TJER 2012, Vol. 9, No. 2, 80-89

RBF Neural Network Approach for Identification and Control of DC Motors

EA Feilat*[a], EK Ma'aita[b]

[a]*Department of Electrical & Computer Engineering, College of Engineering, Sultan Qaboos University, Muscat, Oman*
[b]*Department of Electrical Engineering, College of Engineering, Tafila Technological University, Jordan*

Received 23 January 2011; accepted 22 February 2012

Abstract: In this paper, a neural network approach for the identification and control of a separately excited direct (DC) motor (SEDCM) driving a centrifugal pump load is applied. In this application, two radial basis function neural networks (RBFNN) are used: The first is a RBFNN identifier trained offline to emulate the dynamic performance of the DC motor-load system. The second is a RBFNN controller, which is trained to make the motor speed follow a selected reference signal. Two RBFNN control schemes are proposed using direct inverse and internal model control schemes. The performance of the RBFNN identifier and controller is investigated in terms of step response, sharp changes in speed trajectory, and sudden load change, as well as changes in motor parameters. The performance of RBFNN in system identification and control has been compared with the performance of the well-known back-propagation neural network (BPNN). The simulation results show that both of the BPNN and RBFNN controllers exhibit excellent dynamic response, adapt well to changes in speed trajectory and load connected to the motor, and adapt to the variations of motor parameters. Furthermore, the simulation results show that the step response of RBFNN internal model and direct inverse controllers are identical.

Keywords: DC motors, Identification, Neurocontrol, Neural network

استخدام الشبكات العصبية ذوات الاقترانات الشعاعية للتمييز والتحكم بالمحركات الكهربائية ذات التيار المباشر

اا فيلات* و إك معايطه

الملخص: يقدم هذا البحث استخدام الشبكات العصبية في تمييز المحركات الكهربائية ذات التيار المباشر والتحكم بها. تم بناء شبكتين عصبيتين واحدة لتمييز المحرك والتعرف على أدائه الديناميكي، وشبكه أخرى للتحكم بسرعته وجعلها موافقة لإشارة مرجعية مختارة. وقد تم بناء نموذجين للتحكم، أحداهما النموذج الانقلابي المباشر والآخر النموذج الداخلي. تم بحث أداء الشبكتين العصبيتين باستخدام الاستجابة الخطوية، والتغيير المفاجئ لمسار السرعة، والتغيير المفاجئ للحمل وتغيير ثوابت المحرك. وقد تمت مقارنة أداء الشبكتين العصبيتين المقترحتين مع أداء شبكة عصبية مرجعية ذات انتشار مرتد، وبينت نتائج البحث أن المحكمات الشبكية تميزت بأداء ديناميكي متميز ومتكيف مع التغير المفاجيء لمسار السرعة و التحمل كما تتكيف مع التغير في معاملات المحرك، وكذلك بينت الدراسة أن أداء المحكمين الشبكيين الانقلابي والنموذج الداخلي متماثلين.

الكلمات المفتاحية: المحركات الكهربائية ذات التيار المباشر، التشخيص، التحكم الشبكي العصبي، الشبكات العصبية

1. Introduction

Direct current (DC) motors have been extensively used in automation systems because of their favorable torque and robust speed control characteristics. Examples of DC motor applications in industry include their use in robotic manipulators, automation systems, steel manufacture, paper processing and mining industries. DC motors are customarily modeled linearly to enable the application of linear control theory in controller design. However, most of the existing linear controllers generally do not lead to good tracking and regulation responses when the controlled system is subjected to wide range of operating conditions (Krishnan 2001).

Controllers are often required to be sufficiently robust in the presence of indeterminate variations in the system parameters. A widely used control technique is the application of proportional-integral-derivative (PID) controllers, where control actions are

*Corresponding author's email: eafeilat@squ.edu.om

obtained by pre-tuning the controller's parameters in order to obtain the minimum output error of the controlled system. However, these types of PID controllers possess poor transient response and provide insufficient robustness with respect to variations of the parameters of the system under control (Kuo 2003; Anderson 1993). On the other hand, adaptive control strategies have been shown to offer advantages over classical feedback control methods as they can estimate the parameters of a controlled system and modify a controller online to meet the desired system performances (Sastry and Sidori 1989). Adaptive controllers, however, are complex, highly mathematical, and costly to implement. They also involve extensive computations and lengthy processing times. These constraints limit the applications of the adaptive control strategies.

With the emerging development in artificial intelligence applications, neural networks (NNs) have been used in identification and control of linear and nonlinear systems (Hagan *et al*. 2002; Narendra and Parthasathy 1990; Chen and Billings 1992; Rivals and Personnax 2000; Haykin 1994). NN is considered an attractive tool to establish the mathematical relationship of the dynamic system based on the input output data. A large number of architectures and algorithms for identication and control using NNs were proposed by Chen and Billings (1992). It has been shown that feedforward NNs with one hidden layer can uniformly approximate any continuous function within a prespecified accuracy (Haykin 1994). The trained NN can model the behavior of the system in the forward or inverse mode of operation. The main advantage of NN-based techniques over conventional techniques is the non-algorithmic parallel-distributed architecture for information processing that allows it to learn any complex input-output mapping (Haykin 1994).

Most of the research in the identification and control of DC motors is concentrated on using feed forward NNs with back propagation training, referred to as back propagation neural networks (BPNNs) (Weerasooriya and Elsharkawi 1991; Veeracharry and Yadaiah 2000; Rubaai and Kotaru 2000; Jun and Gi 2000; Hunt *et al*. 1992; Tipsuwanporn *et al*. 2002; Hissein *et al*. 2003, Berrospe *et al*. 2009; Peng and Dubay 2011; Sudarsan *et al*. 2011). In fact, there are several problems associated with these networks. First, BPNNs are prone to getting stuck in local minima on the error surface, giving a solution that is not optimal. Second, BPPNs have a relatively slow convergence rate, thus causing computation time for training such networks with a large number of parameters to be very long. Finally, it is difficult to determine a minimal but adequate architecture that minimizes training time and optimizes generalization as well.

