Pergamon

PII: S0098-1354(96)00281-5

Computers chem. Engng Vol. 21, No. 4, pp. 371–385, 1997
Copyright © 1996 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0098-1354/97 $17.00+0.00

A DYNAMIC NEURAL NETWORK APPROACH TO NONLINEAR PROCESS MODELING

ANDRE M. SHAW[a], FRANCIS J. DOYLE III[a] and JAMES S. SCHWABER[b]

[a] *School of Chemical Engineering, Purdue University, West Lafayette, IN 47907-1283, USA*

[b] *E.I. DuPont de Nemours and Co., Inc., Wilmington, DE 19880, USA*

(Received 7 July 1995; accepted 28 December 1995)

Abstract—The use of feedforward neural networks for process modeling has proven very successful for steady-state applications, but suitable applications for dynamic systems are still being studied. A novel approach is presented in this paper which uses intrinsically dynamic neurons inspired from biological control systems as the processing elements in network architectures. This results in a network which incorporates dynamic elements with continuous feedback. Two case studies show that the recurrent dynamic neuron network (RDNN) does an excellent job of predicting nonlinearities such as asymmetric dynamic response. In addition, the RDNN significantly outperforms linear models and more traditional neural network models for open-loop simulations. Finally it is shown how this RDNN model can be used in model-based control architectures, such as internal model control. Copyright © 1996 Elsevier Science Ltd

Keywords: dynamic neural networks; nonlinear systems; process identification

1. INTRODUCTION

The opportunities for novel process engineering methodologies inspired by biological control systems has received increasing attention. There remains a great incentive to exploit the highly efficient and robust computational mechanisms of natural neuronal processing systems in an effort to improve the tools of process modeling and control. The motivation for this goal follows from an analysis of the numerous parallels in the dynamic attributes of process and biological systems, as well as the parallels in the performance requirements (Henson *et al.*, 1994; Stark, 1993).

Although the details are not well understood, it is acknowledged that biological control systems provide robustly stable control of highly nonlinear plants (Henson *et al.*, 1994). Furthermore, these control systems function properly under such adverse conditions as major sensor damage and/or loss. It is observed that the control actions involve many regulatory mechanisms operating on different, but relatively short, time scales to achieve the desired response. The central nervous system (CNS) serves as the main control center for these regulatory mechanisms and coordinates the various control activities of biological systems.

The fundamental component of the central nervous system is the neuron, which is responsible for the rapid and accurate transfer of information between the CNS and the other parts of the body. Hence, a detailed analysis of neuron functionality—how it encodes and transmits information—will lead to increased insight into the nature of these highly efficient biological computational systems. The coupling of these concepts with engineering principles can enable a reverse engineering of the biological computational elements for process modeling and control applications.

1.1. *Previous approaches to dynamic neural networks*

One of the more popular applications of biological understanding to engineering has been the artificial neural network. Static or feedforward artificial neural networks (FANNs) have emerged as useful tools in chemical engineering systems applications including: (i) steady state modeling; (ii) steady state planning and; (iii) steady state optimization (for representative references see MacGregor *et al.* (1991) on the connection to standard statistical regression tools; Pollard *et al.* (1991) on process identification and control). All of the above represent highly nonlinear but *static* (steady state) problems. In reality, chemical process operations are highly nonlinear as well as highly *dynamic*, and thus networks structures must be modified in other to properly model dynamic systems.

FANNs have been used as the static nonlinearity in Hammerstein and Wiener models (Montague *et al.*, 1991; Narenda and Parthasarathy, 1990) to model highly nonlinear, dynamic systems. The FANN is placed in series with a linear dynamic element to capture nonlinear dynamics. There are, however, processes in which the dominant nonlinearities cannot be separated as a distinct static element, thus there is a motivation to pursue a more general approach.

One approach for introducing dynamics borrows from classical time series analysis (Morris *et al.*, 1994). The idea is to replace static input/output data with appropriate time histories over a window of discrete times.

One possible outcome is a constrained Volterra model, a temporal equivalent of a Taylor series expansion for spatial function approximation, which does not involve feedback of the output time history. The main deficiency of this approach lies in the fact that utilizing time histories to incorporate dynamics is, in this context, both awkward and arbitrary. The most important consequences are that

1. the computational requirements become substantial and unwieldy;
2. the resulting models are grossly overparameterized; and
3. the reliability of such non-parsimonious models, obviously dependent on the arbitrary choice of the time history window, is questionable.

There is also the so-called Memory Neuron Networks (MNN) (Sastry *et al.*, 1993). Here, each neuron has associated with it a memory neuron whose single output summarizes the history of the past activations of the neuron. These memory neurons represent the trainable dynamical elements of the model. Since the connections between a neuron and its memory neuron involve feedback loops, the overall network is now a recurrent one. The overall network structure is similar to the FANN structure, and thus simple incremental learning algorithms based on backpropagation are possible.

Another approach introduces a neuron model (Rao and Gupta, 1993) called the dynamic neural unit (DNU). The DNU is comprised of two distinct operations: (1) the synaptic operation, which determines the optimum feedforward and feedback synaptic weights controlling the dynamics of the neuron, and (2) the somatic operation which determines the optimum gain of the nonlinear activation function for a given task. The function approximation for a network of DNUs is done using linear and trigonometric operators. The development of the DNU is motivated by biological neuronal systems that function with feedback, but its dynamic structure is analogous to a reverberating circuit and does not represent any specific anatomical region within the biological nervous system. The dynamic structure, which is responsible for the synaptic operation, can be represented by a second order network structure with two poles and two zeros.

The most popular approach for introducing dynamics in a network architecture is to connect a feedback path from the network outputs to its input. In feedback networks, the goal is to achieve an asymptotically stable solution that is a local minimum of the dissipated energy function (Zurada, 1992). To recall information stored in these dynamic networks, an initializing input pattern is applied to the network to generate a corresponding output. The initializing input is then removed and the initial output then becomes the new input. This sequential updating continues until the network reaches its equilibrium.

The majority of research on feedback neural systems stems from work published by John Hopfield in the mid-eighties (Hopfield and Tank, 1986), who originated the idea of recurrent networks. The continuous time representation of a single neuron in a recurrent network is given by the following nonlinear differential equation:

$$C_i \frac{dx_i}{dt} = -\frac{x_i}{R_i} + \sum_{j=1}^{n} w_{ij} h(x_j) + u_i \qquad (1)$$

Here w_{ij}, x_i, and u_i are the weights, states and inputs of the network, respectively and n the number of neurons in the layer. R_i and C_i are constants, and $h(x)$ is a threshold function, typically the hyperbolic tangent function or the sigmoidal function, $h(x) = \frac{1}{1+e^{ax}}$. A range of process dynamics can be captured with this structure because of its recurrent nature. However, it is important to note that the structure has a single nonlinearity, hence the gain and time constant of the individual neuron are inherently coupled.

Like the Hopfield network, the proposed approach in this paper is motivated by biological neuronal systems, where the neurons themselves are inherently dynamic. The motivation for such an approach over the standard delayed recurrent approach has been voiced by several researchers in this area (Chappelier and Grumbach, 1994; Scott and Ray, 1994): (i) the resulting models contain fewer parameters, and (ii) the resulting models are in a standard form for the application of model-based control synthesis schemes such as feedback linearization.

2. DYNAMIC NEURAL NETWORK STRUCTURE

In an effort to enrich the dynamic behavior of the Hopfield network, it is proposed that an independently nonlinear *gain* and *time constant* in a single neuron will give rise to rich network behavior with relatively few neurons. In fact, a biological counterpart to such dynamic behavior has been proposed to describe the role of autoregulation in a neuron cell played by intracellular calcium. A plausible mechanism for this autoregulation has been proposed by Abbott and co-workers (Abbott and LeMasson, 1993) which builds upon the Hodgkin–Huxley formalism for biophysical neuron models (Hodgkin and Huxley, 1952). In their model, the maximal channel conductances $\bar{G}$ are proposed to be functions of intracellular calcium concentration $[Ca]$. The mathematical formulation of their mechanism involves modulation by $[Ca]$ of both the time constant and gain of a simple first order rate law for the dynamics of the channel conductance:

$$\tau([Ca])\frac{d\tilde{G}}{dt}=F([Ca])-\tilde{G} \qquad (2)$$

where $F([Ca])$ is the asymptotic value of $\tilde{G}$. This mechanism was found to maintain a constant average level of activity over a wide range of conditions with the intracellular calcium concentration acting as a feedback element linking maximal conductances to electrical activity.

