# Dynamic Optimization of Nonlinear Processes by Combining Neural Net Model with UDMC

**Qi Chen**
Dept. of Chemical Engineering, University of Maryland, College Park, MD 20742

**William A. Weigand**
Center for Biotechnology Manufacturing, Maryland Biotechnology Institute,
Dept. of Chemical Engineering, University of Maryland, College Park, MD 20742

*A new dynamic optimization technique presented combines a neural network model with a universal dynamic matrix control (UDMC) algorithm. This technique utilizes a nonlinear-model-predictive control technique for on-line optimization and feedback control by using a dynamic neural net model. This approach offers two important advantages over conventional UDMC. One is that a dynamic neural net model can be developed from process data and used for optimization calculations, thus achieving optimization without a first principle model. This neural-network-based optimization approach also produces good performance even with process-model mismatch. The other is that our neural-net-model-based UDMC algorithm greatly reduces the computation time required for the nonlinear dynamic matrix used for the successive quadratic programming algorithm. The development of this technique also involved an analysis of the effect of network structure on dynamic optimization. A state-space-based neural network model which utilizes a priori process knowledge is best suited for optimization calculations. Advantages of this technique are illustrated by simulation for two chemical processes.*

## Introduction

With the development of sophisticated methods for nonlinear programming and powerful computer hardware, on-line dynamic optimization problems based on rigorous physical models which involve ordinary differential equations or differential/algebraic equations have been studied by many researchers. Both Cuthrell and Biegler (1987) and Renfro et al. (1987) proposed a very similar approach to solve dynamic optimization problems, and many case studies of chemical engineering applications have been presented. Morshedi (1986) proposed a universal dynamic matrix control (UDMC) algorithm by using a finite dimensional manipulated input vector. This replacement transforms the two-point boundary value problem into an ordinary nonlinear programming problem which can be solved with nonlinear optimization algorithms. Also, the advantages of the feedback control contained in the UDMC algorithm can be used for rejecting unmeasurable disturbances during the course of optimization. This can be implemented easily because feedback control with DMC is well known and established. Several application studies can be found in the literatures. Lin (1987) presented an application of UDMC for dynamic optimization of a biochemical process. Lewis (1989) applied the UDMC algorithm for dynamic optimizing control of a CSTR (continuous stirred tank reactor) with an economic objective. Also, Eaton and Rawlings (1990) further developed a nonlinear model predictive control (MPC) technique to realize the feedback optimal control of nonlinear processes. A detailed review which summarizes many of the relevant issues in nonlinear-model-based control can be found in a recent article by Bequette (1991). However, there still exist several important difficulties with the on-line optimization of chemical processes by using UDMC. One is the time-consuming computation problem required for the on-line application of dynamic optimization (Rawlings and Biegler, 1992); the second is the need to obtain reasonable process models required for dynamic optimization; the third is the need to simultaneously consider an economic optimization trajectory and process con-

Correspondence concerning this article should be addressed to W. A. Weigand.

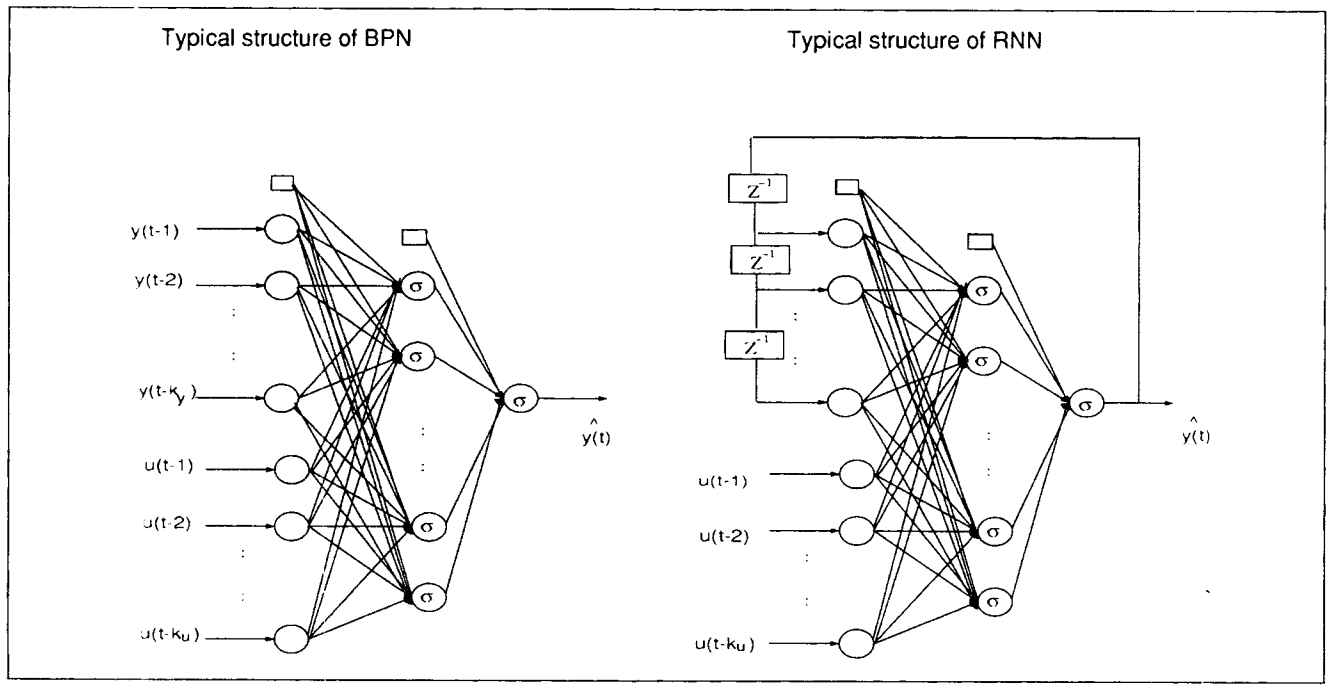


**Figure 1. Typical structure of BPN and RNN model.**

trol, both of which are needed to realize optimal performance with a chemical process.