Radial basis function NN (RBFNN) is another type of feed forward neural network that has universal approximation abilities. Unlike the BPNN, RBFNN has the best approximation property. It has been acknowledged that approximation accuracy properties of RBFNN are advantageous as compared to the other methods. Even more important for many applications, the RBFNNs provide linear approximation in the network weights. This feature makes powerful tools of the linear system theory applicable to the RBFNN identification of nonlinear systems. The "linear in parameters" of the radial basis functions guarantees the convergence of the parameters to the global minimum. Furthermore, RBFNNs are not as sensitive to the architecture as BPNNs (Al-Moudi and Zhang 2000; Shen *et al*. 2002; Park and Sandberg 1993; Sundarajan 1999; Poggio and Girosi 1990).

In this paper, the effectiveness and robustness of RBFNNs in identification and control of a SEDCM driving a centrifugal pump is investigated. Two RBFNN-based controller schemes are applied to construct a robust controller. One of these schemes is the direct inverse RBFNN cascaded controller that represents the inverse model of the system. The other scheme is the internal model RBFNN controller that uses both forward and inverse models. The performance of the proposed RBFNN control schemes will be compared to that of the BPNN schemes.

This paper is organized as follows. Section 2 presents the system under study (SEDCM) and illustrates the discrete mathematical modeling of the system. In section 3, the concept of a NN-based system identification and control is reviewed. Section 4 gives a brief review of the RBFNN for nonlinear system modeling and describes its architecture and training. RBFNN training and testing of the proposed identification and control structures are presented in section 5. Finally, drawn conclusions are stated in section 6.

2. Mathematical Model of the SEDCM

In a SEDCM, the speed can be controlled smoothly over a wide range by adjusting either the armature voltage or the field current. Speed control ranging from zero to nominal speed can be obtained by armature voltage control, while speed control above nominal value can be achieved by flux weakening at constant power output. This paper focuses on armature voltage speed control at constant flux (Krishnan 2003; Kuo 2003). The dynamics of the SEDCM, Figure 1, are described by the following electrical and mechanical differential equations:

$$L_A \frac{di_A}{dt} = -i_A R_A - K\omega + v_A \tag{1}$$

$$J \frac{d\omega}{dt} = K i_A R_A - B\omega - T_L \tag{2}$$

Where v_A is the motor input voltage; i_A is the armature current; ω is the rotor speed; T_L is the load torque; R_A is the armature resistance; L_A is the armature inductance; J is the motor rotational inertia; B is the damping constant; and K is the torque or EMF constant. The name-plate values and parameters of the motor are given in the appendix.

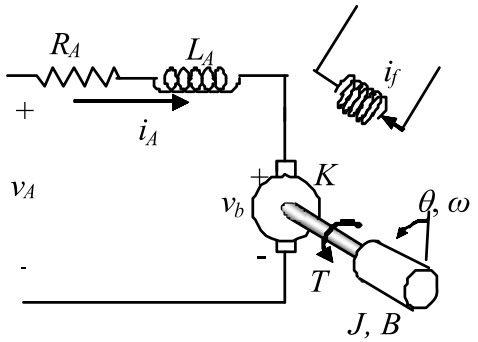

Figure 1. Electrical schematic of a SEDCM

Using first- and second-order finite-backward difference approximation for dx/dt and d^2x/dt^2 respectively, the finite difference equation that governs the discrete-time dynamics of the SEDCM is given by (Rubaai and Kotaru 2000):

$$\omega(k) = \alpha v_A(k) + \beta\omega(k-1) + \delta\omega(k-2) + \theta T_L \quad (3)$$

where α, β, δ, and θ are constants with values that depend on the sampling time interval ΔT and motor parameters as well as given by (Sastry 1989; Chen and Billings 1992):

$$\alpha = \frac{K\Delta T^2}{\Delta T^2\left(R_A B + K^2\right) + \Delta T\left(R_A J + L_A B\right) + L_A J} \quad (4)$$

$$\beta = \frac{\Delta T\left(R_A J + L_A B\right) + 2L_A J}{\Delta T^2\left(R_A B + K^2\right) + \Delta T\left(R_A J + L_A B\right) + L_A J} \quad (5)$$

$$\delta = \frac{-L_A J}{\Delta T^2\left(R_A B + K^2\right) + \Delta T\left(R_A J + L_A B\right) + L_A J} \quad (6)$$

$$\theta = \frac{-R_A \Delta T^2}{\Delta T^2\left(R_A B + K^2\right) + \Delta T\left(R_A J + L_A B\right) + L_A J} \quad (7)$$

3. Neural System Identification and Control

3.1 Neural System Identification

System identification is the process of developing a mathematical model of a dynamic system based on sampled input and output data from the actual system (Narendra and Parthasarathy 1990). An advantage of system identification is evident if the process is changed or modified. System identification allows the real system to be altered without having to calculate the dynamical equations and model the parameters again. For linear time-invariant systems with unknown parameters, the generation of identification models is well established (Narendra and Parthasarathy 1990). For a single-input single-output (SISO) system, the system can be represented as:

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + \sum_{j=0}^{m-1} \beta_j u(k-j) \quad (8)$$

where α_i and β_j are constant unknown parameters. This implies that the output at time $k + 1$ is a linear combination of the past values of both the input and the output. Equation (8) motivates the choice of the following identification models:

Parallel model:

$$\hat{y}(k+1) = \sum_{i=0}^{n-1} \alpha_i \hat{y}(k-i) + \sum_{j=0}^{m-1} \beta_j u(k-j) \quad (9)$$

Series-parallel model:

$$\hat{y}(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + \sum_{j=0}^{m-1} \beta_j u(k-j) \quad (10)$$

where α_i (i = 0, 1, ... , n-1) and β_j (j = 0, 1,. . . , m-1) are adjustable parameters. In the parallel identification model the output at time $k + 1$ is $\hat{y}(k+1)$ and is a linear combination of the past values of the model $\hat{y}(k-i)$ as well as those of the input. On the other hand, the output of the series-parallel identification model $\hat{y}(k+1)$ is a linear combination of the past values of the input and output of the system $y(k-i)$. Different learning algorithms have been suggested for the adjustment of the parameters of the identification models.