If one reformulates the dependence of the conductance signal on calcium concentration implicitly as a modification of the gain and the time constant, then one arrives at a simple empirical model:

$$\tau(y)\cdot\dot{y}=-y+k(y)\cdot u \qquad (3)$$

This expression corresponds to a dynamical system with independent nonlinear gain and time constant. A similar representation has, in fact, been proposed for distillation column modeling (Chien and Ogunnaike, 1992). In that work the functions $\tau(u)$ and $k(y,u)$ were empirically identified. In the present study, we propose the use of this element in simple network configurations to elicit a richer range of dynamic response.

2.1. Empirical neuron structure

The structure described in equation (3) will be defined to be a first order *dynamic neuron* in the remainder of this paper. The nonlinear character of a network of such neurons is strongly influenced by the functional form chosen for the nonlinear dependence of the gain and time constant on the process state. Our initial studies employ a simple Taylor series expansion about the output, y, for these nonlinearities, leading to the specific structure:

$$(\tau_0+\tau_1 y+\cdots)\dot{y}=-y+(k_0+k_1 y+\cdots)u \qquad (4)$$

The degree of nonlinearity is governed by the number of terms considered in the expansion. For example, input and output multiplicities can be induced through the correct choice of the neuron nonlinearity (Beer, 1994). The sigmoidal functional form was also investigated, as it is known to be displayed by biological systems and is commonly found in the nodal representations for FANNs and Hopfield networks. Here $\tau(y)$ and $k(y)$ are represented by:

$$\tau(y)=\frac{1}{1+\exp(-\lambda_1 y)} \qquad (4)$$

$$k(y)=\frac{1}{1+\exp(-\lambda_2 y)} \qquad (5)$$

This representation did not prove to be richer in modeling process systems than the Taylor series-based neurons.

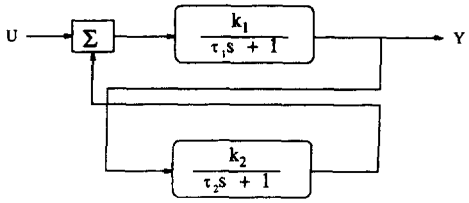

Fig. 1. Dynamic recurrent network model.

2.2. Network architecture

In this section, we consider the general behavior of a recurrent network arrangement of first order dynamic neurons. Consider, for example, a recurrent arrangement of two dynamic neurons in which the Taylor series functions for the gain and time constant are both truncated at order zero. This corresponds to the linear operator which represents the Jacobian of a network of more general nonlinear neurons. The network model is depicted in Fig. 1, and can be represented by the following equation:

$$G(s)=\frac{k_1(\tau_2 s+1)}{\tau_1\tau_2 s^2+(\tau_1+\tau_2)s+1-k_1k_2} \qquad (6)$$

where $G(s)$ is the transfer function which relates the output y to the input u. This network architecture is a Hopfield type structure (Hopfield and Tank, 1986), which relies on its recursive nature for dynamic representation. This structure is general in that it can represent relative degree one linear systems with order less than or equal to two (excluding pure capacitance systems):

•*First order, relative degree* 1

$$G(s)=\frac{K}{\tau s+1} \qquad (7)$$

$$(k_1=K,\ \tau_1=\tau,\ k_2=\tau_2=0)$$

•*Second order, relative degree* 1

$$G(s)=\frac{K(\xi s+1)}{\tau^2 s^2+2\tau\zeta s+1} \qquad (8)$$

$$(k_1=\frac{\xi^2 K}{(2\zeta\xi-\tau)\tau},\quad \tau_1=\frac{\xi\tau}{2\zeta\xi-\tau},\quad k_2=\frac{2\xi\zeta\tau-\tau^2-\xi^2}{\xi^2 K},$$

$$\tau_2=\xi)$$

(The singularities in the latter transformation correspond to the cases where the order is not two (τ=0); the relative degree is not one (ξ=0); a trivial gain (K=0); and the case where the sum of the poles is equal to the process zero.)

$$\left(\frac{2\zeta}{\tau}=\frac{1}{\xi}\right)$$

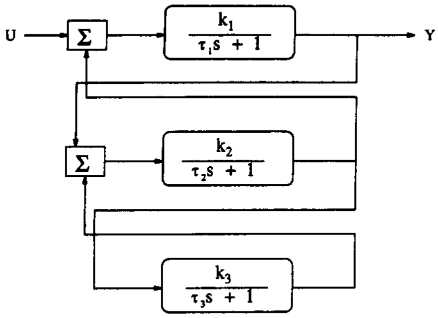

Fig. 2. Generalized 3 neuron structure for dynamic neural network.

For a system which contains more than two neurons, the level of network connectivity is not as straightforward as in the two neuron case. For the three neuron case (Fig. 2), several architectures are possible such as a pyramid structure where each neuron receives the external input, but only one (the neuron whose output is the network output) receives outputs from the other two. There is also the fully recurrent structure where each neuron directly receives outputs from the other neurons plus the external input. In the end the following structure was chosen:

$$\begin{aligned}\tau_1(\cdot)\dot{x}_1 &= -x_1+k_1(\cdot)(u+x_2)\\ \tau_2(\cdot)\dot{x}_2 &= -x_2+k_2(\cdot)(x_1+x_3)\\ \tau_3(\cdot)\dot{x}_3 &= -x_3+k_3(\cdot)(x_2)\\ y &= x_1 \qquad (9)\end{aligned}$$

where $\tau_i(\cdot)$ and $k_i(\cdot)$ can be nonlinear and time-varying. This structure represents a natural extension (in terms of connectivity) of the two neuron network above and generalizes in a straightforward manner to higher order networks. The Jacobian linearization of this structure can be case in standard state–space form:

$$\begin{aligned}\dot{x} &= Ax+Bu\\ y &= Cx \qquad (10)\end{aligned}$$

This system can be transformed into the *controllable canonical* form:

$$\dot{x} = \hat{A}x+\hat{B}u \qquad (10)$$

$$y = \hat{C}x \qquad (11)$$

using the following (invertible) coordinate transformation:

$$\hat{A} = T^{-1}AT \qquad (11)$$

$$\hat{B} = T^{-1}B \qquad (11)$$

$$\hat{C} = CT \qquad (12)$$

where T is a nonsingular transformation matrix. In the case of a network of linear dynamic neurons, a linear identification procedure can be used to determine the coefficients in equation (11) from input/output data, and the set of 15 equations and 15 unknowns in equation (12) can be solved simultaneously to realize the elements in A, B and C. This linear identification problem will determine the starting point for a more general nonlinear network identification problem in Section 2.4. The linear transfer function relating the output y to input u in Fig. 2 is:

$$G(s) = \frac{k_1(\tau_2\tau_3 s^2+(\tau_2+\tau_3)s+1-k_2k_3)}{(\tau_1 s+1)(\tau_2 s+1)(\tau_3 s+1) - (k_2k_3\tau_1+k_1k_2\tau_3)s-k_2k_3-k_1k_2} \qquad (13)$$

This structure is also general in that it can represent relative degree one linear systems with order less than or equal to three (excluding pure capacitance systems):

