International Journal of Electronics and Communication Engineering.
ISSN 0974-2166 Volume 5, Number 5 (2012), pp. 545-566

http://www.irphouse.com

Dynamic Nonlinear System Identification Using Discrete Time Quadratic Neural Units

AshrafE.Ghania[1]andFrans David[2]

[1]Computer science, Faculty of Science, University of Namibia(UNAM),Windhoek,Private Bag13301,Namibia.
e-mail: ashran2002eg@yahoo.com
[2]School of Information Science and Engineering, Central South University, Changsha, Hunan Province, P. R. China, 410083,
E-mail:davidfrans@gmail.com

Abstract

This paper presents a novel Discrete Time Quadratic Neural Units (DT-QNU) for unknown dynamic nonlinear system identification. The proposed Discrete Time Quadratic Neural Units resembles the conventional Static NeuralNetworks. The novelties of our approach include: firstly; the realization of a Modified Mahalanobis Distance (MMD) to reduce the number of adaptable weights without sacrificing the neural performance; secondly; the output depends not only on the current input to the Discrete Time Quadratic Neural Unit , but also on the current or previous inputs, outputs, or states of the Neural Unit and always will converge to one of their asymptotically stable equilibrium points regardless of the initial values of the inputs.Thirdly; the proposed Discrete Time Quadratic Neural Units is capable of accuratelyidentifying of nonlinear dynamic systems using fewer parameters. Computer simulationsand results have successfully confirmed the effectiveness and superiority of the proposed Discrete Time Quadratic Neural Units.

Keywords: Mahalanobis Distance,Dynamic Neural Unit, ContinuousState-Space Model Realization, Non-Linear Dynamic System Identification

Introduction

Artificial Intelligence (AI) systems are widely accepted as a technology offering analternative way to tackle complex and ill-defined problems [1]. They can learn fromexamples, are fault tolerant in the sense that they are able to handle noisy and

incompletedata, are able to deal with non-linear problems, and once trained can perform prediction andgeneralization at high speed [2]. They have been used in diverse applications in control,robotics, pattern recognition, forecasting, medicine, power systems, and manufacturing,and optimization, signal processing and social/psychological sciences. AI systems comprise areaslike expert systems, Artificial Neural Networks (ANNs), Genetic Algorithms, Fuzzy Logic and various hybrid systems, which combine two or more techniques [3].

Artifcial neural networks (ANNs) have been used inrecent years to avoid the problems associated with deterministicapproaches, and have been shown to approximatenonlinear functions up to any desired level ofaccuracy [4]. They are also less sensitive to noise andincomplete information than other approaches such asempirical models and correlations. In recent years, thetechnique has been applied to many control problems[5], among them the prediction of the steady state [6] andthe dynamic behavior of heat exchangers [7-9]. Theadvantage of using ANNs to simulate thermal processesis that, after they are trained, they represent a quick andreliable way of predicting their performance. They canalso be continuously updated. Thus, if we apply thistechnique to the problem of simulation and identification ofnon-linear systems, then we obtain an accurate prediction with a shortcomputational time for the simulation which can be usedin an efficient real-time control scheme.

Also, thegreatest advantage of a neural network is its ability to model complex nonlinear relationshipwithout a priori assumptions of the nature of the relationship like a black box [10]. The capability of neural networks for approximating arbitrary input-output mappings give a simple way to identify unknown dynamic functions in order to predict the needed output one step ahead or more. In a tracking system, measured radar signals mostly have been mixed with additive white noise. In order to filter out or minimize this measured noise on-line and to predict the aircraft position one step ahead, a simple back propagation algorithm has been used.On the order hand, neural networks can be classified into dynamic and static categories.Static (feed-forward) networks have no feedback elements and contain no delays; the outputis calculated directly from the input through feed-forward connections. In dynamic networks,the output depends not only on the current input to the network, but also on the current orprevious inputs, outputs, or states of the network. Dynamic networks are generally morepowerful than static networks (although somewhat more difficult to train). Because dynamicnetworks have memory, they can be trained to learn sequential or time-varying patterns [11].Neural networks can be classified into dynamic and static categories. Static networks have no feedback elements and contain no delays; the output iscalculated directly from the input through feed-forward connections. In dynamic networks,the output depends not only on the current input to the network, but also on the current orprevious inputs, outputs, or states of the network. Dynamic networks are generally morepowerful than static networks (although somewhat more difficult to train). Because dynamicnetworks have memory, they can be trained to learn sequential or time-varying patterns [12].

Dynamic predictions are,of course, harder and it was not until recently that dynamicalmodels started to appear in the literature [13-15].Most of them, in order to make the problem moretractable, rely on assumptions and simplifications thatare not

totally realistic [16-18].

Dynamic system identification is the model estimation process of capturing systemdynamics using measured input-output data, which is a very important prerequisite foranalysis and controller design in most control applications [19]. A good model representationoffers a good capability of representing different systems in terms of modelling accuracy and structure compactness. Therefore, to obtain a best fit for data with few parametershas become a top priority in the selection of model representation. Much of theliterature has widely described model representations for nonlinear system identificationproblems [20].

Among the diverse model representations, the block-oriented (BO) modelsthat are composed of dynamic linear blocks and static nonlinear blocks possess theflexibility of selecting blocks to represent the features of a given unknown system. Thechoices of different linear and nonlinear blocks result in various structures. One of thenotable nonlinear models is the Wiener model, consisting of a dynamic linear part cascadedwith a static nonlinear component [21]. Wiener models have been widely used inindustry such as in polymerization reactor control, fluid flow control, pH neutralizationcontrol, and identification of nonlinear biological systems [22]. The advantages of Wienermodels include: (1) the complexity of system dynamics is contained in the linearsubsystem whereas the complexity of nonlinearity only in the static subsystem and (2)the overall output of Wiener models can be written analytically as a kernel function expansion [22]. Another effective model representation is neural networks that have beentreated as a powerful model for nonlinear system identification problems. To name a few,Kalinli and Sagiroglu [23] presented a new recurrent neural network named ENEM (Elmannetwork with embedded memory) composed of Elman network and NARX neuralnetwork for dynamic nonlinear system identification. Lin [24] proposed a wavelet neuralnetwork with an online partition method and the gradient descent method to identify thenonlinear dynamic system. Lin and Xu [25] designed neuro-fuzzy systems with a modifiedvariable-length genetic algorithm to solve identification and control problems.

Wang and Chen [18] presented a Hammerstein-type recurrent neural networkwith a self-construction algorithm to identify nonlinear dynamic systems. Recently, manyresearchers have integrated neural networks with some linear systems to form Wienermodels. To name a few, AI-Duwaish*et al*. [15] used a linear autoregressive moving average(ARMA) model to represent the dynamic linear block and a multilayer feed-forwardneural network to model the static nonlinear element.