Motivated by the models which have been used in the adaptive systems literature for the identification and control of SISO linear systems, four identification models were proposed in (Narendra and Parthasarathy 1990) for the identification and control of nonlinear systems. For the reader convenience, the four models that describe a discrete-time system by nonlinear difference equations are introduced:

Model **I**:

$$y(k+1) = \sum_{i=0}^{n-1} \alpha_i y(k-i) + g\begin{bmatrix} u(k), \\ u(k-1),\ldots,u(k-m+1) \end{bmatrix} \quad (11)$$

Model **II**:

$$y(k+1) = f\left[y(k), y(k-1),\ldots, y(k-n+1)\right] + \sum_{i=0}^{-1} \beta_i u(k-i) \quad (12)$$

Model **III**

$$y(k+1) = f\left[y(k), y(k-1),..., y(k-n+1)\right] + g\left[u(k), u(k-1),..., u(k-m+1)\right] \quad (13)$$

Model **IV**:

$$y(k+1) = f\begin{bmatrix} y(k), y(k-1),..., y(k-n+1); \\ u(k), u(k-1),..., u(k-m+1) \end{bmatrix} \quad (14)$$

where [*u(k)*, *y(k)*] represents the input-output pair of the SISO system at time *k*, and $m \le n$. In all four models, the output of the plant at the time *k+1* depends both on its past n values *y*(*k-i*) (*i* = 0, 1, . . . , *n*-1) as well as on the past *m* values of the input *u(k-j)*, (*j* = 0, 1,…, *m*-1). The dependence on the past values *y*(*k-i*) is linear in model I, while in Model II the dependence on the past values of the input *u*(*k-j*) is assumed to be linear. In Model III, the nonlinear dependence of *y*(*k*+1) on *y*(*k-i*) and *u*(*k-j*) is assumed to be separable. It is evident that Model IV in which *y*(*k*+1) is a nonlinear function of *y*(*k-i*) and *u*(*k-j*) subsumes Models I-III.

Former studies in system identification have demonstrated that NNs are successful in modeling both linear and nonlinear systems (Narendra and Parthasarathy 1990; Chen and Billings 1992; Rivals and Personnaz 2000; Haykin 1994; Weerasooriya and Elsharkawi 1991; Veerachary and Yadaiah 2002; Rubaai and Kotaru 2000). A NN-based system identification involves the following design considerations: selecting process excitation signal, choosing data sample time, selecting NN architecture, and testing the resulting model. The most common method of NN identification is called forward modeling using either the series-parallel identification model as shown in Figure 2, or the parallel identification models of Figure 3.

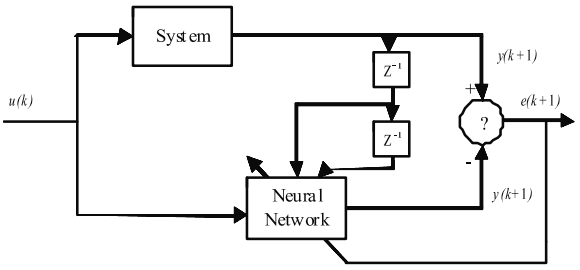

Figure 2. Series-parallel identification model

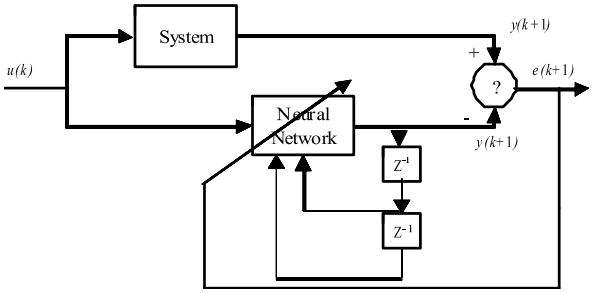

Figure 3. Parallel identification model

The attempts to model the mapping of system input to output. The purpose is to find a NN with response $\hat{y}$ that matches the response *y* of the system for a given set of input *u* received by both the system and the NN. The error *e* is minimized during network weight adjustment. This is an example of supervised learning, where the teacher (system) provides target values for the learner (NN).

The performance of the NN model is tested by calculating the MSE. The MSE gives a good indication of the accuracy of the NN model. The MSE between the NN model and the system should be low. The output from the NN model and system is plotted to compare the dynamics.

3.2 Neural System Control

The basic objective of system control is to provide the appropriate input signal to a given system to yield its desired response. If the actuating signal at the input to the system is generated by a NN-based controller, then it is termed neural control, or neurocontrol (NC). NN uses multilayer feed forward architecture and implements the mapping of the plant inverse.

NC has been defined as using a well-specified NN to emit actual control signals. Thus, a controller is designed to control a system according to the desired set-points which will involve some intelligence. NC has gained widespread acceptance because it can be trained to learn any function eliminating the need for complex mathematical analyses. Using activation functions allows nonlinear mapping ability for solving nonlinear control problems. Moreover, the massive parallelism of NNs offers very fast multiprocessing technique that could be implemented in neural chips. Neurocontrollers are also able to perform well under a wider range of uncertainty. In this paper, neurocontrol structures that have direct reliance on system forward and inverse models are adopted. Accordingly, two control structures have been applied, namely, direct inverse control (DIC) and internal model control (IMC).