•*First order, relative degree* 1

$$G(s) = \frac{K}{\tau s+1} \qquad (14)$$

$$(k_1=K,\ \tau_1=\tau,\ k_2=\tau_2=k_3=\tau_3=0)$$

•*Second order, relative degree* 1

$$G(s) = \frac{K(\xi s+1)}{\tau^2 s^2+2\tau\zeta s+1} \qquad (15)$$

$$\left(k_1=\frac{\xi^2 K}{(2\zeta\xi-\tau)\tau},\ \tau_1=\frac{\xi\tau}{2\zeta\xi-\tau},\ k_2=\frac{2\xi\zeta\tau-\tau^2-\xi^2}{\xi^2 K},\right.$$

$$\left.\tau_2=\xi,\ k_3=\tau_3=0\right)$$

•*Third order, relative degree* 1

$$G(s) = \frac{s^2+b_2 s+b_1}{a_1 s^3+a_2 s^2+a_3 s+a_4} \qquad (16)$$

$$\left(k_1=\frac{1}{a_2-a_1b_2},\right.$$

$$\tau_1=\frac{a_1}{a_1b_2-a_2},$$

$$k_2=\frac{a_3^2-2a_1a_3b_1+a_1^2b_1^2-2a_2a_3b_2+2a_1a_2b_1b_2+a_2^2b_2^2+2a_1a_3b_2^2-a_1^2b_1b_2^2+a_1^2b_2^4}{a_4-a_2b_1-a_3b_2+2a_1b_1b_2+a_2b_2^2-a_1b_2^3},$$

$$\tau_2=\frac{a_3-a_1b_1-a_2b_2+a_1b_2^2}{a_2b_1+a_3b_2-2a_1b_1b_2-a_4-a_2b_2^2+a_1b_2^3},$$

$$k_3=\frac{a_4^2+a_3^2b_1-2a_2a_4b_1+a_2^2b_2^2-2a_1a_3b_1^2+a_1^2b_1^3-a_3a_4b_2-a_2a_3b_1b_2+3a_1a_4b_1b_2-a_1a_2b_1^2b_2+a_2a_4b_2^2+a_1a_3b_1b_2^2-a_1a_4b_2^2}{(a_4-a_2b_1+a_1b_1b_2)(a_3-a_1b_1-a_2b_2+a_1b_2^2)^2},$$

$$\tau_3=\left(\frac{a_3-a_1b_1-a_2b_2+a_1b_2^2}{a_4-a_2b_1+a_1b_1b_2}\right)$$

The analysis in the following subsection is restricted to the two and three neuron cases, however higher order generalizations follow directly from these cases.

2.3. *Mathematical properties of the proposed network models*

In this section, we summarize some of the mathematical properties of a network arrangement (Fig. 1) of the neurons proposed in equation (4). A network based on this structure will be subsequently referred to as the *Recurrent Dynamic Neuron Network* (RDNN). For purposes of comparison, a similar analysis will be applied to a simple arrangement of a FANN followed by a layer of first order linear dynamic nodes (as in Montague *et al.*, 1991).

In Funahashi and Nakamura (1993), it was shown that a special class of Hopfield networks possess the property that it can approximate the finite time trajectory of an n-dimensional dynamical system. For the purposes of the present study, we compare the approximation properties of two different dynamic network architectures by examining the structural properties of their Jacobian approximation.

2.3.1. Proposition 1 (single layer feedforward ANN)

A single layer feedforward network consisting of weighted summing nodes and N linear-first-order-dynamic-filters yields an overall dynamic model with arbitrary order $n\leq N$ and arbitrary relative degree $0\leq r\leq n-1$. However, the general model is restricted in the placement of the system eigenvalues to locations on the real axis.

2.3.2. Proposition 2 (fully recurrent single layer RDNN)

A single layer network consisting of weighted summing nodes and N recurrent linear-first-order-dynamic-neurons yields an overall dynamic model with arbitrary order $n\leq N$ and relative degree $r=1$. Furthermore, the general model can lead to arbitrary placement of the system eigenvalues in the complex plane.

2.3.3. Results for single layer feedforward ANN

Each output channel for this network structure consists of the summed outputs of the neurons (via linear dynamic filters) in the single hidden layer. If k_{fi} represents the Jacobian approximation of the static FANN, and k_i and τ_i are the elements in the layer of dynamic elements; then, for example, in the case of a two neuron network the output y is:

$$y=k_{f1}\frac{k_1}{\tau_1s+1}u+k_{f2}\frac{k_2}{\tau_2s+1}u$$

which for n neurons generalizes mathematically to:

$$y=\frac{\Sigma_{i=1}^{n}k_{fi}k_i\Pi_{j\neq i}^{n}(\tau_js+1)}{\Pi_i^n(\tau_is+1)}u \qquad (17)$$

This is a simple series representation of n linear filters and it clearly represents overdamped, high order linear systems with all of the poles on the real axis. Furthermore, it is straightforward to show that the relative degree of this equation will be less than or equal to $n-1$.

2.3.4. Results for fully recurrent single layer RDNN

As seen in Fig. 2, the network output is equal to the output of the first neuron. For the two and three neuron cases, the transfer function from the input u to output y is given by the expressions, respectively:

$$y=\frac{k_1(\tau_2s+1)}{(\tau_2s+1)(\tau_1s+1)-k_1k_2}u$$

$$y=\frac{k_1(\tau_2s+1)(\tau_3s+1)-k_1k_2k_3}{(\tau_1s+1)(\tau_2s+1)(\tau_3s+1)-k_1k_2(\tau_3s+1)-k_2k_3(\tau_1s+1)}u$$

The generalization to n neurons is straightforward, but has a non-trivial series representation. By considering the coefficients of the highest order terms in s, in the numerator and denominator, y can be approximated mathematically by:

$$y=\frac{k_1\Pi_{j=2}^{n}\tau_js^{n-1}+O^{(n-2)}(s)}{\Pi_{i=1}^{n}\tau_i\cdot s^n+O^{(n-1)}(s)}u$$

where k_i and τ_i are the gain and time constants of each neuron in the layer, and $O^{(n-1)}$ (s) denotes terms that are order $(n-1)$ in s or less. Clearly this transfer function expression will always be rational and has relative degree at most one for any n. The unrestricted placement of eigenvalues is shown by the roots of the characteristic equation. For the two neuron network the roots of the characteristic are:

$$s_{1,2}=\frac{-(\tau_1+\tau_2)\pm[(\tau_1+\tau_2)^2-4\tau_1\tau_2(1-k_1k_2)]^{1/2}}{2\tau_1\tau_2} \qquad (18)$$

which can clearly have complex or real values. The characteristic equation for the three neuron network can be written as:

$$s^3+\theta_2s^2+\theta_1s+\theta_0$$

where:

$$\theta_2=(\tau_1\tau_2+\tau_2\tau_3+\tau_1\tau_3)/\tau_1\tau_2\tau_3$$

$$\theta_1=(\tau_1+\tau_2+\tau_3-k_2k_3\tau_1-k_1k_2\tau_3)/\tau_1\tau_2\tau_3$$

$$\theta_0=(1-k_2k_3-k_1k_2)/\tau_1\tau_2\tau_3$$

The arbitrary unrestricted pole placement is demonstrated by the fact that arbitrarily fixing any three parameters will realize the other three. For example setting:

$$\tau_1 = \tau_2 = k3 = 1$$

then

$$k_1 = \frac{-2+2\theta_2-2\theta_1+\theta_1\theta_2+2\theta_0-\theta_0\theta_2}{8-8\theta_2+2\theta_2^2+2\theta_1-\theta_2\theta_1+\theta_0}$$

$$k_2 = \frac{8-8\theta_2^2+2\theta_2+2\theta_1-\theta_2\theta_1+\theta_0}{(\theta_2-3)(\theta_2-2)}$$

$$\tau_3 = \frac{1}{\theta_2-2} \tag{19}$$

Similar generalizations follow for higher order systems.

2.4. *Network parameter identification*

The identification problem for this network involves the calculation of an optimal parameter vector (the values of τ_i and k_i) which minimizes the squared error between the predicted network output ($\hat{y}$) and the actual process output (y). The structure of the proposed dynamic elements precludes the use of efficient gradient descent techniques because of the presence of local minima in the solution space. Consequently, a random search technique is employed to improve the *possibility* of finding a global solution. Random search techniques have the relative advantage that they are not as easily 'trapped' in local minima as the more common gradient based methods. This is because their search directions and moves are determined randomly and thus will eventually move out of any 'well'. This improvement is achieved at a considerable computational expense in many situations. The random search procedure used here is an adaptive random search algorithm which contains modifications to improved its performance over existing random search algorithms, especially for such problems as sensitivity to initial parameter estimates, the initial search interval, the compression factor, and the random number generator (Salcedo and Azevedo, 1990). The algorithm used can be summarized by two main steps:

- Execute iteration j for P_1 trials to update the parameter vector, Θ:

$$\theta = \theta^* + Zr^{(j)} \tag{20}$$

where Θ^* is the current optimum, Z is a square matrix of dimension n consisting of random numbers between $-1/2$ and $1/2$, and $r(_j)$ is the search region for iteration j.