With a priori knowledge of the given nonlinear system, conventional system identificationapproaches that use either frequency domain or time domain methods canexplicitly approximate and simplify the nonlinar system dynamics in terms of a linearmodel. Among these approaches, fast Fourier transforms, maximum likelihood estimation,and least squares are three representative methods. In the early 1960s, the problemof realization of state-space representations using input-output descriptions has receivedconsiderable attention, which resulted in a wide variety of algorithms to solve theproblem.

Discrete-Time Dynamic Neural Unit (DTQNU)

Consider a general class of discrete-time dynamic neural networks (DTDNNs) with continuous states as shown in Figure (1) described by the following set of difference equations

$$x_i(k+1) = -\alpha_i x_i(k) + \sum_{i=1}^{n} w_{ij}\sigma_j(\mu_j x_j(k)) + s_i, \quad i = 1, 2, \ldots, n \tag{1}$$

or equivalently in a vector form, the discrete-time dynamic neural network is described as

$$\boldsymbol{x}(k+1) = -\boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{W}\boldsymbol{\sigma}(\boldsymbol{\Psi}\boldsymbol{x}(k)) + \boldsymbol{s} \tag{2}$$

where $x = [x_1, x_2, \ldots x_n]^T$ is the neural state vector, $W = [w_{ij}]_{nxn}$ is the synaptic weight matrix, $s = [s_1, s_{2i}, \ldots s_n]^T$ the constant threshold vector, $A = diag\alpha_1, \alpha_2, \ldots\ldots, \alpha_n$ with $|\alpha_i| < 1$ is the self-feedback coefficient matrix, $\omega = diag[u_1, u_2, \ldots\ldots, u_{n]}$ is the matrix of activation gains for controlling the state decay, and $\sigma\omega x = [\sigma(u_1x_1), \sigma(u_2x_2), \ldots\ldots, (u_nx_1)]T$ is the vector-valued activation function with the gain matrix ω. The first term in (2) is called the *self-feedback linear term* of the network, [26].

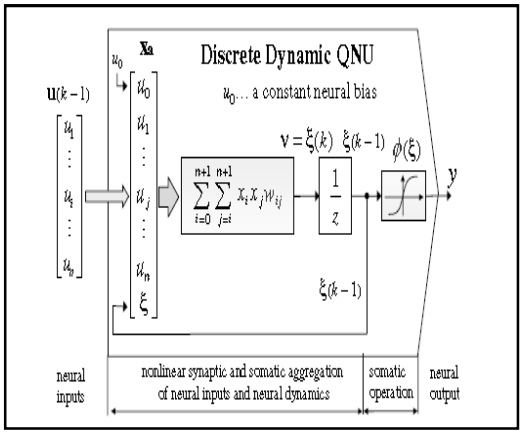


Figure 1: Discrete-time dynamic quadratic neural unit

As in continuous-time DNNs, the nonlinear neural activation function cr(.) may be chosen as a continuous and differentiable nonlinear sigmoid function satisfying the following conditions:

(i) $\alpha(x)$ —> ±1 as x —> ±∞;

(ii) $\alpha(x)$ is bounded with the upper bound 1 and the lower bound - 1;

(iii) $\alpha(x)=0$ at a unique point $x = 0$;

(iv) $\acute{\alpha}(x)>0$ and $\&'\{x\}$ —>0 as x —> ±00;

(v) $\acute{\alpha}(x)$ has a global maximal value of 1.

Now, briefly discusses the Mahalanobis distance and a modified Mahalanobis

distance that is formulated to support the concept of the neural unit with QNU.

The objective of QNU is to reduce the number of adaptable weights without sacrificing the neural performance. The number of parameters (weights) in the covariance matrix increases with the increase in dimensions of the input space. In order to reduce the number of parameters, there is a need to modify the M-distance equation without changing the concept of Mahalanobis. The structure of MM-distance is similar to the distance formula proposed by Mahalanobis except the numbers of elements in the formula are reduced significantly. Consider the elements of the covariance matrix Ω given by (3)

$$\Omega = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1D} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2D} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{D1} & \sigma_{D2} & \cdots & \sigma_{DD} \end{bmatrix} \tag{3}$$

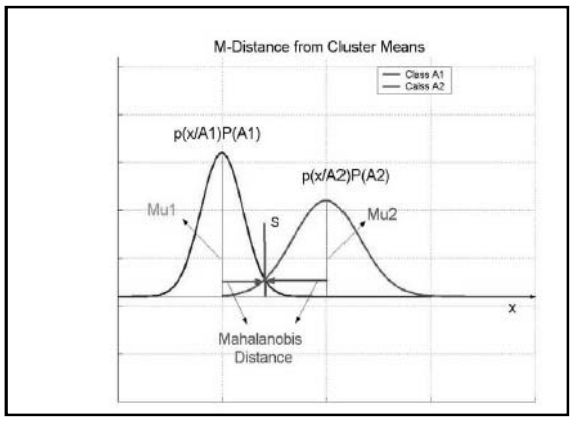


Figure 2:Mahalanobis distance (M-distance) from Class A1 and A2

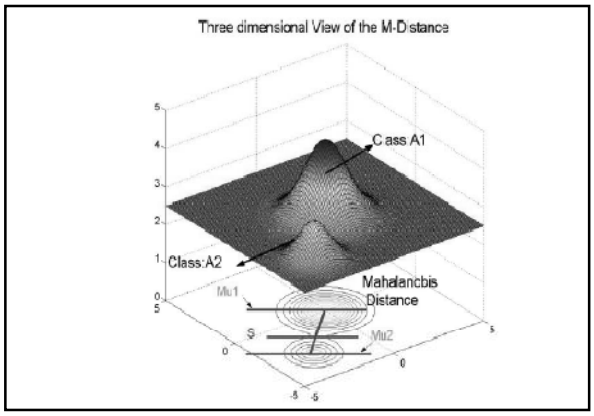


Figure 3:three dimensional view of the M-distance

Distance between S & Mu1, S &Mu2 : M- Distance ,Mu1, Mu2 : Means of the two classes A1 & A2

S : Decision surface The covariance matrix for each class is formed by the sample variance along pairs of directions in the input space[32]. For two a dimensional problems; that is, Class A1 and Class A2, the covariance matrix for each class are

$$\Omega_{A1} = \Omega_{A2} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}_{2\times 2} \tag{4}$$

The covariance matrix measures the density of samples of the data cluster in the radial direction from the cluster center in each dimension of the input space. So, it quantifies the shape of the data cluster. A careful observation to the covariance matrix reveals that the element σ_{12} is same as the element σ_{12}. This holds good even for the D dimensions of the elements in the input space. So, the modified covariance matrix for the two dimension problem is given as

$$\Omega = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ 0 & \sigma_{22} \end{bmatrix}_{2\times 2}$$

and for D-dimensions

$$\Omega = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1D} \\ 0 & \sigma_{22} & \cdots & \sigma_{2D} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \sigma_{DD} \end{bmatrix}_{D\times D} \tag{5}$$