3.3 Direct Inverse Control Scheme

DIC modeling is utilized to generate the inverse of the system. Contrary to the supervised control, inverse control does not require an existing controller in training. The principle of the DIC is that if the system is described by:

$$y(k+1) = g\begin{pmatrix} y(k),..., y(k-n+1), \\ u(k),...u(k-m+1) \end{pmatrix} \quad (15)$$

A NN is trained to perform as the inverse of the process:

$$\hat{u}(k) = \hat{g}^{-1}\begin{pmatrix} y(k+1), y(k),..., y(k-n+1), \\ u(k-1),...u(k-m) \end{pmatrix} \quad (16)$$

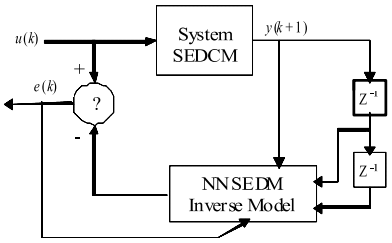

Figure 4. Inverse modeling of a system

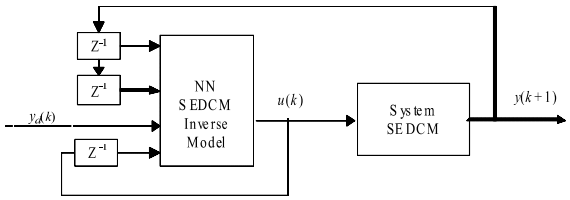

Figure 5. Direct inverse control (DIC) scheme

Figures 4 and 5 show an example of NN-DIC. A NN is trained to identify the inverse of the system (Narendra and Parthasarathy 1990). The SEDCM) output is used as an input to the NN inverse identifier. The NN output is compared with the training signal (the system input), and the error signal is used to train the NN. This training method will force the NN inverse identifier to represent the inverse of the system. In a DIC scheme, the NN inverse identifier plays the role of the controller when cascaded with the system. In this scheme, the input of the controller is y_d, which is the desired output of the system, and the actual output of the combined system is equal to the desired output y_d if the NN inverse model accurately represents the inverse of the SEDCM as shown in Figure 5. The inverse nonlinearities in the controller cancel out the nonlinearities in the system. The advantages of the NN controller is if an uncertainty in the system occurs, the NN will be able to adapt its parameters and maintain control of the plant when other robust controllers would fail.

3.4 Internal Model Control Scheme

IMC is based on DIC modeling. A NN forward identifier model is placed in parallel with the real system. The controller is an inverse model of the system as shown in Figure 6. With the IMC scheme, the aim is to eliminate the unknown disturbance affecting the system. The difference between the system and the NN model *d(k)*, is determined. If the NN forward model is a good approximation of the system, then the error *d(k)* is equal to the unknown disturbance. The disturbance $d(k)$ is the information that is missing from the NN model and can be used to improve the control. The disturbance *d(k)* is subtracted from the input reference y_d, resulting in perfect control.

3.5 Excitation Signal

A system excitation signal is used to generate input-

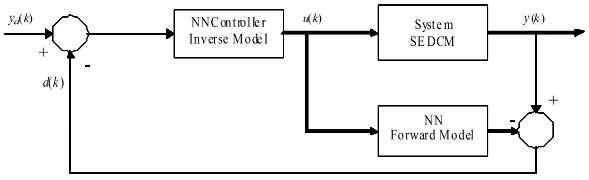

Figure 6. Internal model control (IMC) scheme

output system data which contain sufficient information for a NN to identify the system dynamics over the entire operating range. A NN should be trained with dynamically rich data which covers a wide range of the system operating region. The trained NN is provided with the sequence of recent output of the system in addition to present system output. A random amplitude signal is commonly used as the system excitation signal to generate an open loop data for network training. This signal consists of a uniformly distributed random variable applied to the system input.

4. Radial Basis Function Neural Network

Over the past decades, NNs have received great attention among scientists for solving some complex engineering problems for which conventional approaches have proven ineffective (Al-Moudi and Zhang 2000; Shen *et al*. 2002; Park and Sandberg 1993; Sundarajan 1999). In particular, RBFNNs have universal approximation ability. Comparing with BPNNs, the RBFNNs have some better approximation properties, such as high accuracy of approximation, especially, the connection weights from the hidden layer to the output layer are linear (which implies that linear optimal algorithms can be used in RBFNNs and guarantee the global convergence of the parameters). Moreover, while training RBFNNs, only one part of the nodes is affected by a given input, and only a portion of the parameters may need to be adjusted, thus reducing the training time and computational burden.

A RBFNN has an input layer, nonlinear hidden layer with RBF units, and a linear output layer. The nodes within each layer are fully connected to the previous layer nodes. The input variables are each assigned to nodes in the input layer and connected directly to the hidden layer without weights. The nodes calculate the Euclidean distances between the centers and the network input vector, and pass the results through the

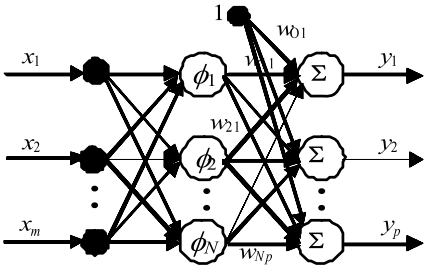

Figure 7. Radial basis function neural network architecture

RBF units. The output layer nodes are weighted linear combinations of the hidden layer. The structure of a RBFNN with m inputs, p outputs and N hidden nodes is depicted in Figure 7.

For a RBFNN with m input nodes, p output nodes and N hidden nodes, the hidden unit can be expressed as a matrix $\Phi = [\phi_1 \quad \phi_1 \quad \cdots \quad \phi_N]$ and NN weight vector $W = \{w_{ij}, i = 1, 2, \cdots N, j = 1, 2, \cdots p\}$, where the input vector is $X(k) = [x_1(k) \quad \cdots \quad x_m(k)]^T$, the output vector is $\hat{Y}(k) = [\hat{y}_1(k) \quad \cdots \quad \hat{y}_p(k)]^T$, and $\phi_j(X(k))$ is a nonlinear Gaussian activation function:

$$\phi_j(X(k)) = exp(||X(k) - C_j||^2 / \sigma_j^2) \tag{17}$$

where C_j (j = 1,2,...,N) is the vector of centers of the jth hidden unit with the same dimension as the input vector $X(k)$, σ_j the width of the j^{th} RBF hidden unit, and ||o|| is the Euclidean norm. The i^{th} RBF network output can be represented as a linearly weighted sum of N basis functions:

$$\hat{y}_i(k) = w_{oi} + \sum_{j=1}^{N} w_{ji}\phi_j(X(k)), \quad i = 1, 2, \cdots, p \tag{18}$$

where w_{ji} and w_{oi} are the weights, w_{oi} is used to compensate for the difference between the average value over the data set of the RBF activation and the corresponding average value of the target outputs (Al-Moudi and Zhang 2000). With the structure described above, the transformation from the input layer to the hidden layer is nonlinear, due to the use of Gaussian functions $\phi(\bullet)$ for RBF, and the connection of the hidden layer to the output layer is linear.