- After P_1 iterations (Luus and Jaakola, 1973), replace Θ^* by the best value of Θ, then contract the search region r:

$$r^{(i+1)} = r^{(i)}(1-\epsilon^{\eta}) \tag{21}$$

where ϵ is an appropriate factor. The algorithm ends after P_2 such interval reductions, which serve the purpose of increasing the probability of convergence to the global optimum.

The values of P_1, P_2 and ϵ chosen are dependent on the structure of the particular problem. From experience with the results in this paper, P_1 can range between 100 and 250 depending on the number of parameters to be identified. A good rule of thumb is that P_1 should be approximately twenty times the number of parameters to be identified. As the optimization is done off-line, P_2 is chosen large enough to ensure that an an adequate search is performed. An optimal parameter vector is normally returned within 2,000 P_2 iterations. The value of ϵ is fixed at either 0.03 or 0.05, as recommended in the original reference.

2.5. *Modified identification procedure*

The following procedure is proposed to increase the computational tractability of the random search procedure for network parameter identification. The key feature that is exploited is the fact that the first term in the Taylor series expansion for τ and k is exactly equal to the Jacobian approximation of the network function. These parameter values can be estimated from a linear identification of the process (using a low amplitude excitation). Thus, a two step procedure is proposed, where the nonlinear random search procedure is warm-started with estimates of these linear parameters:

1. The true ṗlant' is excited with small input perturbations to produce local (linear) fluctuations in the output. Linear identification starts with using an autoregressive moving average with exogenous input model (*arx* function in MATLAB) to create a discrete time transfer function. This discrete transfer function was then converted to a continuous time transfer function (*d2cm* function in MATLAB). The order of this resulting transfer function determines the number of dynamic neurons in the network.
2. The linear parameters are then used as an initial guess (nonlinear terms set to zero) to ẁarm start' the random search procedure. After the initial P_2 iterations, if the squared error is sufficiently small, the current solution vector can be used as the initial guess for a gradient-based method to improve the accuracy. If the error is not sufficiently small, then this step is repeated with the number of P_2 iterations modified.

The purpose of the linear identification is to determine the underlying dynamic order of the process. This is a critical step as the determined dynamic order is exactly equal to the number of neurons which is to be employed in the RDNN architecture. In the results that follow, the following normalized fit metric was used to compare various models:

$$F=\left(1-\frac{\Sigma_{i=1}^{n}(y_i-\hat{y}_i)^2}{\Sigma y_i^2}\right)*100 \qquad (22)$$

This quantity ranges from minus infinity (poor model) to 100 (perfect model).

3. SIMULATION RESULTS

3.1. Case study I

In this case study we consider a first order, exothermic, irreversible reaction carried out in a well mixed stirred tank reactor (Uppal *et al.*, 1974). The fresh feed of pure reactant (A) is mixed with a perfect (undelayed) recycle stream with recycle flow rate $(1-\lambda)F$. The mass and energy balances are described by the following equations:

$$V\frac{dc_A}{dt'}=\lambda Fc_{Af}+F(1-\lambda)c_{Af}-Fc_A-Vk_0\exp\left(\frac{-E}{RT}\right)c_A \qquad (23)$$

$$V\rho C_p\frac{dT}{dt'}=\rho C_pF(\lambda T_f+(1-\lambda)T-T)$$

$$+V(-\Delta H)k_0\exp\left(\frac{-E}{RT}\right)c_A-hA(T-T_c) \qquad (24)$$

where the last term in equation (24) accounts for the heat removed from a cooling jacket maintained at temperature T_c. Using the dimensionless variables defined in Table 1, these balances can be rewritten as follows:

Table 1. Dimensionless variables for case study I

x_1	$\frac{c_{Af}-c_A}{c_{Af}}$
x_2	$\frac{T-T_f}{T_f}\left(\frac{E}{RT_f}\right)$
t	$\frac{t'F\lambda}{V}$
Da	$\frac{k_0e^{\gamma}V}{F\lambda}$
B	$\frac{(-\Delta H)c_{Af}}{{}_{\rho}C_pT_f}\left(\frac{E}{RT_f}\right)$
β	$\frac{hA}{F\lambda_{\rho}C_p}$
γ	E/RT_f
u	$\frac{T_c-T_f}{T_f}\left(\frac{E}{RT_f}\right)$

$$\frac{dx_1}{dt}=-x_1+Da(1-x_1)\exp\left(\frac{x_2}{1+x_2/\gamma}\right) \qquad (25)$$

$$\frac{dx_2}{dt}=-x_2+BDa(1-x_2)\exp\left(\frac{x_2}{1+x_2/\gamma}\right)-\beta(x_2-u) \qquad (26)$$

where x_1 is the conversion, x_2 is the dimensionless temperature, Da is the Damköhler number, ß is the dimensionless heat transfer coefficient, γ is the dimensionless activation energy, B is the dimensionless adiabatic temperature rise, t is the dimensionless time, and u is the dimensionless cooling jacket temperature. For this case study, the following values of the system parameters were used: $B=209.2$, $Da=7.2E10$, ß$=2.092$, and $\gamma=8750.0$ The nominal operating point considered is defined by: $x_{10}=0.09341$, $x_{20}=385.0$, and $u_0=311.1$.

The identification problem consists of modeling the reactor temperature response as a function of the cooling jacket temperature. The identification procedure described in the last section was carried out using a linear second order system for step 1. The input sequence used for this linear identification is shown in Fig. 3 and the following linear transfer function was identified ($F=96.51$):

$$G(s)=\frac{0.1s+1.076}{0.054s^2+0.14s+1} \qquad (27)$$

which gave the linear parameters shown in Table 2. The Taylor series expansions for the expressions, $k(y)$ and $\tau(y)$, in equation (3) were truncated after two terms, resulting in the following network structure:

$$\dot{x}_1=\frac{-x_1}{\tau_{10}+\tau_{11}x_1}+\frac{k_{10}+k_{11}x_1}{\tau_{10}+\tau_{11}x_1}(x_2+u) \qquad (27)$$

$$\dot{x}_2=\frac{-x_2}{\tau_{20}+\tau_{21}x_2}+\frac{k_{20}+k_{21}x_2}{\tau_{20}+\tau_{21}x_2}x_1 \qquad (27)$$

$$y=x_1 \qquad (28)$$

where there are eight parameters to be identified. Using the values from the linear identification step as the initial guess for the linear terms, and zeroes for the nonlinear terms, the random search routine was implemented to determine the nonlinear parameters shown in Table 3. The parameters used for this search were $P_1=150$, $P_2=500$, and $\epsilon=0.05$. The corresponding fit for these parameters was quite good ($F=94.01$), thus it was unnecessary to use a gradient based method to improve the fit. The performance of this dynamic neural network was compared against a linear model and a traditional

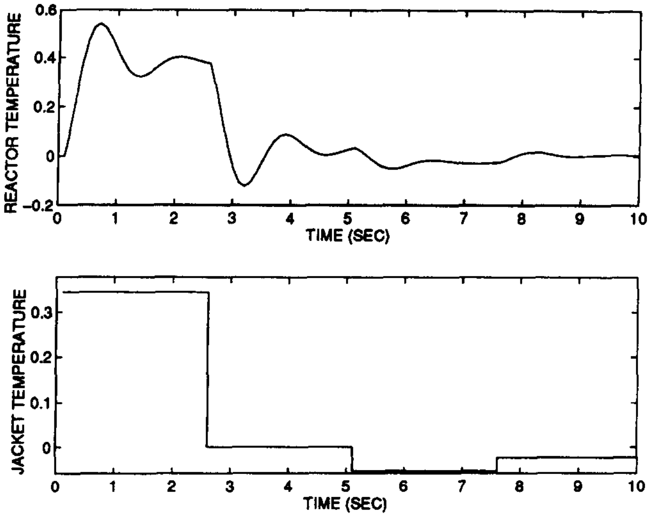

Fig. 3. Input/output data for linear identification.

neural network. The input/output set for the nonlinear identification procedure is shown in Fig. 4. The Jacobian approximation of the nonlinear model derived from this data is:

$$G(s)=\frac{0.071s+1.079}{0.046s^2+0.121s+1} \tag{29}$$

which is slightly different than the linear model identification in equation (27).

