. 72–80, 2015.

[23] T.-Y. Kuc, J. S. Lee, and K. Nam, "An iterative learning control theory for a class of nonlinear dynamic systems," *Automatica*, vol. 28, no. 6, pp. 1215–1221, 1992.

[24] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[25] R. Hecht-Nielsen, "Theory of the back propagation neural network," *Neural Networks*, vol. 1, pp. 593–605, 1989.

[26] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. De Jesus, *Neural Network Design*, Cengage Learning, 2nd edition, 2014.

[27] F.-J. Chang and Y.-C. Chen, "A counterpropagation fuzzy-neural network modeling approach to real time streamflow prediction," *Journal of Hydrology*, vol. 245, no. 1–4, pp. 153–164, 2001.

[28] A. Dwivedi, N. S. C. Bose, A. Kumar, P. Kandula, D. Mishra, and P. K. Kalra, "A novel hybrid image compression technique: wavelet-MFOCPN," in *Proceedings of the 9th Asian Symposium on Information Display (ASID '06)*, pp. 492–495, 2006.

[29] C. J. C. Burges, P. Simard, and H. S. Malvar, *Improving Wavelet Image Compression with Neural Networks*, Microsoft Research, Redmond, Wash, USA, 2001.

[30] D. Woods, "Back and counter propagation aberrations," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 473–479, 1988.

[31] D. Mishra, N. S. C. Bose, A. Tolambiya et al., "Color image compression with modified forward-only counterpropagation neural network: improvement of the quality using different distance measures," in *Proceedings of the 9th International Conference on Information Technology (ICIT '06)*, pp. 139–140, IEEE, Bhubaneswar, India, December 2006.

[32] G. E. Box and G. M. Jenkins, *Time Series Analysis Forecasting and Control*, Holden Day, San Francisco, Calif, USA, 1970.

[33] J. Kim and N. Kasabov, "HyFIS: adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems," *Neural Networks*, vol. 12, no. 9, pp. 1301–1319, 1999.