Before the NN can be employed to yield the desired outputs, the weights must be determined. The process of determining the weights is called the training (or learning) process. In the training process, a set of input patterns is presented at the input along with the desirable output patterns. Weights in the network are then adjusted such that an error measure (the difference between the desired output y_j and the actual outputs of the network $\hat{y}_j$) is minimized. In this paper, the criterion of training a RBFNN is to minimize the MSEs:

$$MSE = \frac{1}{N_p} \sum_j \sum_k \left(y_j(k) - \hat{y}_j(k)\right)^2 \tag{19}$$

where N_p is the number of training patterns, and $y_j(k)$ and $\hat{y}_j(k)$ denote the desired and model output at the kth sample points. The weights w_{ji} can be determined by using the least-squares (LS) method after selecting suitable values for the spread and centers of the radial basis function. Random selection of fixed centers is adopted in this paper. The training of the RBFNN is terminated when an MSE goal value is achieved.

5. Simulation Results

In this paper, an artificial NN approach has been applied to the identification and control of the SEDCM. A discrete mathematical model representing the SEDCM as described by Anderson (1993) was developed in Matrix Laboratory (MATLAB). The mechanical load is assumed to be a centrifugal pump with a load characteristic given as:

$$T_L = 0.001\omega^2(k\text{-}1) \tag{20}$$

5.1 Generation of the NN Input-Output Training Patterns

The MATLAB was used to perform the training and simulation of the system under study. Input-output training sampled data of 6000 patterns, containing sufficient information for a BPNN or RBFNN to identify the SEDCM dynamics over the entire operating range were generated using uniformly distributed random amplitude/ frequency voltage signals as shown in Figure 8. Each input-output pattern consisted of 3 input samples and one output sample. The input-output training data samples were obtained at sampling time ΔT = 1 minutes. Figure 9 shows the speed response of the discrete model of the SEDCM to a ranom input voltage.

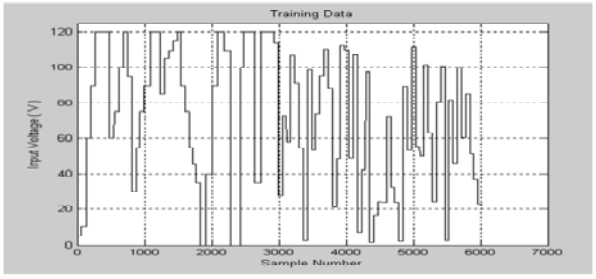

Figure 8. Random amplitude/frequency input voltage signal

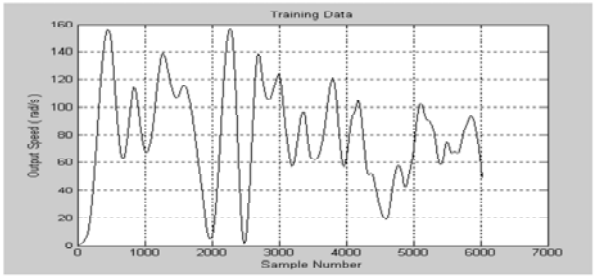

Figure 9. Motor speed response for random input voltage signal

5.2 Training the BPNN and RBFNN Identification Models

For identification purpose, the series-parallel model,

as shown in Figure 2, was used to model the SEDCM which is governed by the difference Eqn (3). The trained NN was provided with a random sequence of input voltage in addition to the past output values of the motor speed. The input vector of the NN identifier can be rewritten as:

$X(k) = \left[v_A(k) \quad \omega(k-1) \quad \omega(k-2) \right]$, and the only output of the NN identifier is the identified motor speed $\hat{\omega}(k)$.

For the BPNN, the batch mode training with the fast Levenberg-Marquardt back propagation (LMBP) algorithm was selected for weights adjustment (weights and biases were only updated after all of the inputs and targets were presented). It was found that the number of hidden neurons and their sensitivity to initial weights are the two main factors that affecting the performance of the LMBP algorithm and might lead to convergence to local minima. Different BPNN architectures were investigated to obtain the best performance as shown in Table 1. It was found that an architecture of 3-5-1 gives excellent performance in the shortest training time with a MSE of = 2.2 10^{-8}.

Table 1. Performance of the BPNN system identifier

# Hidden Neuron	# Epochs	MSE
5	10000	2.2×10^{-8}
7	10000	7.3×10^{-9}
10	10000	1.6×10^{-9}

Likewise, the training of a RBFNN identifier is accomplished in two stages. At first, the centers and the measure of spread (σ) were selected randomly. Then a direct inversion matrix training algorithm was applied to obtain the weights vector. Different network structures were examined. The number of hidden units was determined through trial-and-error. Based on satisfying the minimum MSE and the shortest training time, the optimal number of hidden neurons was found to be 4. Table 2 shows the performance of the RBFNN model in terms of the MSE as affected by the number of hidden neurons and the value of σ.

Table 2. Performance of the RBFNN system identifier

# Hidden Neuron	σ	MSE
4	0.03	1.69×10^{-9}
7	0.24	1.59×10^{-6}
10	0.42	1.10×10^{-5}

The speed responses for a 3-5-1 BPNN identification model with an MSE = 2.2 x 10^{-8} and a 3-4-1 RBFNN identification model with an MSE = 1.69 x 10^{-9} are depicted in Figure 10. It can be seen in Figure 10 which illustrates an exact fit between the SEDCM and each of the BPNN and RBFNN identifiers.