The covariance matrix is always symmetric and positive definite (because of quadratic form and inverse always exist). It is positive definite, that is, the determinant is always greater than zero. The diagonal elements are the variance of the input data along each dimension. The off-diagonal terms are the covariance along pairs of dimensions. It is stated earlier that the placement of the decision region depends on three factors; that is, the distance between the class centers, the variance of each class centers and the threshold. The covariance matrix encapsulates the effect of covariance beautifully, but ignores other two factors completely. Hence, it is reasonable to incorporate the threshold term (bias) and the cluster mean to precisely determine the placement of the decision surface. Then the MM-distance is given by (3)

$$p(x) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Omega|^{\frac{1}{2}}} \exp\left(-\frac{(x-\boldsymbol{\mu})^T \Omega^{-1} (x-\boldsymbol{\mu})}{2} \right) \tag{6}$$

where T indicates the transpose, $|\Omega|$is the determinant of Ω, and Ω^{-1} the inverse of Ω which is given by (6)

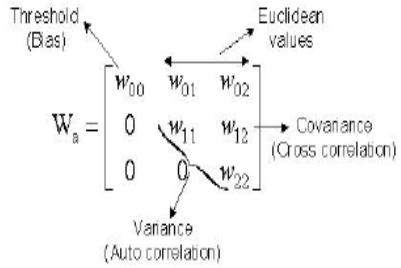
$$\Omega = \begin{bmatrix} \sigma_{00} & \sigma_{01} & \sigma_{02} & \cdots & \sigma_{0D} \\ 0 & \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1D} \\ 0 & 0 & \sigma_{22} & \cdots & \sigma_{2D} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_{DD} \end{bmatrix}_{[(D+1)\times(D+1)]}$$

The first row of the modified covariance matrix encapsulates the effect of the bias and cluster mean. The elements of the covariance matrix are the product of dispersions among sample pairs in the i^{th} and j^{th} coordinates:

$$\sigma_{ij} = \frac{1}{\Gamma - 1}\sum_{k=0}^{\Gamma}\sum_{m=k}^{\Gamma}(x_{i,k} - \mu_i)(x_{i,k} - \mu_j) \tag{7}$$

The covariance matrix Ω is an upper or lower triangle matrix that provides the sufficient condition for the placement of decision surface. The modified covariance matrix incorporates the effect of the Euclidean distance between the cluster centers, the threshold and the dispersion of the samples around the cluster mean which affects the placement of the thresholds for optimal classification. Hence, the structure of the covariance matrix is critical for the placement and shape of the discriminant functions in pattern space. Since the distance metric for classification is normalized by the covariance, if the class means stay the same but the covariance changes, the placement and shape of the discriminate function will change. Finally, it is concluded that the modified covariance matrix (weight matrix of the neural unit with QSO) is associated with the following terms

- Threshold (bias);
- Distance of the cluster means from the decision boundary (Euclidian distance);
- Covariance's (cross-correlation) among pairs of dimensions above the diagonal elements; and
- Variances (auto-correlation) of the input data of each dimension along the diagonal.



(8)

State Convergence for Symmetric Weight Matrix of (DTQNU)

Like continuous-time neural networks, one can also explore the global state

convergence of the discrete-time DNN with a symmetric weight matrix. For instances, when the synaptic weight matrix *W* is symmetric, that is, when $W = W^T$, in [27]it *was* proposed an energy function for the neural system given in (9), which is of the form

$$\begin{aligned} E(k) &= -\tfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_i w_{ij} y_i(k) y_j(k) - \sum_{i=1}^{n} y_i(k)\mu_i s_i \\ &\quad + \sum_{i=1}^{n}\int_{0}^{y_i(k)} \sigma_i^{-1}(\tau)d\tau \\ &= -\tfrac{1}{2}\boldsymbol{y}^T(k)\boldsymbol{\Psi W y}(k) - \boldsymbol{y}^T(k)\boldsymbol{\Psi s} + \sum_{i=1}^{n} G_i(y_i(k)) \end{aligned}$$

where

$$G_i(y_i) = \int_{0}^{y_i} \sigma_i^{-1}(\tau)d\tau$$

(9)

Using symmetry, $W_{ij} = W_{ji}$, the change in *E(k)* between the time *k* and *k* + 1, defined as *AE(k)* = *E(k* + 1) - *E(k)*, can be given as

$$\begin{aligned} \Delta E(k) &= -\tfrac{1}{2}\Delta\boldsymbol{y}^T(k)\boldsymbol{\Psi W}\Delta\boldsymbol{y}(k) - \Delta\boldsymbol{y}^{\boldsymbol{T}}(k)\boldsymbol{\Psi W y}(k) \\ &\quad - \Delta\boldsymbol{y}^T(k)\boldsymbol{\Psi s} + \sum_{i=1}^{n}\left[G_i(y_i(k+\mathbf{1})) - G_i(y_i(k))\right] \\ &= -\tfrac{1}{2}\Delta\boldsymbol{y}^T(k)\boldsymbol{\Psi W}\Delta\boldsymbol{y}(k) - \Delta\boldsymbol{y}^{\boldsymbol{T}}(k)\boldsymbol{\sigma}^{-1}(\boldsymbol{y}(k+1)) \\ &\quad + \sum_{i=1}^{n}\Delta G_i(k) \end{aligned}$$

where $\Delta\boldsymbol{y}(k) \triangleq \boldsymbol{y}(k+1) - \boldsymbol{y}(k)$ and

$$\Delta G_i(k) = G_i(y_i(k+1)) - G_i(y_i(k))$$

(10)

Considering up to the second derivatives, one obtains the following inequality [29]

$$\Delta G_i(k) \le G_i'(y_i(k+1))\Delta y_i(k) - \tfrac{1}{2}\min_{y_i}\left(\frac{d^2 G_i(y_i)}{dy_i^2}\right)\left[\Delta y_i(k)\right]^2$$

(11)

Where $\grave{G}_i(y_i(k + 1))$ is the derivative of $G_i(y_i)$ at the point $y_i = y_i(k + 1)$. Since the minimum curvature of $G_i$is given by the inverse number of the maximum slope of the function σ (.); that is 1, the minimum second derivative can be expressed as

$$\min_{y_i}\left(\frac{d^2G_i(y_i)}{dy_i^2}\right) = 1^{-1} = 1 \tag{12}$$

Eqns. (5.12)-(5.14) and equality $G_i(y_i) = \sigma_i^{-1}(y_i)$ from (12) Yield

$$\begin{aligned}\Delta E(k) &\leq -\tfrac{1}{2}\Delta \boldsymbol{y}^T(k)\boldsymbol{\Psi}\boldsymbol{W}\Delta \boldsymbol{y}(k) - \tfrac{1}{2}\Delta \boldsymbol{y}^T(k)\Delta \boldsymbol{y}(k)\\ &= -\tfrac{1}{2}\Delta \boldsymbol{y}^T(k)\left[\boldsymbol{\Psi}\boldsymbol{W} + \boldsymbol{I}\right]\Delta \boldsymbol{y}(k)\end{aligned} \tag{13}$$