The feedforward artificial neural network had one hidden layer with five neurons. From the literature it is known that input sets which contain both auto-aggressive and moving average terms results in better FANN performance. Coupled with the knowledge that the apparent 'best-fit' linear model is a delay-free, relative degree one, second order transfer function, it was decided to employ an ARMA structure with memory 2 in both the past inputs u and past outputs y:

$$U_{train}=[u(k-2),\, u(k-1),\, y(k-2),\, y(k-1)]$$

$$Y_{train}=[y(k)] \tag{30}$$

The network was trained via standard batch back-

Table 2. Linear parameters for case study I

τ_2	0.094
τ_1	−0.124
k_1	−0.232
k_2	−5.251

Table 3. Nonlinear parameters for case study I

k_{10}	−2.33E-01
k_{11}	1.25E-03
τ_{10}	−1.17E-01
τ_{11}	1.116E-03
k_{20}	−5.25
k_{21}	6.159E-03
τ_{20}	9.039E-02
τ_{21}	1.109E-03

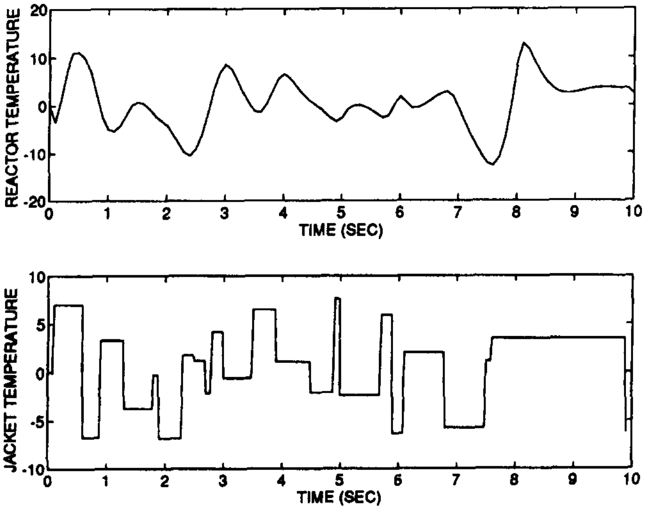

Fig. 4. Input/output data for nonlinear identification.

propagation. For validation, a recursive method was employed using backpropagation with each predicted value of the output used as an input for the next prediction.

A simple evaluation of the predictive ability of the three models is illustrated for a random input sequence in Fig. 5. The corresponding fits for the three are as follows; RDNN = 98.91, Linear = 96.89 and the FANN = 88.47. The response to +3 and −3 step changes in cooling temperature is shown in Fig. 6. As expected, the linear model cannot capture the nonlinear behavior in the system. In addition, the traditional neural network was unable to capture this behavior. This is attributed to the fact that the FANN is essentially class of ARMA (Auto-Regressive Moving Average) models, which are not known to perform well for prediction horizons of more than one-step-ahead in time (Morris *et al.*, 1994). On the other hand, the proposed dynamic neural network accurately predicted the dynamic output. For larger inputs, which correspond to incipient instability in the CSTR, none of the models were able to accurately represent the true nonlinear behavior (Fig. 7), although the dynamic neural network outperformed the other two models.

3.2. *Case study II*

The second application considers the production of cycolpentenol (B) from cyclopentadiene (A) by acid catalyzed electrophilic addition of water in dilute solution (Engell and Klatt, 1993). The unwanted by-

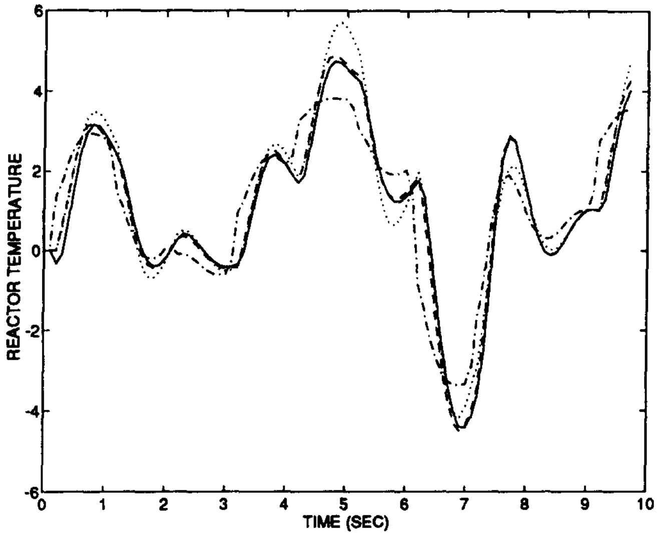

Fig. 5. Model comparison for random input sequence. Actual reactor (solid), linear (dotted), FANN (dash-dot), RDNN (dashed).

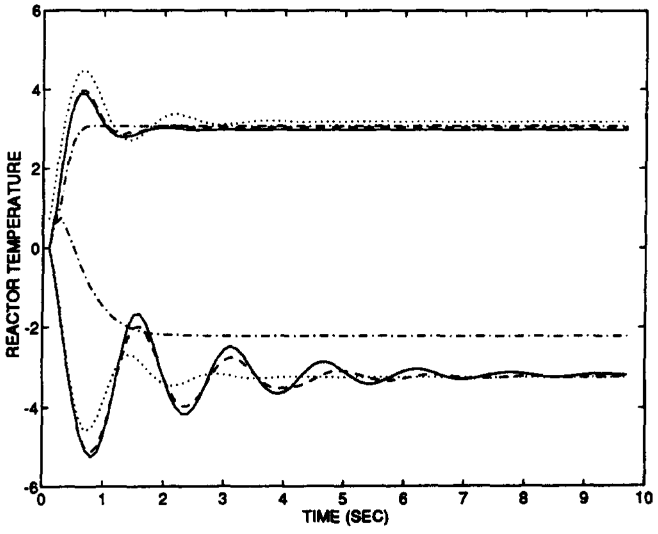

Fig. 6. Model comparison for ±3 step changes in input. Actual reactor (solid), linear (dotted), FANN (dash-dot), RDNN (dashed).