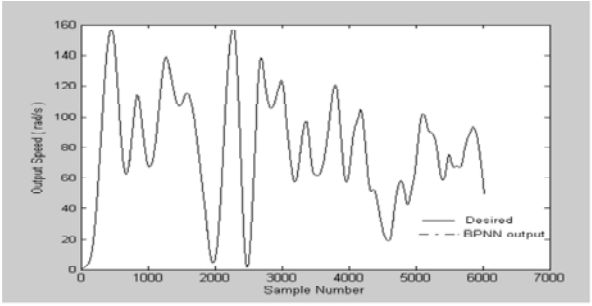

Figure 10-a. SEDCM identification using 3-5-1 BPNN model

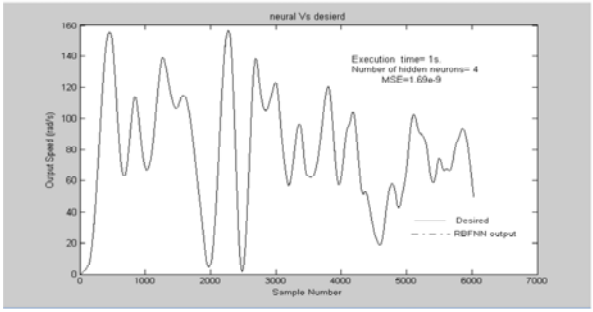

Figure 10-b. SEDCM identification using 3-4-1 RBFNN model

5.3 Validating the BPNN and RBFNN Identification Models

The adequacy and generalization capabilities of the trained NN identification models were subjected to model validity test by using a validation signal of a different magnitude range and/or clock period of the random amplitude excitation signal. The SEDCM model along with each of the BPNN and RBFNN identification models were subjected separately to a random input voltage excitation signal with 1800 samples and the difference between the outputs of two models were used to calculate the MSE. The generalization performance of the BPNN and RBFNN identifiers with an MSE = 2.4 x 10^{-4} and an MSE = 3.9 x 10^{-5} are demonstrated in Figure 11-a and Figure 11-b, respectively.

5.4 BPNN and RBFNN Inverse Models of the SEDCM

Figure 4 displays the NN inverse model structure of the SEDCM, where the input voltage is the desired output of the NN inverse model. The NN inverse model is trained using 6000 input-output training patterns. Each pattern consists of 3 input samples representing the speed, and one output representing the desired input voltage. Several BPNN and RBFNN architectures were investigated. A BPNN inverse model with a 3-5-1 architecture and a RBFNN inverse model with 3-5-1 architecture were found very satisfactory. The performance simulations of both of the

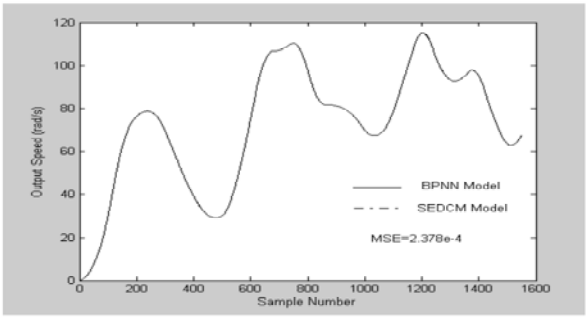

Figure 11-a. Validity of 3-5-1 BPNN model, MSE = 2.4 x 10^{-4}.

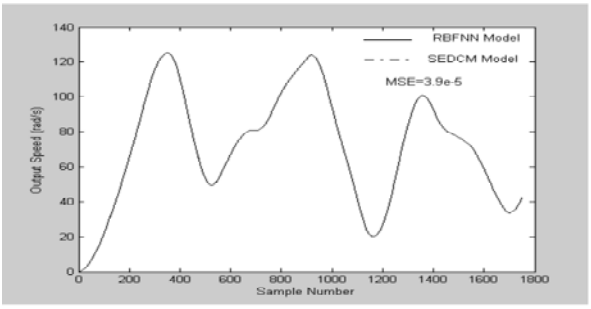

Figure 11-b. Validity of 3-4-1 RBFNN model, MSE = 3.9 x 10^{-5}.

Table 3. Performance measures of the BPNN and RBFNN DIC schemes

DIC NN Model	Maximum overshoot	Rise time	Settling time	Steady State Error
BPNN	146.9 (22.42%)	0.156 s	0.705 s	0.0
RBFNN	146.6 (22.17%)	0.155 s	0.702 s	0.0

BPNN and RBFNN inverse identification models for a random input voltage are illustrated in Figures 12-a and b, respectively. The responses of both NN inverse models accurately fit the desired random input voltage of the SECM. The performance simulations of the BPNN and the RBFNN as neural controllers in both DIC and IMC schemes are presented in section 5.

5.5 Performance of Direct Inverse Control (DIC) Scheme

The dynamic performance for the proposed BPNN- and RBFNN-DIC schemes are evaluated in terms of the step response measures as depicted in Table 3 and Figure 12. It can be seen that the both BPNN and RBFNN inverse models schemes were able to track the step response of the motor with zero steady-state error.

The dynamic performance of the BPNN-DIC and RBFNN-DIC schemes following changes in the desired speed, change of motor load and variation of motor parameter (inductance) is investigated as shown