Recently, the application of neural networks for modeling and control has been considered. Bhat and McAvoy (1990), Chen et al. (1990), and Narenda and Parthasarathy (1990) have illustrated that neural networks can be used effectively for the identification and control of nonlinear systems. Nguyen and Widrows (1990) proposed a neural network self-learning control system by using the network's adaptive ability. Saint-Donat et al. (1992) and Psichogios and Ungar (1992) have proposed neural network model-predictive control and illustrated its application to chemical processes. In this article the main objective is dynamic optimization rather than only control. In order to use the model predictive control technique to achieve the objective of dynamic optimization, the structure of the neural net model and the difference between batch processes and continuous processes are investigated in detail.

There are two special features of neural-network-model-based UDMC which are addressed in this article. One is a computation method for the nonlinear dynamic matrix based on a neural network, which results in a large reduction of the computation time for dynamic optimization. This time reduction can be achieved since a neural net model is actually an algebraic equation, which in turn leads to the algebraic computation of the dynamic matrix. The other feature is illustrated by simulation studies of two typical nonlinear chemical processes which are accomplished without *a priori* models. In this case a neural net model is trained with input/output data obtained from an input signal added to the process. These simulation results illustrate that the proposed technique has great potential to realize on-line optimization in industry, where accurate models are often not available.

## Dynamic Neural Network Model

The neural network approach is capable of modeling any continuous nonlinear function through supervisory learning, provided that the number of hidden units is sufficient and all the activation functions of the hidden units are continuously differentiable (Hornik et al., 1989, 1990). It has also been demonstrated that neural networks can be used for dynamic nonlinear modeling (Bhat and McAvoy, 1990; Narenda and Parthasarathy, 1990; Chen et al., 1990) for model predictive control (Saint-Donat et al., 1992; Psichogios and Ungar, 1991; Kolvisto et al., 1991), adaptive control (Nguyen and Widrows, 1990) and adaptive steady-state optimization (Chen and Weigand, 1992).

Backpropagation neural networks (BPN) and recurrent neural networks (RNN) are the most attractive for dynamic process modeling (Bhat and McAvoy, 1990; Narenda and Parthasarathy, 1990; Su and McAvoy, 1991). Typical BPN and RNN structures are shown in Figure 1. In most dynamic neural modeling, a nonlinear autoregressive with exogenous (NARX) input is used to build the model. This procedure uses a number of past plant inputs and outputs to predict the future process outputs:

$$y(k+1)=f_n[y(k), \ldots, y(k-n_y), u(k), \ldots, u(k-n_u), w] \quad (1)$$

The difference between BPN and RNN is that RNN uses the predicted output as part of the input for the next iteration of prediction. The training delta rule is identical and the calculation using a three layer neural network is similar, except that the predicted output values are used for the input rather than past outputs of the process. For either the BPN or the RNN, we have the following set of equations:

$$\text{Hidden layer:} \quad H_j=f\left(\sum_{i=1}^{n_{\text{inp}}+1} w_{ij}U_i\right), \quad 1\le j\le n_{\text{hid}} \quad (2)$$

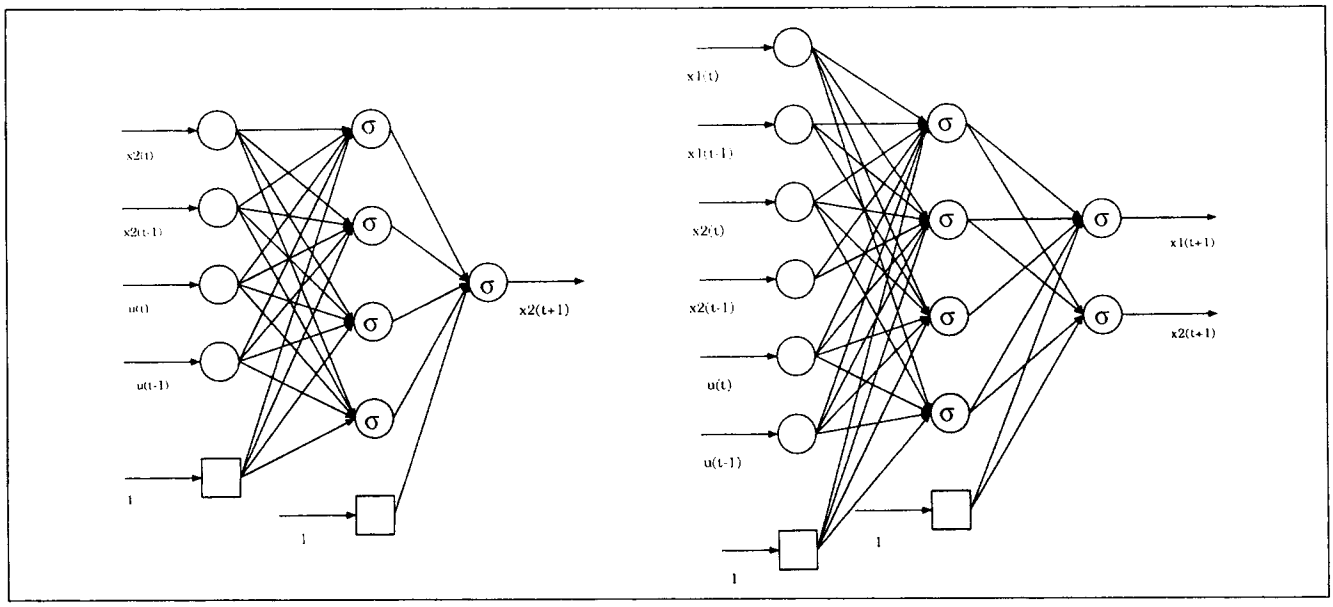


**Figure 2. Input/output neural net model and state-space neural net model for semibatch chemical reactor.**

$$\text{Output layer:}\quad Y_o = f\left(\sum_{j=1}^{n_{\text{hid}}+1} w_{jk} H_j\right), \quad 1 \le k \le n_{\text{out}} \qquad (3)$$

where $U_i$ is the scaled input vector, $H_j$ is the output of the neurons of the hidden layer, $Y_o$ is the output vector of the output layer, and $n_{\text{inp}}$, $n_{\text{hid}}$, and $n_{\text{out}}$ are the numbers of neurons in the input, hidden, and output layers, respectively. Also, $f$ is the nonlinear activation function of each neuron which is usually a sigmoid. The choice of using either the backpropagation or recurrent neural network models for dynamic optimization depends on the processes to be optimized. For example, in order to use the model predictive control technique for a batch process, the prediction horizon is often chosen as the same as the control horizon, thus only a one-step ahead prediction will be needed for the neural network model. Therefore, a BPN model will be much better to use for the purpose of optimization. However, for a continuous process, normally, the prediction horizon is greater than the control horizon, and therefore a RNN model is better to use for the purpose of multiple steps ahead prediction. The details will be discussed in the simulation studies later. In addition, most neural network models are based on input-output data such as for a black box model. In this article, a state-space realization of the neural network model is used for on-line optimization purposes. This approach possesses several advantages:

(a) The model includes prior information in a natural way even when the more detailed knowledge necessary for a first principles model is not available.