If the matrix $\psi W+$ I is positive-definite; that is

$$\boldsymbol{W} + \boldsymbol{\Psi}^{-1} > 0$$

then

$$\Delta E(k) \leq 0; \quad \Delta E(k) = 0 \Longrightarrow \Delta \boldsymbol{y} = 0 \tag{14}$$

Therefore, all the attractors of the dynamic neural system described in (12) are fixed points, and the condition in (13) is a global convergence condition. A sufficient condition for $W + \psi^{-1}$ to be positive-definite is

$$|\lambda_{min}(\boldsymbol{W})| < \frac{1}{\mu} \tag{15}$$

Where $\lambda_{min}(W)$ represents the minimum Eigenvalue of the matrix *W*. If this condition is satisfied, the states of the system in (9) or (10) will always converge to one of their asymptotically stable equilibrium points regardless of the initial values of the states, [28]

Development of Dynamic Back-propagation (DBP) learningAlgorithms for DTQNU

Since the late 1980s, there has been much interest in developing learning algorithms that are capable of modeling time-dependent phenomena. In particular, considerable attention has been devoted to capturing the time-dependent dynamics of dynamic neural systems embedded in some known or observed temporal sequences. Note that this temporal learning can be applied for providing time-independent equilibrium neural outputs for time-independent inputs. The problem of temporal learning can typically be formulated as a minimization of an appropriate error index function over an arbitrary but finite time interval. The gradients of the index with respect to the parameters of the neural system are essential elements of the minimization process. We discuss the basic framework of temporal learning in a dynamic neural unit (DTQNU), [29], [30], [31] and [32].

Using Euler's method, the first-order derivative is approximated as

$$\left.\frac{dx}{dt}\right|_{t=kT} = \frac{x((k+1)T) - x(kT)}{T} \tag{16}$$

Where T is the sampling period and k is the sampling instant. If T = 1, this derivative can be approximated to

$$\frac{dx}{dt} = x(k+1) - x(k) \tag{17}$$

Thus, for a continuous-time dynamic neural unit (CT-DNU)

$$\begin{aligned} \frac{dx}{dt} &= -\alpha x(k) + wy(t) + s \\ y(t) &= \sigma(x(t)) \end{aligned} \tag{18}$$

the equivalent model of the DNU in discrete time (DTDNU) is given by

$$\begin{aligned} x(k+1) &= -(\alpha - 1)x(k) + wy(k) + s \\ y(k) &= \sigma(x(k)) \end{aligned} \tag{19}$$

The block diagram of the above discrete-time model is given in Figure (4). Usually, the discrete-time representations of dynamic neural systems may provide some computational advantages on digital computers.

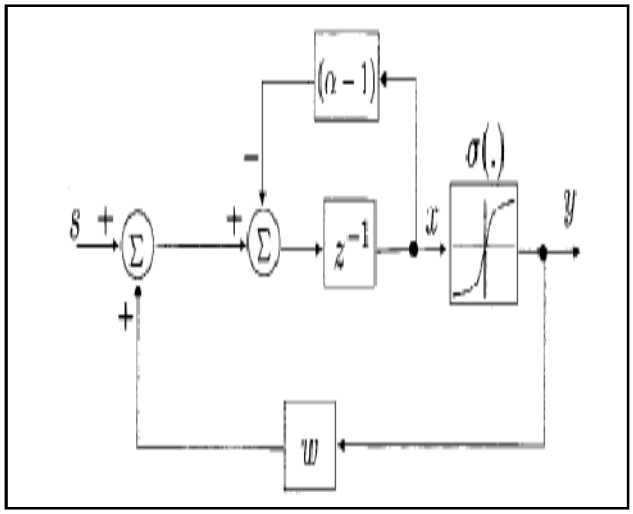


Figure 4:Block diagram of a discrete-time dynamic neural unit DTQNU

Given a finite length discrete-time sequence $X_d(k)$, k = 1 , 2 , . . . , N, we wish to design a discrete-time temporal learning algorithm such that the state of the following discrete-time dynamic neural unit (DTDNU)

$$\begin{aligned} x(k+1) &= -(\alpha-1)x(k) + \sum_{i=1}^{n} a_i \sigma(b_i x(k) + c_i) + s(k) \\ &= -(\alpha-1)x(k) + \boldsymbol{a}^T \boldsymbol{\sigma}(\boldsymbol{b}x(k) + \boldsymbol{c}) + s(k) \\ &= -(\alpha-1)x(k) + f(x(k), \boldsymbol{w}) + s(k) \end{aligned} \tag{20}$$

will asymptotically track the sequence $x_d(k)$ Here

$$f(x, \boldsymbol{w}) = \sum_{i=1}^{n} a_i \sigma(b_i x + c_i) = \boldsymbol{a}^T \boldsymbol{\sigma}(\boldsymbol{b}x + \boldsymbol{c}) \tag{21}$$

In this case, an error index with quadratic form is defined by

$$\begin{aligned} E(k) &= \tfrac{1}{2}(x_d(N) - x(N))^2 + \tfrac{1}{2}\sum_{k=0}^{N-1}[x_d(k) - x(k)]^2 \\ &= \tfrac{1}{2}e^2(N) + \tfrac{1}{2}\sum_{k=0}^{N-1} e^2(k) \end{aligned} \tag{22}$$

Where $e(k) = x_d(k) - x(k)$ and $e(N) = x_d(N)] - x(N)$. Using the discrete time variation principle, a discrete-time Lagrangian is defined by

$$\begin{aligned} \Phi &= \tfrac{1}{2}(x_d(N) - x(N))^2 + \sum_{k=0}^{N-1}\{\tfrac{1}{2}(x_d(k) - x(k))^2 \\ &\quad - z(k+1)[x(k+1) + (\alpha-1)x(k) - f(x(k), \boldsymbol{w}) - s(k)]\} \end{aligned} \tag{23}$$

$$\begin{aligned} &= \tfrac{1}{2}e^2(N) + \sum_{k=0}^{N-1}\{\tfrac{1}{2}e^2(N) - z(k+1)[x(k+1) \\ &\quad + (\alpha-1)x(k) - f(x(k), \boldsymbol{w}) - s(k)]\} \end{aligned} \tag{24}$$