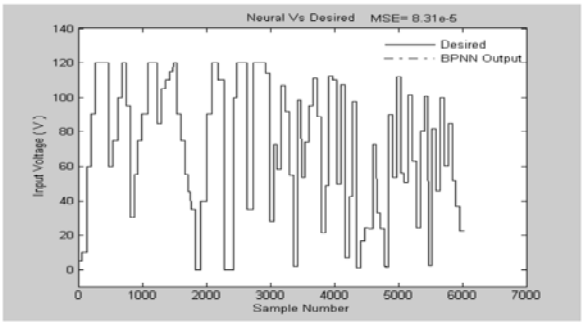

Figure 12-a. Performance of 3-5-1 BPNN inverse model

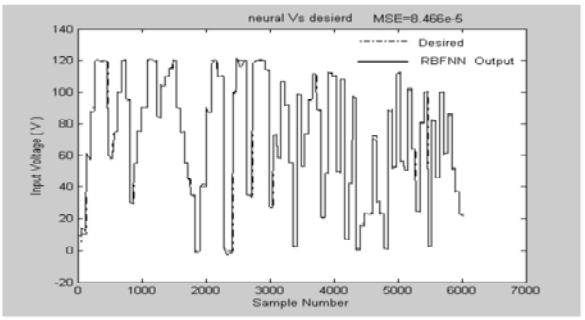

Figure 12-b. Performance of 3-5-1 RBFNN inverse model

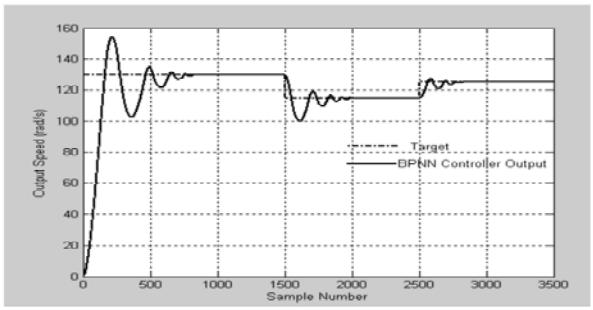

Figure 13-a. Step response of the BPNN-DIC scheme

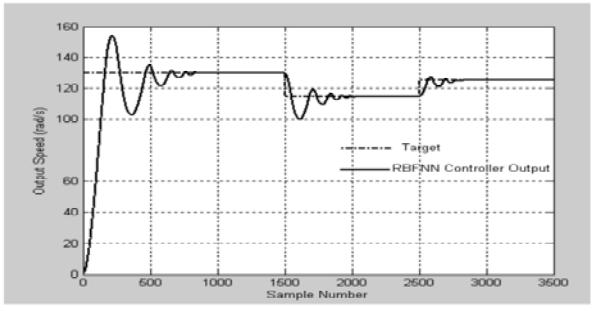

Figure 13-b. Step response of RBFNN-DIC scheme

in Figures 14-16, respectively. Examining Figures 14-16, it can be seen that the developed BPNN-DIC and RBFNN-DIC schemes exhibit identical excellent step responses and adapt well to sharp instantaneous changes in speed trajectory, sudden changes in connected loads, and variation of motor parameters.

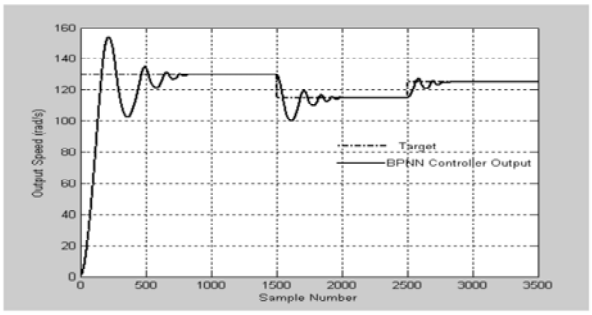

Figure 14-a. BPNN-DIC scheme response following a speed change

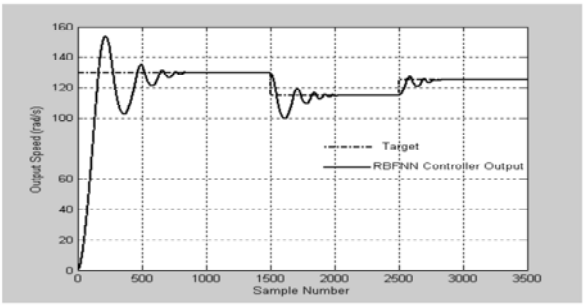

Figure 14-b. RBFNN-DIC scheme response following a speed change

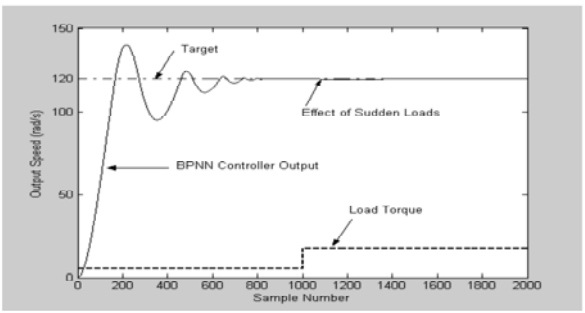

Figure 15-a. BPNN-DIC scheme response following a sudden load change

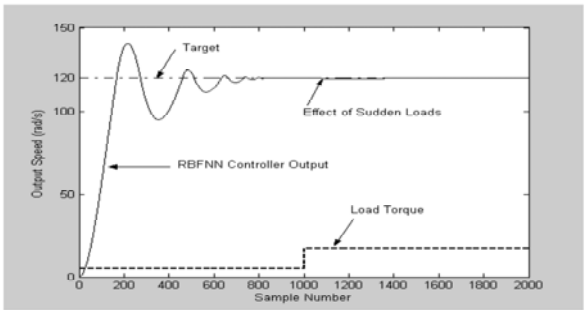

Figure 15-b. RBFNN-DIC scheme response following a sudden load change

5.6 Performance of IMC Scheme

The IMC scheme for the SEDCM was implemented in this paper by connecting the forward NN and inverse NN models as shown in Figure 6. In this study, the RBFNN models were implemented as it was found that the performance of the RBFNN-IMC scheme is exactly identical to that of the BPNN-IMC