(b) The state variables generally represent physical quantities and nonlinear relations are easily expressed.

(c) Constraints on any process variables can be included in the model.

The diagrams of the input/output neural net model and state-space model are shown in Figure 2. A detailed comparison of input/output neural network models with state-space neural network models will be discussed in the section on Chemical Process Applications.

## UDMC and Nonlinear Dynamic Matrix of Neural Net Model

UDMC is an extended algorithm of DMC for nonlinear systems. In the absence of linearity, the optimal control vector cannot be calculated by using the dynamic matrix and superposition. The optimal control vector must be recalculated during each interval. This recalculation uses the Jacobian that quantifies the effect of a control variable change with respect to the response variable for each interval. The computation of the Jacobian of a dynamic system described by a set of nonlinear, first-order ordinary differential equations was discussed by many researchers (Lin, 1987; Morshedi et al., 1986; Lewis, 1988). Now let us briefly review the dynamic matrix of the UDMC algorithm. Suppose the input variable, $u$, can be completely characterized as a vector where each element, $u_i$, represents the value of the input at the $i$th interval. The Jacobian for a time horizon can be described as a dynamic matrix for a single-input single-state system, that is:

$$J = \begin{bmatrix} \frac{\partial x_1}{\partial u_0} & 0 & \cdots & \cdots & \cdots & 0 \\ \frac{\partial x_2}{\partial u_0} & \frac{\partial x_2}{\partial u_1} & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & 0 & \vdots & 0 \\ \vdots & \vdots & \frac{\partial x_p}{\partial u_l} & \vdots & 0 & 0 \\ \frac{\partial x_P}{\partial u_0} & \cdots & \cdots & \cdots & \cdots & \frac{\partial x_P}{\partial u_L} \end{bmatrix} \qquad \begin{matrix} i = 1, \ldots, P \\ j = 1, \ldots, L \end{matrix}$$

Here the variable $\partial x_i/\partial u_j$ denotes the response of the state variable during the $i$th time interval due to a change in the control variable during the $j$th time period. The dynamic ma-

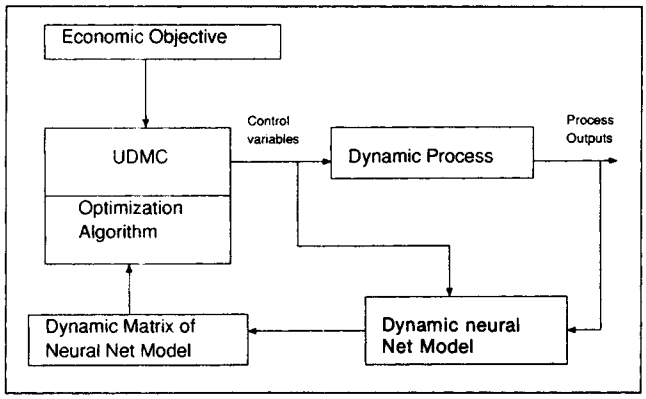


**Figure 3. Framework of neural-net-model-based dynamic optimization.**

trix is the Jacobian of the state variables, and so it is named the dynamic matrix to distinguish it from the usual Jacobian used by the search algorithm to determine the optimal control parameters. Note that $\partial x_j/\partial u_i = 0$ for all $i \le j$. For MIMO (multiple input/multiple output) systems, the dynamic matrix can be defined by the matrix of partial derivatives:

$$A = [A_{km}] = \left[\frac{\partial x_k(t_i)}{\partial u_m(t_j)}\right] \quad \begin{matrix} i = 1, \ldots, P \\ j = 1, \ldots, L \\ k = 1, \ldots, K \\ m = 1, \ldots, M \end{matrix}$$

It can also be partitioned into $K$ by $M$ dynamic submatrices for $K$ state variables and $M$ manipulated variables as shown by matrix $A$:

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1M} \\ A_{21} & A_{22} & \cdots & A_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ A_{K1} & A_{K2} & \cdots & A_{KM} \end{bmatrix}$$

Note the prediction horizon $P$ is usually different from the control horizon $L$ (Lewis, 1989).

The significant part of calculation time required for the dynamic matrix becomes the main bottleneck to on-line application of UDMC. In this article, a neural-network-model-based UDMC algorithm is proposed to realize the optimization of dynamic processes, as shown in Figure 3. The Jacobian matrix based on a neural net model has a convenient structure to apply to this algorithm. A simple representation of a state-space model in terms of a dynamic neural network model is shown in Figure 4. This structure is useful for the computation of gradient information. For a more general structure which is convenient for analysis of the neural net dynamic model, we define:

$$w1(z) = [1, z^{-1}, z^{-2}, \ldots, z^{-n_u}] \tag{4}$$

$$w2(z) = [z^{-1}, z^{-2}, \ldots, z^{-n_x}] \tag{5}$$

Therefore, the state vector, $x$, can be represented as:

$$x = f_n[\theta_1, \theta_2] \tag{6}$$

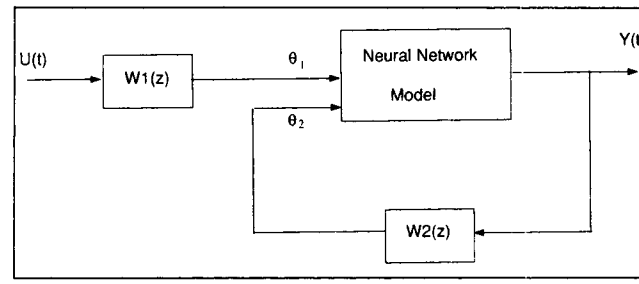


**Figure 4. Simple structure of dynamic neural network model for computation of gradient information.**

where $\Theta = [\theta_1, \theta_2]$ is a vector which represents the input information of the neural network model, that is, the past and current inputs and past state variables of the process. Therefore, the partial derivatives of the state variables with respect to the input can be calculated as:

$$\frac{\partial x}{\partial u} = \left[\frac{\partial f_n}{\partial \theta_1}\right]\frac{\partial \theta_1}{\partial u} + \left[\frac{\partial f_n}{\partial \theta_2}\right]\frac{\partial \theta_2}{\partial u} \tag{7}$$

then,

$$\frac{\partial x}{\partial u} = \frac{\partial f_n}{\partial \theta_1} w1(z) + \frac{\partial f_n}{\partial \theta_2}\frac{\partial x}{\partial u} w2(z) \tag{8}$$

and the $\partial f_n/\partial\theta_1$, $\partial f_n/\partial\theta_2$ are the Jacobian matrices of the back-propagation neural network model, which can be calculated by the current inputs values of neural net at the current interval $k$ (Chen and Weigand, 1992). Therefore, for a given operating point, the static Jacobian matrices of the process can be obtained by invoking the final value theorem for $z$-transforms, or letting $z \to 1$ in Eq. 8 or

$$\frac{\partial x}{\partial u} = \frac{\left[\frac{\partial f_n}{\partial \theta_1}\right]}{\left[1 - \frac{\partial f_n}{\partial \theta_2}\right]} \tag{9}$$

It is shown that this calculation is actually an algebraic computation which will significantly reduce the computation time for the dynamic matrix.

## Neural-Network-Model-Based UDMC

The neural net model based optimal control problem to be solved may be stated as:

$$\max_{u(k), \ldots, u(k+L-1)} \Phi[x(k), \ldots, x(k+L-1), u(k), \ldots, u(k+L-1)] \tag{10}$$

subject to

$$x(k+1) = f_n[x(k), \ldots, x(k_{nx}), u(k), \ldots, u(k_{nu}), W] \tag{11}$$

$$y_m = g(x) \tag{12}$$

$$h[x(k), \ldots, x(k-n_y), u(k), \ldots, u(k-n_u)] \leq 0 \tag{13}$$

$$u(k-1) - \Delta u_{\max} \leq u(k) \leq u(k-1) + \Delta u_{\max} \tag{14}$$

$$u(k) = u(i+L-1) \quad \text{for all} \quad k > i+L-1 \tag{15}$$

$$u_{\min} \leq u(k) \leq u_{\max} \tag{16}$$

$$x_{\min} \leq x(k) \leq x_{\max} \tag{17}$$

$$y_{\min} \leq y(k) \leq y_{\max} \tag{18}$$

where the objective function is an economic function of the manipulated and state variables of the process and $k$ is the current sampling instant. Further, $u$ is the manipulated variable, $x$ is the vector of state variables, $y_m$ is the model output, $L$ is the number of future manipulated variable moves to be optimized, and $P$ is the prediction horizon. The predicted output, $\hat{y}$, is given by:

$$\hat{y}(k+l) = y_m(k+l) + d(k) \quad \text{for} \quad l = 1, 2, \ldots, P \tag{19}$$

where, $d(k) = y(k) - y_m(k)$, is the disturbance vector and is assumed to be constant over the prediction horizon, and $y(k)$ is the measurement of the plant. In the method in this article we consider the state variables to be measurable.

To obtain the optimum of a general economic objective function, $\Phi$, the gradient of the objective function can be easily obtained by the following equation and the dynamic matrix mentioned earlier:

$$\frac{d\Phi}{du} = \frac{\partial \Phi}{\partial x}\frac{\partial x}{\partial u} + \frac{\partial \Phi}{\partial u} \tag{20}$$

The optimization decision variables are control actions $L$ steps into the future. Although the optimization is based on a control horizon, only the first control action is implemented. After the first control action is implemented, plant output measurements are obtained and compensation for plant/model mismatch is performed. The optimization calculation is then performed again. Note that, we assumed that all of the state variables of the processes can be measured in our study. In practice, state estimation techniques, such as Kalman filtering (Gattu and Zafirou, 1992) and horizon-based estimation techniques, can be applied when measurements are not available.

The procedure for implementation of the neural network model based UDMC algorithm is summarized as follows:

*Step 1.* A designed signal pseudorandom binary sequence (or PRBS for a continuous process or sinusoidal for a batch process) is input to the open-loop process in order to generate the training data for the neural network model. We assume that the process is open-loop stable. For an open-loop unstable process, a predictive controller based on an open-loop observer might be used to stabilize the process (Sistu and Bequette, 1992).

*Step 2.* The trained neural net model is tested to determine if it represents the process dynamics accurately. If the network model is satisfactory then go to step 3; if not, then go to step 1 and generate additional data for continued training of the model.

*Step 3.* The trained dynamic neural network model is used to obtain the predicted process output in order to evaluate the objective function. It is also used to calculate the dynamic matrix which is required by the successive quadratic programming (SQP) algorithm.

*Step 4.* The feasible successive quadratic programming (FSQP) algorithm (Zhou and Tits, 1990) is used to solve for the optimal control profile for an economic objective function.

*Step 5.* The first control action is applied to the process and continued until the next sample time.

*Step 6.* The current state variables are measured, the new value of the objective is calculated and combined with model/process mismatch information (Eq. 19) and used for the next optimization calculation. Then, steps 3 and 5 are repeated.

*Step 7.* For the optimization of a batch process, steps 3 to 5 are repeated until the final step is reached; for continuous process, steps 3 to 6 are repeated in accordance with the control horizon and prediction horizon.

## Chemical Process Applications

The following two examples are provided to illustrate several features of the approach presented in this article. The first example demonstrates the computation of the optimal economic temperature profile for a batch chemical reactor. The second example presented is a dynamic optimization of a continuous biochemical reactor. In both examples, the objective is related to process economics since the desired product is optimized.