The reason that the discrete time $(k + 1)$ is associated with the Lagrange multiplier is due to the simplicity of the final condition, as will be apparent in the following discussion. Like the method used for the continuous-time case, the first variation of ϕ may be represented as

$$\begin{aligned}\delta\Phi &= e(N)\delta x(N) + \sum_{k=0}^{N-1}\{e(k)\delta x(k) \\ &\quad - z(k+1)[\delta x(k+1) + (\alpha - 1)\delta x(k) + x(k)\delta\alpha \\ &\quad - f_x(x(k), \boldsymbol{w})\delta x(k) - \boldsymbol{f}_w(x(k), \boldsymbol{w})^T\delta\boldsymbol{w}]\} \\ &= e(N)\delta x(N) + \sum_{k=0}^{N-1}\{[e(k) - (\alpha - 1)z(k+1) \\ &\quad + f_x(x(k), \boldsymbol{w})z(k+1)]\delta x(k) - z(k+1)\delta x(k+1) \\ &\quad - z(k+1)x(k)\delta\alpha + z(k+1)(\boldsymbol{f}_w(x(k), \boldsymbol{w}))^T\delta\boldsymbol{w}\}\end{aligned} \tag{25}$$

Let the Lagrange multiplier *z(k)* satisfy

$$z(k) = e(k) + [f_x(x(k), \boldsymbol{w}) - (\alpha - 1)]z(k+1) \tag{26}$$

or

$$z(k+1) = \frac{z(k) - e(k)}{f_x(x(k), \boldsymbol{w}) - (\alpha - 1)} \tag{27}$$

Then

$$\begin{aligned}\delta\Phi &= e(N)\delta x(N) + \sum_{k=0}^{N-1}[z(k)\delta x(k) - z(k+1)\delta x(k+1) \\ &\quad - z(k+1)x(k)\delta\alpha + z(k+1)(\boldsymbol{f}_w(x(k), \boldsymbol{w}))^T\delta\boldsymbol{w}] \\ &= z(0)\delta x(0) - [z(N) - e(N)]\delta x(N) \\ &\quad + \sum_{k=0}^{N-1}[-z(k+1)x(k)\delta\alpha + z(k+1)(\boldsymbol{f}_w(x(k), \boldsymbol{w}))^T\delta\boldsymbol{w}]\end{aligned} \tag{28}$$

Since the initial value *x(0)* does not depend on the parameters, $\delta_x(0) = 0$. If we choose additionally the final condition of the Lagrange multiplier

$$z(N) = e(N) \tag{29}$$

then

$$\begin{aligned}\delta\Phi &= \sum_{k=0}^{N-1}[-z(k+1)\boldsymbol{x}(k)\delta\alpha + z(k+1)(\boldsymbol{f}_w(x(k),\boldsymbol{w}))^T\delta\boldsymbol{w}] \\ &= \left(\sum_{k=0}^{N-1} -z(k+1)x(k)\right)\delta\alpha + \left(\sum_{k=0}^{N-1} z(k+1)(\boldsymbol{f}_w(x(k),\boldsymbol{w}))^T\right)\delta\boldsymbol{w}\end{aligned} \tag{30}$$

Therefore, the partial derivatives of the error index with respect to the parameters are given by

$$\frac{\partial E}{\partial \alpha} = -\sum_{k=0}^{N-1} z(k+1)x(k) \tag{31}$$

$$\frac{\partial E}{\partial \boldsymbol{w}} = \sum_{k=0}^{N-1} z(k+1)\boldsymbol{f}_w(x(k),\boldsymbol{w}) \tag{32}$$

and the incremental terms of the parameters are

$$\Delta\alpha(k) = -\eta_\alpha\frac{\partial E}{\partial \alpha} = \eta_\alpha\sum_{k=0}^{N-1} z(k+1)x(k) \tag{33}$$

$$\Delta\boldsymbol{w}(k) = -\eta_w\frac{\partial E}{\partial \boldsymbol{w}} = -\eta_w\sum_{k=0}^{N-1} z(k+1)\boldsymbol{f}_w(x(k),\boldsymbol{w}) \tag{34}$$

That is, the updating equations are obtained as

$$\alpha(k+1) = \alpha(k) + \eta_\alpha\sum_{k=0}^{N-1} z(k+1)x(k) \tag{35}$$

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) - \eta_w\sum_{k=0}^{N-1} z(k+1)\boldsymbol{f}_w(x(k),\boldsymbol{w}) \tag{36}$$

The learning algorithm given above for such a fixed time sequence learning problem involves a discrete-time two-point boundary-value problem (TPB VP) that can be solved, in general, by reiterative technique. Here, the initial condition *x(0)* of the state is known, and the final condition *z(N)* of the Lagrange multiplier is a linear function of the unknown final condition *x(N)* of the state.

DTQNU for Satellite Identification

Satellites usually require attitude control equipment such as antennas, sensors, and

solar panels should be properly oriented. Antennas are pointed towards a particular location on the earth, while solar panels need to be oriented towards the sun for maximum power generation.

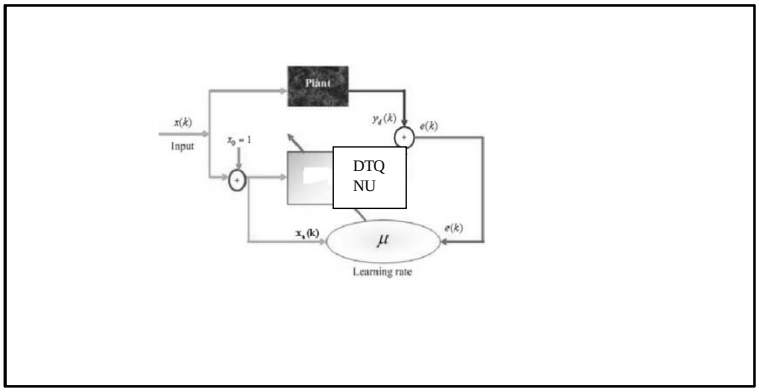


Figure 5:Identification of a Plant using a neural unit with QNU

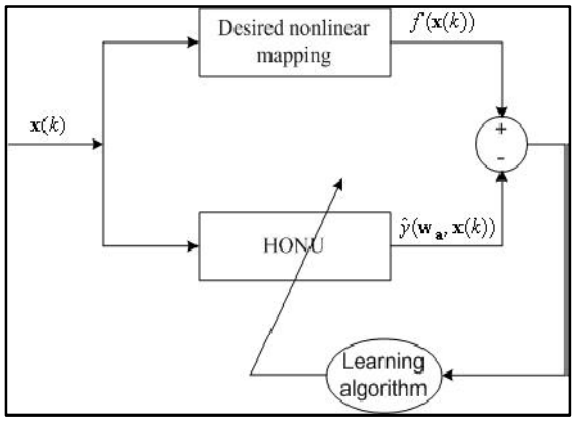


Figure 6:The learning scheme for functional approximation using a HONU.