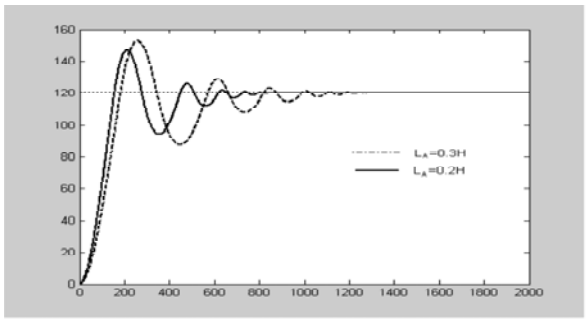

Figure 16-a. BPNN-DIC scheme response following a parameter change

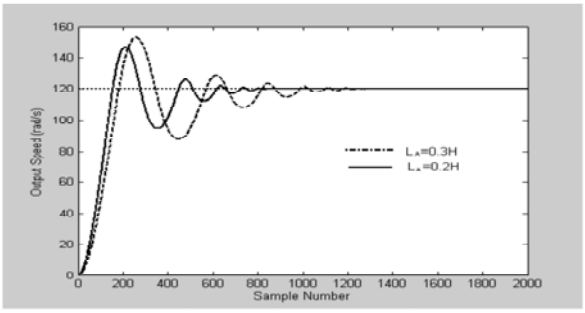

Figure 16-b. RBFNN-DIC scheme response following a parameter change

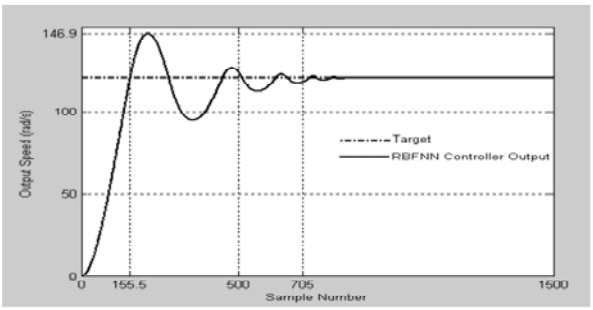

Figure 17. Step response of the RBFNN-IMC scheme

scheme. The forward RBFNN model of the SEDCM is placed in parallel with the discrete model of the SEDCM. The difference between the outputs of the discrete and forward models is used to subtract the effect of the control signal from the system output. The step response of the RBFNN-IMC scheme is shown in Figure 17. As can be seen, the dynamic behavior of the RBFNN-IMC scheme is identical to that of the RBFNN-DIC scheme as illustrated in Figure 13. This can be attributed to the high accuracy of the NN inverse model of the SEDCM.

6. Conclusions

In this paper, a RBFNN approach for the identification and control of SEDCM with nonlinear load characteristics is presented. The performance of RBFNN for off-line identification and control of SEDCM had been compared with the well-known BPNN. For accurate

forward and inverse identifications, choices of the proper system excitation signal, data sampling time, and neural network model structure are investigated by time-domain simulations. Cross-validation test is also applied on the NN identification models. Two neuro-contrllers using direct inverse and internal model control schemes are used. The simulation results have shown that both the RBFNN controller and the BPNN controller exhibit excellent performance. However, the RBFNN model is simpler to implement and faster to train than the BPNN model. The simulation results have also shown that RBFNN, with small number of hidden neurons, can be effective in designing robust neurocontrollers of SEDCM with excellent dynamic behaviors. The robust neurocontrollers are not affected by the changes in load torque, motor parameters or instantaneous changes in speed trajectory. Both IMC and DIC exhibited identical dynamic behavior.

References

Al-Amoudi A, Zhang L (2000), Application of radial basis function network for solar-array modeling and maximum power-point prediction. IEE Proc. Gener. Transm. Distrib, 147(5):310-316.

Anderson ZYO (1993), Controller design: moving from theory to practice. IEEE Contr. Sys. Mag, 16-25.

Berrospe E, Gonzalez-Olvera MA, Tang Y (2009), Identification and speed control of a DC motor using an input-output recurrent neurofuzzy network. Advances in Computational Intelligence 116:239-248.

Chen S, Billings SA (1992), Neural networks for nonlinear system modeling and identification. International Journal Control 56:319-346.

Hagan MT, Demuth HB, Jesus OD (2002), An introduction to the use of neural networks in control systems. International Journal Robust Nonlinear Control 12:959-985.

Haykin S (1994), Neural networks a comprehensive foundation. Macmillan College Publishing Company, NY.

Hissein A, Hirasawa K, Hu J (2003), Online identification and control of a PV-supplied DC motor using universal learning networks. European Symposium on Artificial Neural Networks, Belgium173-178.

Hunt KJ, Sbarbaro D, Zbikowski R, Gawthrop PJ (1992), Neural networks for control systems: A survey. Automatica 28(6):1083-1112.

Jun OJ, Gi JJ (2000), A parallel neuro-controller for DC motors containing nonlinear friction. Neurocomputing 30:233-248.

Kirshnan R (2001), Electric motor drives modeling, analysis, and control. Prentice Hall, Upper Saddle River, NJ.

Kuo BC (2003), Automatic control systems. Wiley.

Narendra KS, Parthasarathy K (1990), Identification and control for dynamic systems using neural networks. IEEE Trans. on Neural Networks 1:4-27.

Park J, Sandberg IW (1993), Approximation and radial-basis function networks. Neural Computation 5:305-316.

Peng J, Dubay R (2011), Identification and adaptive neural network control of a DC motor system with dead-zone characteristics. International Society of Automation (ISA) 50(4):588-598.

Poggio T, Girosi F (1990), Networks for approximation and learning. Proceedings of IEEE 78:1481-1497.

Rivals I, Personnaz L (2000), Nonlinear internal model control using Neural Networks: Application to Processes with Delay And Design Issues," IEEE Transations Neural Networks 11(1):80-90.

Rubaai A, Kotaru R (2000), Online identification and control of a DC motor using learning adaptation of neural networks. IEEE Transations Industry Application 36:935-942.

Sastry SS, Sidori IA (1989), Adaptive control of linearizable systems. IEEE Transations Automatic Control 34(11):1123-1131.

Shen C, Cao GY, Zhu XJ, (2002), Nonlinear modeling of MCFC stack based on RBF neural networks identification. Simulation Modeling Practice and Theory 10:109-119.

Sudarsan MV, Murthy GRK, Nagakumar GRS (2011), Designing of ANN based speed controller for phase controlled DC motor. International Journal of Engineering and Science (IJEST), 3(7):5837-5844.

Sundararajan N (1999), Radial basis function neural networks with sequential learning: MRAN and Its Applications. World Scientific, Singapore.

Tipsuwanporn V, Piyarat W, Tarasantisuk C (2002), "Identification and control of brushless DC motors using on-line trained artificial neural networks. Power Conversion Conference 3:1290-1294.

Veeracharry M, Yadaiah N (2000), ANN based peak power tracking for PV supplied DC motors. Solar Energy 69(4):343-350.

Weerasooriya S, Elsharkawi M (1991), Identification and control of a DC motor using back propagation Neural Networks. IEEE Transations Energy Conversion 6:663-669.