### *Example 1: final time optimization of batch reactor*

The example used here is from Ray (1981) and it has also been used by Eaton and Rawlings (1990). The reaction is the exothermic formation of $B$ from $A$ and is carried out in a batch reactor as shown in Figure 5. The objective of this process is to maximize the product concentration at the end of a given reaction time by manipulating the coolant flow rate. The dimensionless dynamic model was given by Eaton and Rawlings (1990) as follows:

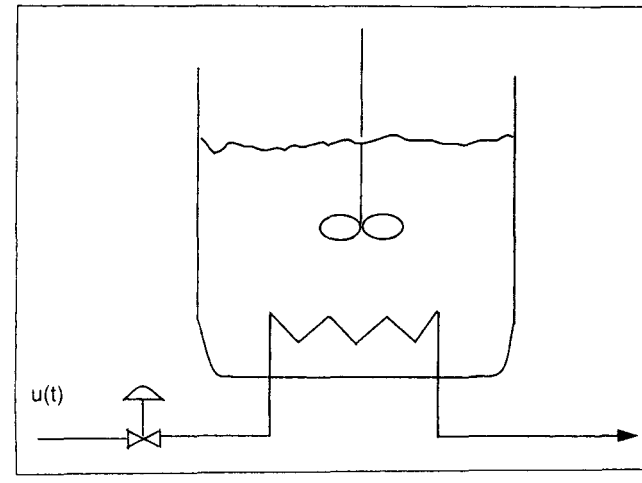


**Figure 5. Batch chemical reactor.**

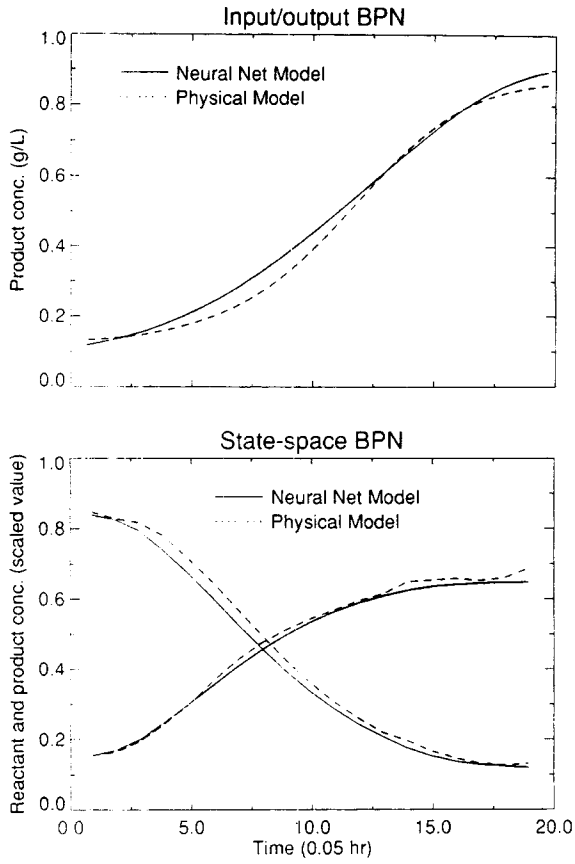


**Figure 6. Prediction of input/output BPN and state-space BPN.**

$$\frac{dx_1}{dt} = -(u + \alpha u^p)x_1, \quad x_1(0) = 1, \tag{21}$$

$$\frac{dx_2}{dt} = ux_1, \quad x_2(0) = 0, \tag{22}$$

$$k_i = k_{i0}e^{-E_i/RT}$$

where $x_1 = C_A/C_{A0}$, $x_2 = C_B/C_{A0}$ and $u$ is the dimensionless coolant flow rate. The model parameters, $p$ and $\alpha$, are defined as follows: $p = E_2/E_1$ is the ratio of activation energies and $\alpha = k_{20}/k_{10}^p$ is the ratio of the pre-exponential factors. For the example used below, $p = 2$, $\alpha = 0.5$, and $u$ is required to be between 0 and 5. The objective function is to maximize the yield of product B at the final time:

$$\max \Phi(u(t)) = x_2(t_f) \tag{23}$$

First, a neural network model should be established by using the experimental data of the process. For a batch process, there are no standard methods to generate process data to ensure process identifiability by a neural network model. A sinusoidal function at different frequencies is used to generate the training data as recommended by Narendra et al. (1990).

Neural network models are usually used to represent an input/output model relating the manipulated and controlled variables for process control and optimization. In this article, both an input/output neural network and a state-space neural network model are studied and compared for the purpose of optimization. As mentioned before, the neural network model is used for a one-step prediction for the optimization of a batch process. Normally a multiple step ahead prediction will achieve a better prediction performance for a batch process. However, it is not necessary to use a multiple step ahead prediction in our study, since we assumed that the prediction horizon is the same as prediction horizon for the batch process. A three-layer backpropagation neural net model was trained for this dynamic optimization problem. By using a trial and error method, the number of the hidden nodes were chosen as 4 with the learning rate, $\beta$, at 0.15 for this process. Figure 6 shows that both the BPN input/output and the BPN state-space trained neural network models produced a good prediction of the output variables for just one-step ahead. Note that the input/output BPN only gives product concentration, that is, only one output.

Next, neural net based optimization is studied and compared for different neural net models. The dynamic matrix of neural network model is obtained by using the technique mentioned in the third section. A FSQP procedure developed by Zhou and Tits (1990) is used for the UDMC algorithm. The results show that the state-space model based neural network produces a better performance than the input/output neural network model as is seen in Figure 7, that is, 0.56 compared to 0.42. This is in spite of the fact that both the input/output model and state-space model produced a good simulation performance (Figure 6). Also, the "jaggedness" of the control variable in Figure 7b can be explained if it is recalled that Eaton and Rawlings found that the objective function was not sensitive to the control moves made toward the end of the optimization time. The reason for this is obvious, since the state-space model contains more physical information that the input/output model as mentioned previously.

A comparison of the neural-network-model-based optimization is made with the differential equations model (for example, the "true" model) based optimal profile. The state-space-neural-net-model-based optimization produces a performance which is very close to the value obtained with the true model, for example, the final objective function is 0.56 compared to 0.58, as shown in Figure 8. Finally, the feedback algorithm of UDMC is used to study on-line optimization when the process experiences disturbances and time-varying parameters. This is simulated by adding a measurable noise. Figure 9 shows that the optimization performance remains good, for example, 0.55, even when the output of the process is subjected to 5 percent noise.