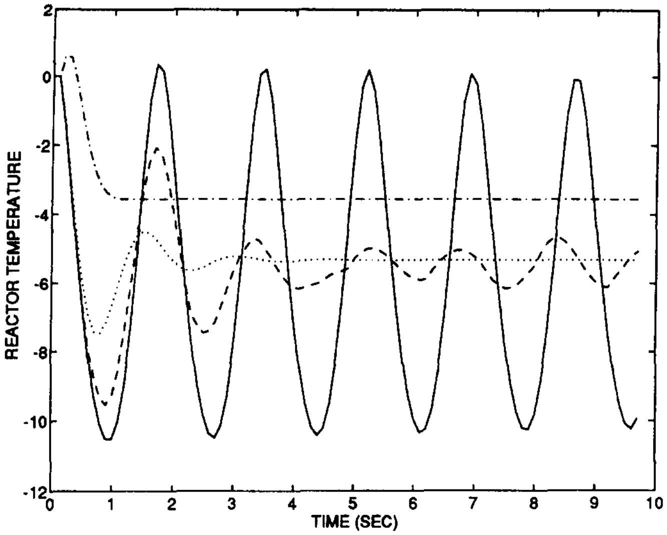

Fig. 7. Model comparison for − 5 step change in input. Actual reactor (solid), linear (dotted), FANN (dash-dot), RDNN (dashed).

products are dicyclopentadiene (D)-a side product, and cyclopentanediol (C)-a consecutive product. The reaction mechanism can be written as follows:

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

$$2A \xrightarrow{k_3} D \tag{31}$$

For constant density and ideal reactor residence time, the balance equations for the concentrations of cyclopentadiene, c_A, and cycolpentenol, c_B, are:

$$\frac{dc_A}{dt} = \frac{\dot{V}}{V_r}(c_{A0} - c_A) - k_1(T)c_A - k_3(T)c_A^2 \tag{32}$$

$$\frac{dc_B}{dt} = -\frac{\dot{V}}{V_r}c_B + k_1(T)c_A - k_2(T)c_B \tag{33}$$

where V_R is the volume of the reactor, and $\dot{V}$ is the volumetric flow through the reactor. The energy balance yields the differential equation for the temperature T in the reactor:

$$\frac{dT}{dt} = -\frac{1}{\rho C_P}(k_1(T)c_A\Delta H_R + k_2(T)c_B\Delta H_{R_{BC}} + k_3(T)c_A^2\Delta H_{R_{AD}}) \tag{33}$$

$$+\frac{\dot{V}}{V_r}(T_0 - T) + \frac{k_W A_R}{\rho C_P V_R}(T_K - T) \tag{34}$$

where ΔH_{Rij} are the various reaction enthalpies, T_0 is the temperature of the inlet stream, T_K is the temperature of the reactor coolant, k_W is the heat transfer coefficient, and A_R the surface area of the cooling jacket. The rate coefficient k_1, k_2 and k_3 depend on the reaction temperature via an Arrhenius expression:

$$k(T) = k_0 \cdot \exp\frac{-E_A}{R(T + 273.15)} \tag{35}$$

The system parameters for this study are found in Table 4 and the standard operating conditions are in Table 5. Following the treatment in the original reference (Engell and Klatt, 1993), the coolant dynamics will be neglected, yielding a third order nonlinear dynamical system.

Following the identification procedure which was outlined earlier, a linear model was identified from a low amplitude input signal. These results indicated that the third order approximation (Fig. 8) was only slightly better than the second order approximation. This implies that the dynamic neural network structure should contain at most three dynamic neurons, but that a two neuron

Table 4. System parameters for case study II

$k_{0_{ab}}$	1.287E12 h^{-1}
$k_{0_{bc}}$	1.287E12 h^{-1}
$k_{0_{ad}}$	9.043E9 (mol.h)$^{-1}$
$E_{A_{ab}}$	9758.3 K
$E_{A_{bc}}$	9758.3 K
$E_{A_{ad}}$	8560 K
V_r	10 l
ρ	0.9342 $\frac{\text{kg}}{\text{l}}$
C_P	4.01 $\frac{\text{kJ}}{\text{kg.K}}$
$H_{R_{AB}}$	4.2 $\frac{\text{kJ}}{\text{molA}}$
$H_{R_{BC}}$	− 11.0 $\frac{\text{kJ}}{\text{molB}}$
$H_{R_{AD}}$	− 41.85 $\frac{\text{kJ}}{\text{molA}}$
Qks	− 4496 $\frac{\text{kJ}}{\text{h}}$

structure may be able to properly represent the system. The second order transfer function from this linear identification procedure is found to be:

$$G(s)=\frac{-0.1805s+57.6514}{s^2+69.8s+4385.2} \qquad (36)$$

and the third order model is:

$$G(s)=\frac{-0.3776s^2-51.6338s+41656.0}{s^3+560.2s^2+52207.2s+2553000} \qquad (37)$$

Table 5. Nominal operating point for case study II

c_{A0_s}	$5.1\,\frac{mol}{l}$
c_{A_s}	$1.235\,\frac{mol}{l}$
c_{B_s}	$0.9\,\frac{mol}{l}$
T_S	134.14°C
T_0	130°C
$k_{1,2}$	$50.6\ h^{-1}$
k_3	$6.74\,\frac{l}{mol\cdot h}$
$\frac{\dot{V}}{V_R}$	$18.83\ h^{-1}$
C_P	$4.01\,\frac{kJ}{kg.K}$
$H_{R_{AB}}$	$4.2\,\frac{kJ}{molA}$
$H_{R_{BC}}$	$-11.0\,\frac{kJ}{molB}$
$H_{R_{AD}}$	$-41.85\,\frac{kJ}{molA}$
Qks	$-4496\,\frac{kJ}{hr}$

Based on the observations from the linear identification step, the second order structure was first studied. Following the procedure employed in the previous case study, the linear parameters were found (Table 6). The random search procedure was then employed to identify the nonlinear structure given by the parameters in Table 7. The parameters used for this search were $P_1=150$, $P_2=1{,}500$, and $\epsilon=0.05$.

The performance of the RDNN for this system was compared to a linear model and a traditional FANN. The data used for the linear model estimation and traditional FANN training was larger in magnitude than that used for the linear fluctuations in Step 1. The FANN structure was similar to that used in the previous case study, and the linear model employed was a third order transfer function which was found to be:

$$G(s)=\frac{-0.8s^2+78.25s+7595.6}{s^3+210.0s^2+13520.0s+433880.0} \qquad (38)$$

As before, the models were evaluated with respect to their predictive ability for various sequences of input flow rate. For the random input sequence (Fig. 9), all the models performed relatively well in predicting the process output; Linear = 91.51, RDNN = 92.65, FANN = 70.00. In Fig. 10 are shown the responses for the ±6 (mol/l) input step change in input. For the −6 (mol/l) step the RDNN model yielded the highest accuracy, whereas for the +6 (mol/l) step the RDNN and the linear model performed equally well, while the traditional FANN was less accurate.

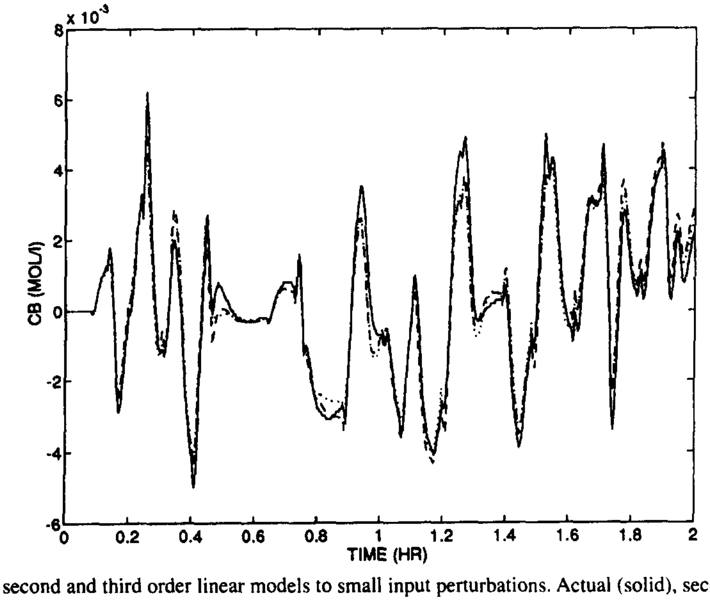

Fig. 8. Comparison of second and third order linear models to small input perturbations. Actual (solid), second order (dotted), third order (dashed).

Table 6. Linear parameters for (two neuron) case study II

τ_2	−3.129E-03
τ_1	2.568E-03
k_1	−4.633E-04
k_2	−2.234E03

Table 7. Nonlinear parameters for (two neuron) case study II

k_{10}	−4.6330E-4
k_{11}	−3.6293E-5
τ_{10}	2.5682E-3
τ_{11}	3.4488E-4
k_{20}	−2.234E+3
k_{21}	9.5473E-4
τ_{20}	−3.1290E-3
τ_{21}	1.4639E-6

The procedure was repeated for the case of a third order RDNN model. The low-amplitude training set in Fig. 8 was used to identify the six linear parameters in Table 8. The fit for these linear parameters for the linear training data was reasonable (F=89.90). These values were used to warm start the random search procedure to identify a nonlinear model based upon the training set shown in Fig. 11. The parameters used for this search were P_1=200, P_2=3000 and ϵ=0.05. The best solution obtained from the random search procedure (F=92.05) is shown in Table 9. For validation, the model was subjected to ±6 (mol/l) step changes in the input (Fig. 12). It can be seen that the performance of the three neuron model does a better job of capturing the overshoot behavior than to the two neuron model.