In the particular case of satellite attitude control system, which is represented by a vector, which contains discrete values in consecutive steps? Other input values are introduced by the dynamic feedback y_k and $_{(k+1)}$, and the threshold constant u_0. Thus we have the input vector with four values (37). The input vector enters the aggregation function (38) neural units, so that after the breakdown of the aggregate functions depending on the number of inputs to the internal functions of the neural units contain sums of products of different weights to specific neural inputs. (39) Shows the broken aggregate function.

Defining the augmented vectors of neural inputs and neural weights, the synaptic operation for neural unit with DT-QNU is given as

$$f_{QNU} = x_{\alpha} \cdot W \cdot x_{\alpha}^{T} \tag{37}$$

Where

$$x_\alpha = [x_0\, u_{(k)}\, y_{(k)}\, y_{(k+1)}]^T \tag{38}$$

$$f_{QNU} = \sum_{i=0}^{n-1}\sum_{j=i}^{n-1} x_i x_j w_{ij} \tag{39}$$

$$, kde\, n = 4$$

Then from the mathematical model

$$\begin{aligned} f_{QNU} = {} & w_{00} + w_{01}u_{(k)} + w_{02}y_{(k)} + w_{03}y_{(k+1)} + w_{11}u_{(k)}^2 \\ & + w_{12}u_{(k)}y_{(k)} + w_{13}u_{(k)}y_{(k+1)} + w_{22}y_{(k)}^2 \\ & + w_{23}y_{(k)}y_{(k+1)} + w_{33}y_{(k+1)}^2 \end{aligned} \tag{40}$$

Neural weights matrix is given in (37).Complete the general registration of array aggregate functions is given in (38) and specific matrix notation for the discrete dynamic QNU is given in (40).

$$W^T = \begin{bmatrix} w_{00} & w_{01} & w_{02} & w_{03} \\ 0 & w_{11} & w_{12} & w_{13} \\ 0 & 0 & w_{22} & w_{23} \\ 0 & 0 & 0 & w_{33} \end{bmatrix} \tag{41}$$

$$f_{QNU} = \begin{bmatrix} u_0 & u_{(k)} & y_{(k)} & y_{(k+1)} \end{bmatrix} \cdot \begin{bmatrix} w_{00} & w_{01} & w_{02} & w_{03} \\ 0 & w_{11} & w_{12} & w_{13} \\ 0 & 0 & w_{22} & w_{23} \\ 0 & 0 & 0 & w_{33} \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ u_{(k)} \\ y_{(k)} \\ y_{(k+1)} \end{bmatrix} \tag{42}$$

$$f_{QNU} = \begin{bmatrix} u_0 & u_{(k)} & y_{(k)} & y_{(k+1)} \end{bmatrix} \cdot \begin{bmatrix} w_{00} & w_{01} & w_{02} & w_{03} \\ 0 & w_{11} & w_{12} & w_{13} \\ 0 & 0 & w_{22} & w_{23} \\ 0 & 0 & 0 & w_{33} \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ u_{(k)} \\ y_{(k)} \\ y_{(k+1)} \end{bmatrix} \tag{43}$$

Calculation of increases of neural weights in matrix notation, for a particular discrete dynamic QNU, given in (44).

$$\Delta W_{ij} = \mu \cdot e \cdot \begin{bmatrix} \frac{\partial f_{QNU}}{\partial w_{00}} & \frac{\partial f_{QNU}}{\partial w_{01}} & \frac{\partial f_{QNU}}{\partial w_{02}} & \frac{\partial f_{QNU}}{\partial w_{03}} \\ 0 & \frac{\partial f_{QNU}}{\partial w_{11}} & \frac{\partial f_{QNU}}{\partial w_{12}} & \frac{\partial f_{QNU}}{\partial w_{13}} \\ 0 & 0 & \frac{\partial f_{QNU}}{\partial w_{22}} & \frac{\partial f_{QNU}}{\partial w_{23}} \\ 0 & 0 & 0 & \frac{\partial f_{QNU}}{\partial w_{33}} \end{bmatrix} = \mu \cdot \epsilon \cdot \begin{bmatrix} 1 & u_{(k)} & y_{(k)} & y_{(k-1)} \\ 0 & u_{(k)}^2 & u_{(k)} y_{(k)} & u_{(k)} y_{(k-1)} \\ 0 & 0 & y_{(k)}^2 & y_{(k)} y_{(k-1)} \\ 0 & 0 & 0 & y_{(k-1)}^2 \end{bmatrix}$$

(44)

DTQNU Computer Simulation and Results

Continuous adaptation in real time, when the instantaneous input values calculated value of the output, and each step is converted into neural weights is not as accurate as the Batch-Training method, but it captures well the dynamics of the system and its main advantage is using the real-time when the input data may influence some disorders fluctuate.There will be identified satellite system using discrete dynamic QNU Figure (7). Unlike static QNU trained algorithm Levenberg -Marquardt, where the input data directly by three vectors u_k ,y_k and y_{k+1} and the dynamic QNU is the input vector only y(k) and the remaining values are fed into an aggregate function as a feedback output of the QNU.