Also, the effect of model uncertainty on system performance is examined by considering the simulated process to have some parameter error. Suppose that one of the parameters in this simulation model, $\alpha$, is varied randomly between 0.5 and 0.7 from batch to batch. Using the data generated by this random variation, the neural network model can still be trained to predict the process quite well. Therefore, the trained neural-network-model-based optimization can also achieve a good optimization performance even when a model parameter is varied randomly, as shown in Figure 10. This figure shows that when $\alpha = 0.7$, the optimization performance based on the neural net model can reach about 0.52 which is still very close to the optimum value ($x_2 = 0.54$) based on using the true model with $\alpha = 0.7$.

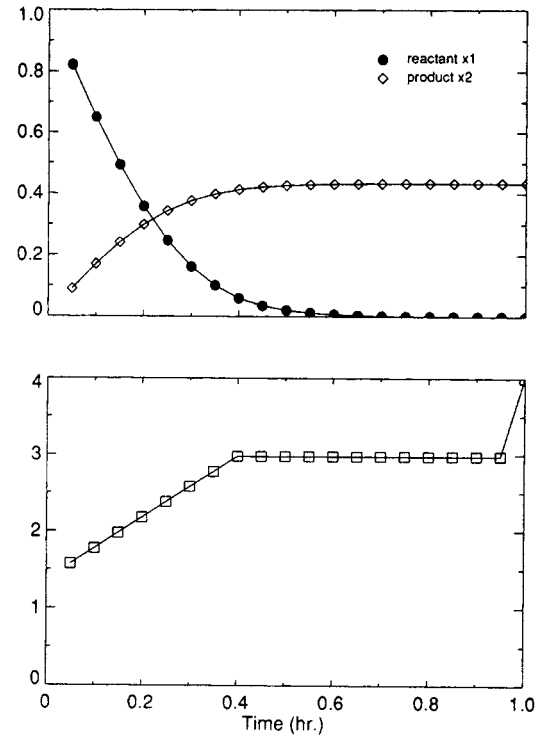


Figure 7a. Optimal trajectory based on input/output BPN.

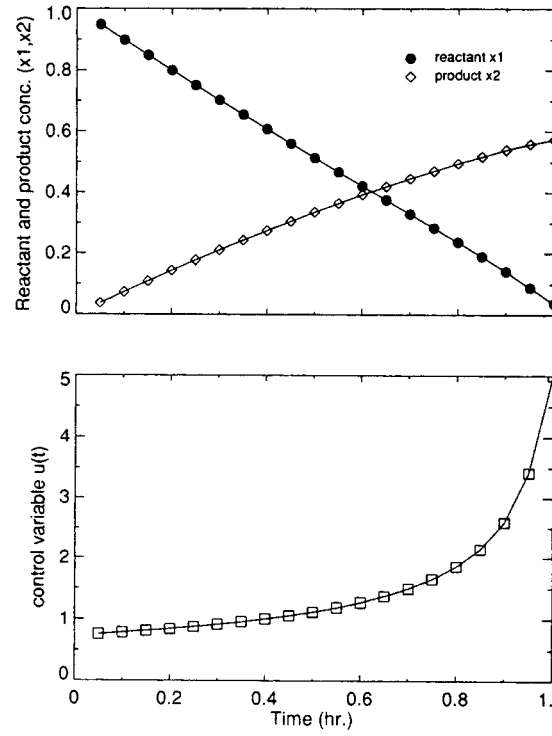


Figure 8. Comparison of optimal trajectory based on state-space BPN with true dynamic model.

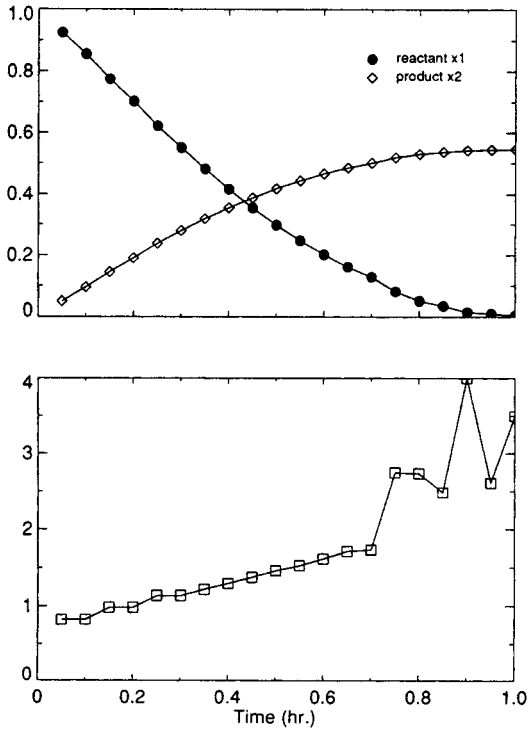


Figure 7b. Optimal trajectory based on state-space BPN.

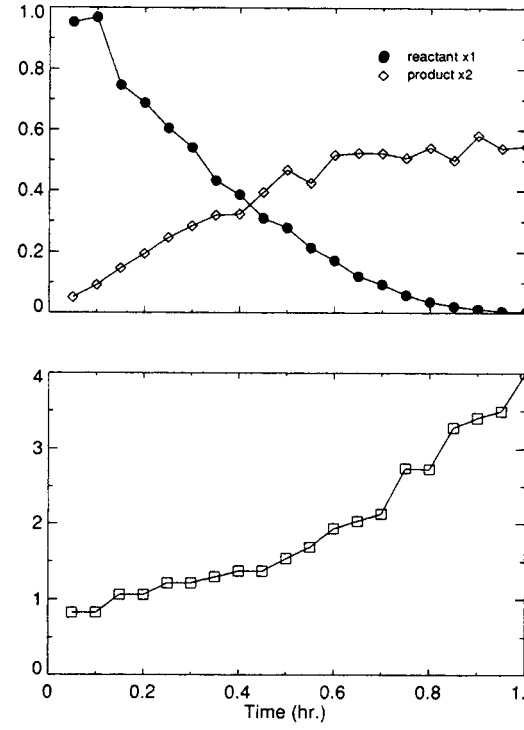


Figure 9. Optimal profile obtained by using state-space recurrent neural-net-model-based UDMC with 5 percent measurement noise.