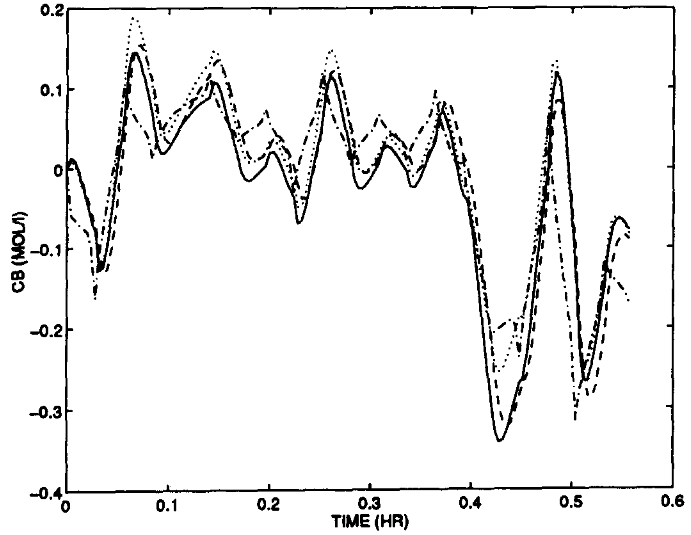

Fig. 9. Model comparison for random input sequence. Actual reactor (solid), linear (dotted), FANN (dash-dot), 2 neuron RDNN (dashed).

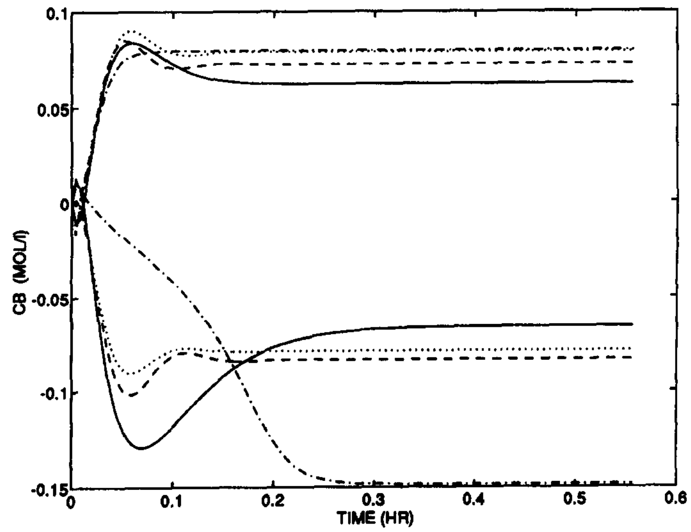

Fig. 10. Model comparison for ±6 (mol/l) step changes in input. Actual reactor (solid), linear (dotted), FANN (dash-dot), 2 neuron RDNN (dashed).

Table 8. Linear parameters (three neuron) for case study II

k_{10}	− 8.9153E-04
τ_{10}	2.3612E-03
k_{20}	− 8.289E-02
τ_{20}	− 2.9915E-03
k_{30}	− 3.61129E-04
τ_{30}	2.123E-03

Table 9. Nonlinear parameters (three neuron) for case study II

k_{10}	− 8.94000E-04
k_{11}	1.4336E-04
τ_{10}	2.35772E-03
τ_{11}	2.46439E-03
k_{20}	− 8.289E-02
k_{21}	8.76978E-04
τ_{20}	− 2.99372E-03
τ_{21}	3.48682E-06
k_{30}	− 3.61923E-04
k_{31}	− 5.40545E-04
τ_{30}	2.13113E-03
τ_{31}	1.26467E-03

4. MODEL-BASED CONTROLLER DESIGN

The biologically motivated dynamic network (RDNN) model proposed in this paper can be implemented in model-based control schemes such as Internal Model Control (IMC) or Model Predictive Control (MPC). These schemes normally rely on a model inverse for control move computations and recent results for Volterra-series-based models (Doyle *et al.*, 1995) describe a method for constructing a nonlinear model inverse which only requires linear model inversion. The control structure is displayed in Fig. 13 and is composed of two parts:

1. the dynamic model (RDNN) which contributes to a feedback signal representing the difference between the true process and the modeled output; and
2. a model inverse loop which contains the RDNN

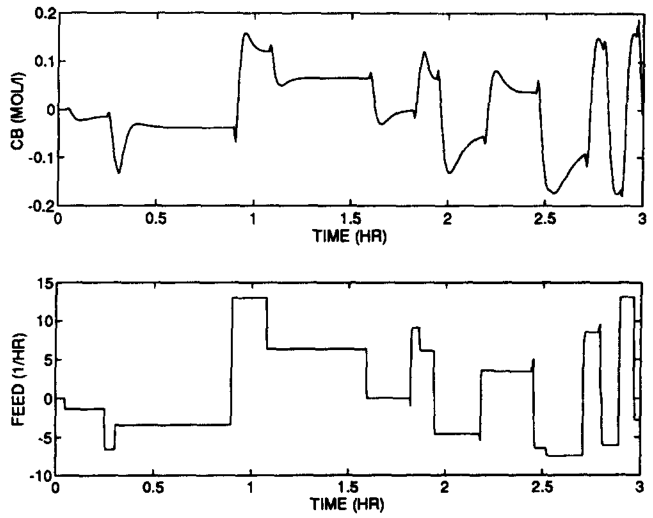

Fig. 11. Nonlinear training set.

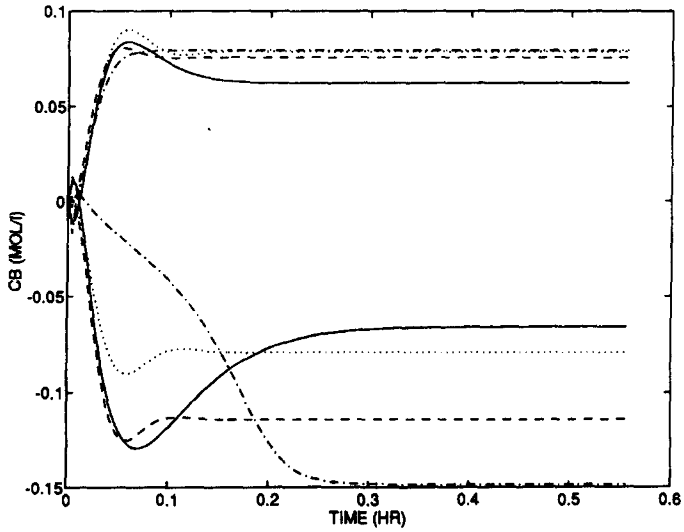

Fig. 12. Asymmetric Model response for ±6 (mol/l) step changes in input. Actual reactor (solid), linear (dotted), 3 neuron RDNN (dashed).

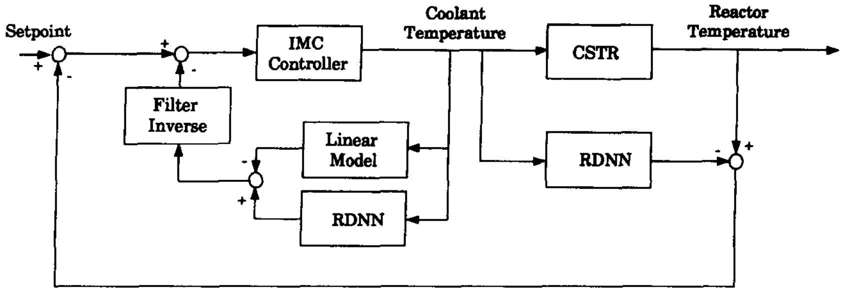

Fig. 13. Closed-loop control structure.