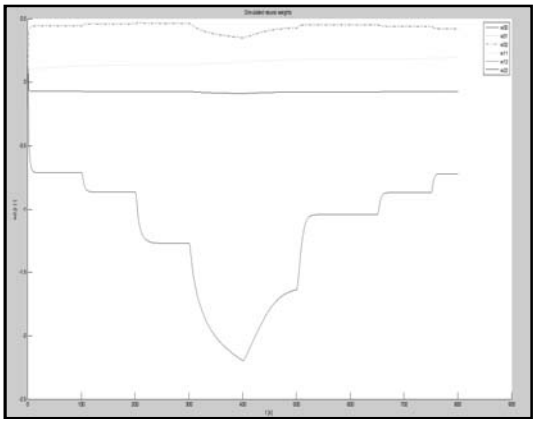


Figure 7: Application movements of DT-QNU weights

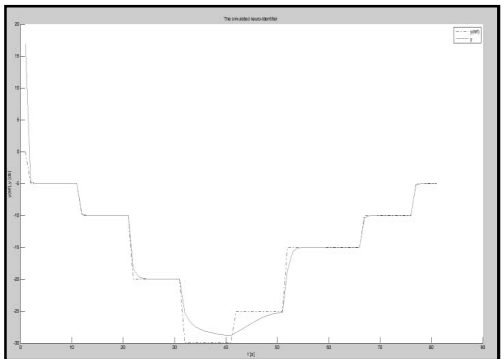


Figure 8:Application of DT-QNU Outputs $y_r$and y

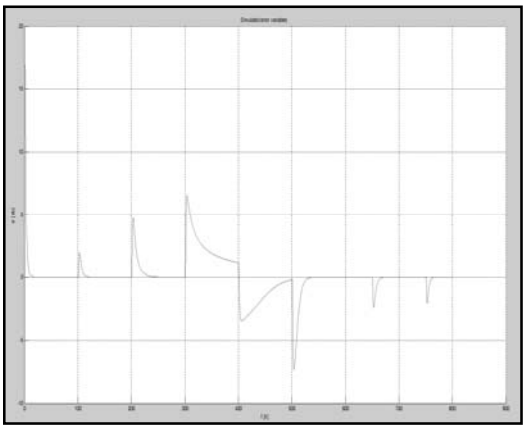


Figure 9: Application of DT-QNU errors

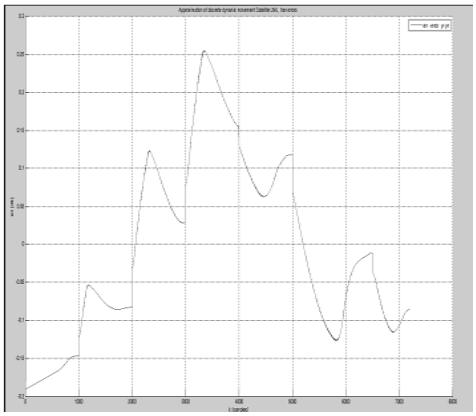


Figure 10:Application of DT-LNU errors

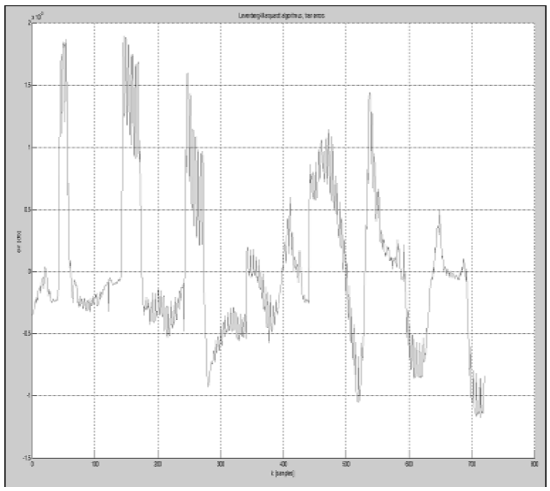


Figure 11:Application of Static-QNU errors

In Figure (8), shows a move y_r satellite approximations and discrete dynamic movement satellite QNU y_n. It is obvious that DT-QNU satellite approximates the course of movement very well. But there are visible tip. These cause a step change in input signal to each decision instant neural computing scales. However, the neural unit counter acts these changes and tries to immediately suppress and minimize. This fact was found by experiment, which was the duration of step changes in the input signal is turned off ($\mu = 0$) or minimizes the learning rate. Reducing or learning during off peak times resulted in longer lasting spike or a higher value and thus not be justified as desirable. It follows that neural drive is teaching seeks to minimize the peaks and as short as possible duration.

Suppression of the size of peaks is clearly visible in Figure (8), where it shows neural weights depending on the number of epochs of training the neural units. It is seen that particular scale neural W_{01}, W_{02} and W_{03} are in the beginning (during the first epoch), a relatively sharp upwards to approximately 20-epochs are sharp break occurs when the neural weights no longer pointing upward, and begin to converge to some value . This value can be achieved only after very many periods, but correct identification is important to converge. And this sharp break in the neural weights is to inhibit the growth of the size of peaks, which brings a jump in the input signal. While the size of the breakthrough peaks rising after the break, their growth stops or is very slight. Does this mean that the DT-QNU-identification system could be managed, along with step changes in inlet angles? Step changes (edges) of input signals are generally a problem not only for neural units, but in general engineering practice. The fact that the neural units by teaching seeks to minimize these effects, evidenced in their favor. A course error value is shown in Figure (9) and shows that the error is very small (in places where they do not show peaks from the edges of the input signal) and reaches only approximately 6.5% because DT QNU manages a very good approximation of the system.

Although, the BP learning algorithm provides a method for training MFNNs to accomplish a specified task in terms of the internal nonlinear mapping

representations, it is not free from problems. Many factors affect the learning performance and must be dealt with in order to have a successful learning process. Mainly, these factors include the initial parameters, learning rate and network size. A good choice of these items may greatly speed up the learning process to reach the target.

The essential difference between the two approaches lies in the manner in which the discrete approximation is made. The discrete-time DBP algorithm yields a TPBVP in the form of a set of nonlinear difference equations whose solution is precisely the solution that optimizes the stated discrete temporal learning problem. The continuous-time DBP algorithm yields a TPBVP in the form of a set of nonlinear differential equations whose solution is precisely the solution that minimizes the stated continuous-time temporal learning problem. The solution of a discrete version of this continuous time DBP yields a temporal state trajectory that does not optimize either the continuous-time problem or a discrete-time version of the continuous-time problem. For most situations, this creates no difficulties.

Error Calculations

The error between two signal of satellite and all neural types output was shown in Figures (7),(8),(9),(10) and (11) respectively. The corresponding Table error is shown in Table (1) as follows:

NN Type	Stable	Unstable	Unstable with Noise
DT-QNU	-0.000342803	-0.000342803	-0.412376
DT-LNU	-5.55844e+068	-5.55844e+068	-5.55844e+068
Static QNU	0.0405782	0.0663567	0.0615402

Table 1: NNS Error Table

$$Error=y_r-y_n \quad (45)$$

Where :y_r is Satellite output,
y_n: Neural Unit output

Experimental results reveal that the results obtained by the new DTQNU are quite superior to those obtained by other competitive approaches. The proposed DTQUU has revealed its optimality, efficiency and consistency. Related to optimality the DTQNU proved to have the very low quantization error per pixel for the input. We have discovered the consistency of the DTQNU due to its robustness to the variations of Noise parameter and also due to insensitivity to the initial conditions.

Concerning DT-LN Genetic controller, it has achieved very large negative computation error i.e., observing output results were very high than the expected satellite results

Finally, notice that static-QN Genetic approach output error is somehow close for the satellite out

CONCLUSION

A novel DTQNU has been proposed for nonlinear unknown dynamic system identification problems.The concept of the neural unit with DTQNU appears to be promising as it can process lower and higher-order inputs similar to the processing function of the biological neuron. The advantages of our approach include:: firstly; the realization of a Modified Mahalanobis Distance (MMD) to reduce the number of adaptable weights without sacrificing the neural performance; secondly; the output depends not only on the current input to the Discrete Time Quadratic Neural Unit , but also on the current or previous inputs, outputs, or states of the Neural Unit and always will converge to one of their asymptotically stable equilibrium points regardless of the initial values of the inputs. Thirdly; the proposed Discrete Time Quadratic Neural Units is capable of accurately identifying of nonlinear dynamic systems using fewer parameters.Even though the developed nonconventional neural architectureDTQNU is very promising universal approximators of complex systems and work very promisingly for technical systems, its use does not outperform the proper derivation of a mathematical model of a complex system if such an analysis can be done. Even though application of the proposed neural units to systems that are difficult to analyze is believed to introduce considerable improvements, the combination of customization of the internal neural architecture together with the proper mathematical analysis of a system can maximize accuracy of the neural units; minimize the time of adaptation, and find initial neural parameters from which the unit would converge to a more the primary challenges of further research regarding the proposed neural units in common engineering problems have to to search of research of neural units with an adaptable signal input preprocessor for identification of unknown system input signals for the purpose of advanced monitoring of internal as well as external system perturbations.