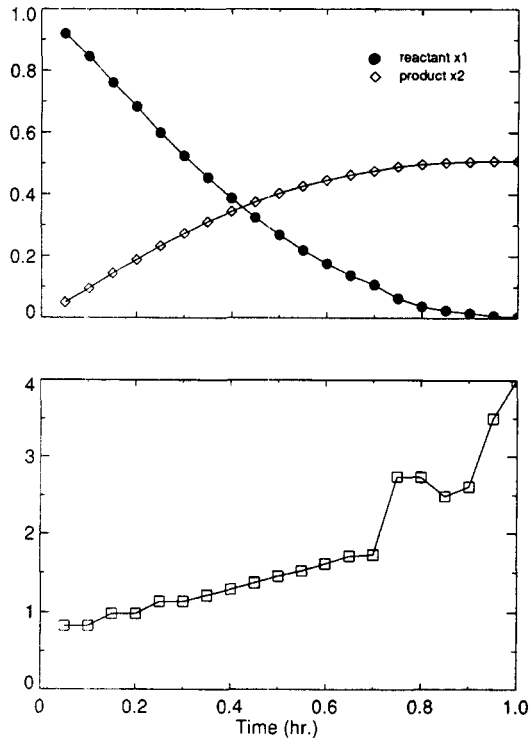


**Figure 10. Optimal trajectory of state-space recurrent neural network model with uncertain parameters.**

### *Example 2: a continuous baker yeast fermentation process*

The objective of the optimization is to maximize an appropriate objective function for the process. For this continuous fermentation process, the objective function is the cellular productivity which is the product of the dilution rate, $D$, and the cell concentration, x:

$$\max_{x} \Phi = \int_{t_0}^{t_f} Dx dt \tag{24}$$

This objective function is usually used for optimization of a continuous process (Harmon et al., 1987; Jang et al., 1987). It is based on a receding horizon control strategy. A computer simulation of a continuous baker's yeast fermentation was carried out to provide another test of the feasibility of the neural-net-model-based dynamic optimization method. The baker's yeast process was simulated using a time delay model (O'Neil and Lyberators, 1986) which is described by the following equations:

$$\frac{dx}{dt} = \frac{\mu_m z x}{z + K_s} - Dx \tag{25}$$

$$\frac{ds}{dt} = \frac{1}{Y_s} \frac{\mu s x}{(K_s + s)} + D(s_f - s) \tag{26}$$

$$\frac{dz}{dt} = a(s - z) \tag{27}$$

**Table 1. Simulation Model Parameters of a Continuous Fermentor**

| | | |
|---|---|---|
| $\mu_m = 0.7\ h^{-1}$, | $S_f = 30$ g/L | |
| $Y = 0.5$, | $a = 3\ h^{-1}$, | $K_s = 22$ g/L |

where $\mu_m$ is the maximum specific growth rate, $K_s$ is the Monod constant, $Y_s$ is the cell yield, and $z$ is a weighted average of the previous substrate concentrations. The parameter $a$, called the delay parameter, is a measure of the organism's ability to adjust its growth rate when a change in the conditions of the chemostat is brought about (Harmon, Svoronos, and Lyberators, 1987). The model parameters used for the simulations are given in Table 1. The input-output data needed to train the neural network were generated by submitting the system to a series of changes in the input which is varied as step inputs to a PRBS signal to cause variations from the initial steady-state conditions of x = 12 g/L, $s = z = 6.9$ g/L, and $D = 0.15\ h^{-1}$. The specific sequence is given by:

*Step 1*: input from $D = 0.2$ to $D = 0.4$

*Step 2*: input from $D = 0.4$ to $D = 0.1$

*Step 3*: input generated by the addition of an independent, persistently exciting signal to the system input (Gustavsson et al., 1977), PRBS signal between 0.1 and 0.4.

Also the data is scaled to fit the sigmoidal function used to train the neural network model of this fermentation process. As mentioned previously, a multiple step ahead prediction by the neural network model is normally necessary to achieve dynamic optimization for a continuous process. Therefore, a recurrent neural network model is much better to use for optimization purposes. The recurrent neural net model is trained using the simulated data with a three-layered net structure. The initial weight matrix for this recurrent neural network model is chosen as the optimal weight matrix obtained by first training a BPN. It is known that the prediction of steady-state points is a multistep ahead problem. This is equivalent to fixing the input to the system while obtaining the output from the network for multiple time steps until the steady-state output is reached. Therefore, the steady-state behavior is used to estimate the quality of the multiple step ahead prediction of the neural net model. In Figure 11, the steady-state behavior of the trained RNN and BPN models are compared. The network models are both trained in an off-line, open-loop mode. It is clearly seen that the RNN has much better multiple step prediction capability as described previously. If a much better signal to excite the process could be found, it could be possible to improve the steady-state prediction performance of the BPN. However, no elegant method exists to find a better signal for a general nonlinear process. In our study, the PRBS was used to generate the training data. The steady-state prediction performance is shown in Figure 11. Similar results are reported by Su and McAvoy (1992). The results shown in Figure 12 occur after the loop is closed and the neural network implementation of UDMC is realized. The optimization results obtained with this recurrent neural network model demonstrate that the proposed technique has significant capability for optimization of continuous processes. This initial nonoptimum point corresponds to $D = 0.05\ h^{-1}$, x = 14.153/L, and $s = z = 1.692$ g/L. The results in Figure 12 illustrate that the proposed algorithm can search for and achieve the optimal