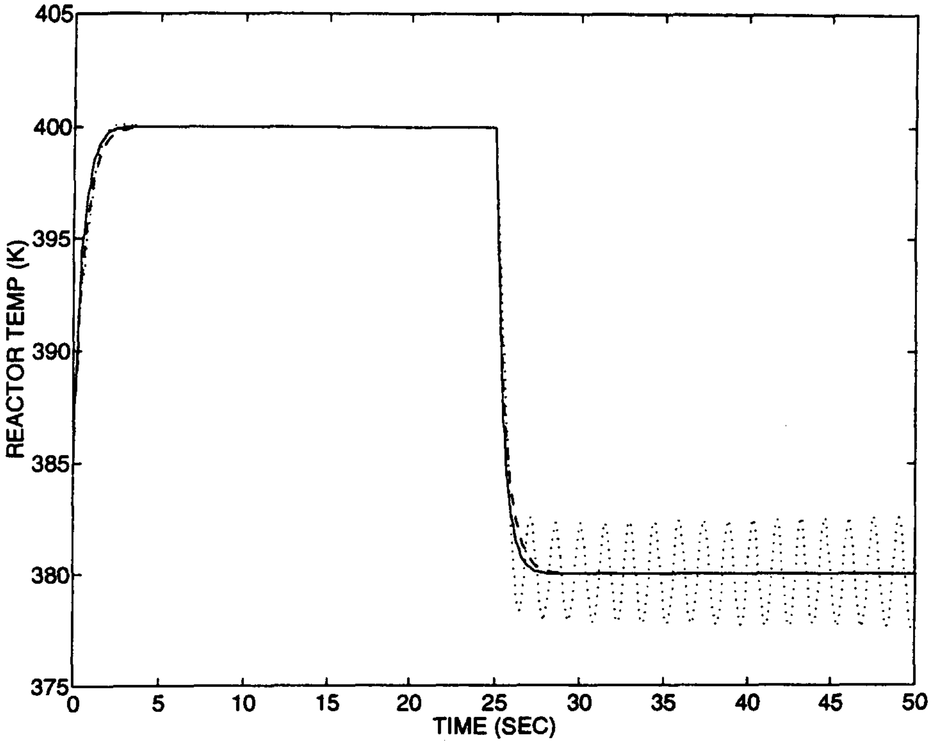

Fig. 14. Closed-loop setpoint response. Reference trajectory (solid), RDNN controller (dotted), linear controller (dashed).

model, a linear approximation to the RDNN model and a linear IMC controller.

Simulations were carried out for two control schemes: (i) a standard *linear* IMC controller which utilizes a linear model and its inverse; and (ii) the nonlinear controller depicted in Fig. 13. In both cases, the IMC filter time constant is equal to 0.5 sec. The closed-loop response for a step change in the setpoint from 385 to 400 K at $t=0$ and back down to 380 K at $t=25$ is shown in Fig. 14. The dashed line represents the response of the linear controller, the dotted line represents the response of the nonlinear controller, and the solid line the ideal reference trajectory that would be achieved with perfect control. The nonlinear controller achieves vastly superior trajectory following than the linear controller which is unstable for the lower setpoint change. This demonstrates the improved performance that can be attained with a more accurate nonlinear model, such as RDNN.

5. CONCLUSIONS

In this paper, a simple dynamic element was presented for use in network architectures to model process systems. This dynamic neuron is motivated by biology, and was shown to be capable of capturing a rich range of nonlinear dynamic behavior. A formal identification procedure was outlined for this model, which consisted of three steps: (i) linear (low amplitude input) identification for initialization purposes, (ii) random search identification for the nonlinear parameters, and (iii) a gradient descent search to find the local minimum. In addition, it was shown that the proposed architecture can be readily implemented in a standard model-based control methodology (IMC) to yield a nonlinear model based controller. Two process case studies were presented to illustrate the superior performance of the proposed approach over both simple linear models and traditional feedforward artificial neural networks.

Future research directions will include the application of gradient descent algorithms to train the network models. In addition, more complex process applications are currently under investigation and synthesis of input/output feedback linearizing controllers are being formulated.

Acknowledgements—We gratefully acknowledge funding from the National Science Foundation (CTS 9257059 and BCS 9315738). We would also like to acknowledge the collaboration with the DuPont Company, notably Tunde Ogunnaike.

REFERENCES

Abbott L. and LeMasson G., Analysis of Neuron Models with Dynamically Regulated Conductances, Neural Computation, **5**, (1993) 823–842

Beer R., On the Dynamics of a Continuous Hopfield Neuron with Self-Connection, *Technical Report Technical Report CES*-94-1, Case Western Reserve University, (1994).

Chappelier J. and Grumbach A., Time in Neural Networks, SIGART Bulletin, **5**, (1994) 3–11

Chien I.-L. and Ogunnaike B., Modeling and Control of High-Purity Distillation Columns, AIChE national meeting, (1992).

Doyle F., Ogunnaike B. and Pearson R., Nonlinear model-based control using second order volterra models, Automatica, **31**, (1995) 697–714

Engell S. and Klatt K.-U., Nonlinear Control of a Non-Minimum-Phase CSTR, *Proceedings of American Control Conference*, San Francisco, 2941–2945, (1993).

Funahashi K.-I. and Nakamura Y., Approximation of Dynamical Systems by Continuous Time Recurrent Neural Networks, Neural Networks, **6**, (1993) 801–806

Henson M., Ogunnaike B. and Schwaber J., The Baroreceptor Reflex: A Biological Control System with Applications in Chemical Process Control, IEC Research, **33**, (1994) 2453–2466

Hodgkin A.L. and Huxley A., A Quantitative Description of Membrane Current and its Application to Conduction and Excitation in Nerve, J. Physiol., **117**, (1952) 500–544

Hopfield J.J. and Tank D., Computing with Neural Circuits: A Model, Science, **233**, (1986) 625–633

Luus R. and Jaakola T.I., Optimization by Direct Search and Systematic Reduction of the Size of Search Region, AIChE J., **19**, (1973) 760–766

MacGregor J., Marlin T., Kresta J. and Skagerberg B., Multivariate Statistical Methods in Process Analysis and Control, *Chemical Process Control-CPCIV*, AICHE, 79–99, (1991).

Montague G., Tham M. and Morris A., Predictive Control of Distillation Columns using Dynamic Neural Networks, *Technical report*, University of Newcastle-upon-Tyne, (1991).

Morris A., Montague G. and Willis M., Artificial Neural Networks: Studies in Process Modeling and Control, Trans IChemE, **72**, (1994) 3–19

Narenda K. and Parthasarathy K., Identification and Control of Dynamical Systems Using Neural Networks, IEEE Trans. on Neural Networks, **1**, 1 (1990) 1–25

Pollard J. F., Garcia C. E. and Ramaker B., Total Process Control-Beyond the Design of Model Predictive Control, *Chemical Process Control-CPCIV*, AICHE, 335–361, (1991).

Salcedo R.M.J.G. and Azevedo S.F.D., An Improved Random-Search Algorithm for Nonlinear Optimization, Comp. Chem. Engng., **14**, 10 (1990) 1111–1126

Rao D. and Gupta M., Dynamic Neural Units and Function Approximation, *IEEE Int. Conf. on Neural Networks*, San Francisco, 743–748, (1993).

Sastry P., Santharam G. and Unnikrishan K., Memory Neuron Networks for Identification and Control of Dynamical Systems, *Technical Report GMR*-7916, General Motors Research Laboratories, (1993).

Scott G. and Ray W., Neural Networks Process Models Based on Linear Model Structure, Neural Computation, **6**, (1994) 718–738

Stark L., Neural nets, Random design and reverse engineering, *IEEE Int. Conf. on Neural Networks*, San Francisco, 1313–1320, (1993).

Uppal A., Ray W. and Poore A., On the Dynamic Behavior of Continuous Stirred Tank Reactors, Chem. Eng. Sci., **29**, (1974) 967–985

Zurada J., *Artificial Neural Systems*, 1st edn, West Publishing Company, (1992).