Reference

[1] S. Kalogirou, Artificial intelligence for the modeling and control of combustionprocesses: A Review. Progress in Energy and Combustion Science 29(2003)515–566.

[2] R. Kamwa, V. Grondin, K. Sood, C. Gagnon, V. Nguyen, J. Mereb, Recurrentneural Networks for phasor detection and adaptive identification in power system controland protection.IEEE Transactions on Instrumentation and Measurement 45(1996)657–664.

[3] N. Karayiannis, A. Venetsanopoulos, Artificial Neural Networks: LearningAlgorithms. Performance Evaluation and Applications. Kluwer Academic Publishers,Boston, USA,1993.

[4] L. Medsker, L. Jain, Recurrent neural networks: design and applications, BocaRaton, FL:CRC Press, 2000.

[5] J. Racine, On The Nonlinear Predictability of Stock Returns Using Financial andEconomic Variables, forthcoming. Journal of Business and Economic Statistics 19(2001) 80-382.

[6] M. Sen, K. Yang, Applications of artificial neuralnetworks and genetic algorithms in thermal Engineering,in: F. Kreith (Ed.), CRC Handbook of Thermal Engineering,2000, pp. 620±661 (Section 4.24).
[7] M. Spiga, G. Spiga, Step response of the cross low heatexchanger with finite wall Capacitance,Int. J. Heat MassTransfer 35 (2) (1992) 559±565.
[8] W. Roetzel, Y. Xuan, Dynamic Behaviour of HeatExchangers, WIT Press, Boston, 1999.
[9] H. Thal-Larsen, Dynamics of heat exchangers and theirmodels, ASME J. Basic Eng. (1960) 489±504.
[10] H. Yamashita, R. Izumi, S. Yamaguchi, Analysis of thedynamic characteristics of cross-¯ow heat exchangers withboth fluids unmixed, Bull. JSME 21 (153) (1978) 479±485.
[11] R. Hecht-Nielsen, Kolmogorov's mapping neural networkexistence theorem, Proceedings of The First InternationalConference on Neural Networks, IEEE 3 (1987) 11±13.
[12] G. Deõaz, M. Sen, K. Yang, R. McClain, Simulation ofheat exchanger performance by artificial neural networks,Int. J. HVAC and R Res. 5 (3) (1999) 195±208.
[13] S. Bittanti, L. Piroddi, Nonlinear identification and controlof a heat exchanger: a neural network approach, J.Franklin Inst. 334B (1) (1997) 135±153.
[14] G. Deõaz, M. Sen, K.T. Yang, R. McClain, Use ofarti®cial neural networks for Temperaturecontrol, in:Proceedings of the Fourth ISHMT/ASME Heat and MassTransfer Conference, Pune, India, 5±7 January 2000.
[15] H. AI-Duwaish, M. Karim, and V. Chandrasekar, Use of multilayer feedforwardneural networks in identification and control of Wiener model, *IEE Proceedings ofControl Theory and Applications*, Vol. 143, 1996, pp. 255-258.
[16] G. Chen, Y. Chen, and H. Ogmen, Identifying chaotic systems via a Wiener-typecascade model, *IEEE Control System Magazine*, Vol. 17, 1997, pp. 29-36.
[17] T. Clarke and X. Sun, Minimal state-space model realization of a nonlinear helicopter, *IEE Proceedings of Control Theory and Applications*, Vol. 145, 1998, pp.415-422.
[18] Y. Chen and J. Wang, An automated Hammerstein recurrent neural networkfor dynamic Applications, in *Proceeding of IASTED International Conference onComputational Intelligence*, 2005, pp. 193-198.
[19] B. Ho and R. Kalman, Effective construction of linear state-variable modelsfrom input/output data, in *Proceedings of the 3rd Annual Allerton Conference onCircuit and System Theory*, 1965, pp. 449-459.
[20] J. Juang and R. Pappa, An eigensystem realization algorithm for modal parameter identification and model reduction, *Journal of Guidance*, Vol. 8, 1985, pp.620-627.
[21] J. Juang, M. Phan, L.Horta, and R. Longman, Identification of observer/Kalman filter Markov parameters: theory and experiments, *Journal of Guidance, Control, and Dynamics*, Vol. 16, 1993, pp. 320-329.

[22] J. Juang and M. Phan, Linear system identification via backward-time observermodels, *Journal of Guidance, Control, and Dynamics*, Vol. 17, 1994, pp. 505-512.

[23] A. Janczak, *Identification of Nonlinear Systems Using Neural Networks and Polynomial Models*, Springer-Verlag, New York, 2005.

[24] A. Kalinli and S. Sagiroglu, Elman network with embedded memory for system identification, *Journal of Information Science and Engineering*, Vol. 22, 2006, pp.1555-1568.

[25] C. Lin, Wavelet neural networks with a hybrid learning approach, *Journal ofInformation Science and Engineering*, Vol. 22, 2006, pp. 1367-1387.

[26] P. Churchland, Neuro-philosophy, MIT Press, Cambridge, MA,2000.

[27] M. Hiramoto, Y. Hiromi, E.Giniger andY. Hotta, The Drosophila Netrin Receptor Frazzled Guides Axons by Controlling Netrin Distribution. Nature, Vol. 406, No. 6798, pp. 886-888,2000.

[28] K. Kohara, A. Kitamura, M.Morishima, and T. Tsumoto, Activity-Dependent Transfer of Brain-Derived Neurotrophic Factor to Postsynaptic Neurons.Science, Vol. 291 (March), pp. 2419-2423,2001.

[29] C. Marcus, and R.Westervelt, Dynamics of Iterated Map Neural Networks. Phys. Rev. A, Vol. 40, No. 1, pp. 577-587,1989.

[30] D.C.Dracipoulos and A.Jones(1994).Neuro-Genetic Adaptive Attitude Control.Neural Comp.&Applic.,2(40:183-204,1994.

[31] I.Bukovsky.Modeling of complex Dynamic systems by non conventional Artificial Neural Archtitectures and adaptive approaches to evalution of chaotic time series.PhD Dissr.Chic Technical University,Faculty of mechanical Engineering,2007.

[32] S. Kumar.Development of Neural Units with Higher-Order Synaptic Operations and their Applications to Logic Circuits and Control Problems.Msc Thesis, Department of Mechanical Engineering, University of Saskatchewan,2004