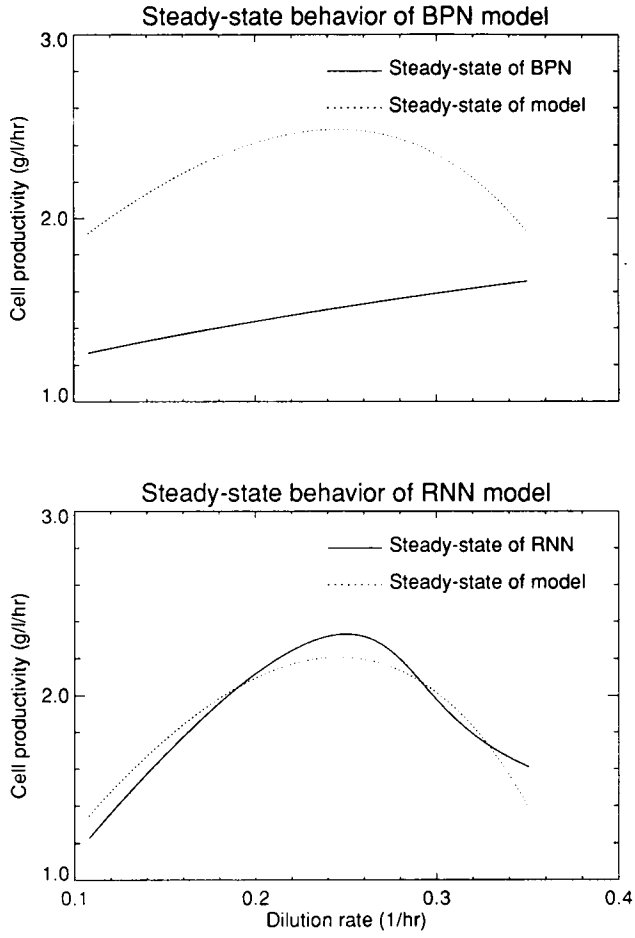


**Figure 11. Comparison of steady-state behaviors of BPN and RNN for continuous biochemical process.**

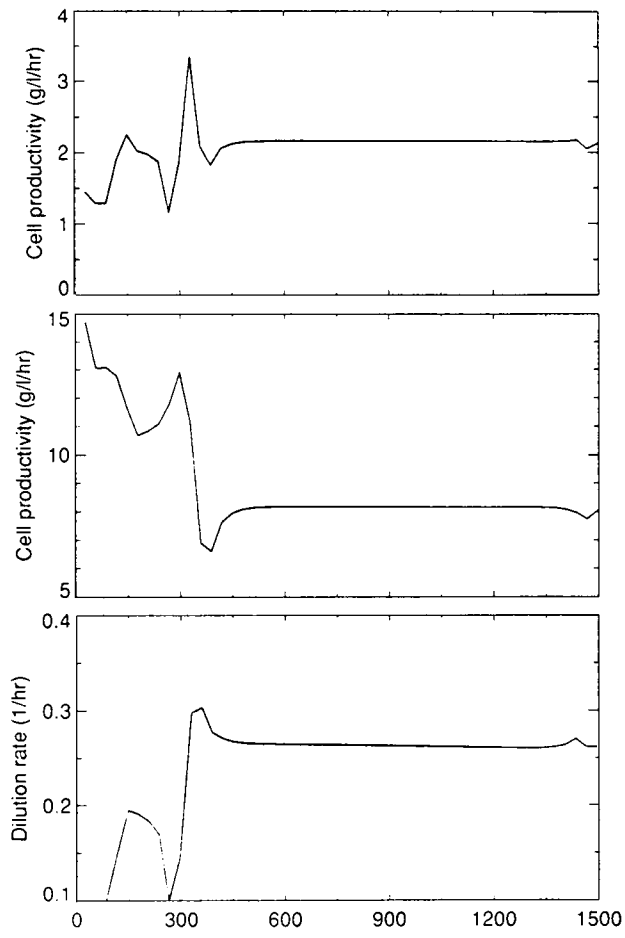


**Figure 12. RNN based dynamic optimizing control profiles for continuous biochemical process.**

performance when starting at some initial nonoptimum operating condition. The value of objective function (cell productivity) is increased from 1.25 to 2.21 g/L/h, after running this dynamic optimization algorithm. Therefore, the simulation study of dynamic optimization based on a neural network model shows that this technique achieves good performance, even when a conventional model is not available. Besides, this neural-net-model-based UDMC algorithm reduces the computation time when compared to a differential equation-model-based optimization. This result occurs because the network-based method enables calculation with algebraic equations. A significant improvement in computation time produced by using an algebraic equation was recently noted by Sistu et al. (1993). In this article, the computation time for a number of model predictive control techniques were compared. They showed that an approach based on orthogonal collocation on finite elements (which results in a set of algebraic equations) is much faster than directly integrating the ordinary differential equations, as is done in the UDMC approach. The neural network approach used here produces fewer algebraic equations than the orthogonal collocation approach, and therefore a reduction in computation time would also be expected. This smaller number of equations occurs because the neural network approach enables calculation with algebraic equations based on the relationship of the input/output variables. This result indicates promise for neural-network-based optimization.

## Conclusions

A dynamic optimization method which combines the neural network technique with the UDMC algorithm is illustrated in this article. An analysis of the different neural network structures for optimization purposes is discussed and illustrated by using two typical chemical processes. These kinds of processes are highly nonlinear and often difficult to model using first principles. The simulation studies demonstrate that the proposed technique has a strong potential for on-line application to complex nonlinear chemical processes. This is true because a neural network has the capability to adequately determine a dynamic model using process data even when the conventional models are not available. Also, this neural-net-model-based UDMC algorithm enables faster computation of the dynamic matrix required for the optimization algorithm.

## Acknowledgment

The authors are grateful to Dr. A. L. Tits for the use of the FSQP software package.

## Notation

$a$ = delay constant of baker-yeast fermentation model
$d_k$ = disturbance vector
$d\Phi/dt$ = gradient of the objective function
$D$ = dilution rate, $h^{-1}$
$f_n$ = neural network model
$H_j$ = output of the neurons of the hidden layer
$J$ = Jacobian matrix
$k_i$ = constants for chemical reaction rate coefficients
$K_s$ = Michaelis-Menten constant
$L$ = control time horizon
$n_u$, $n_y$ = maximum time lags for discrete time process
$n_{inp}$ = number of neurons in the input layer
$n_{hid}$ = number of neurons in the hidden layer
$p$ = ratio of activation energies
$P$ = prediction time horizon
$s$ = substrate concentration (g/L)
$s_f$ = inlet substrate concentration (g/L)
$u$ = vector of control variables
$U_i$ = scaled input vector
$w_{ij}$, $w_{jk}$ = synaptic weight matrix
$w1$, $w2$ = vectors of $z$ transformation for inputs of neural net model
x = biomass concentration (g/L)
$x$ = state variables
$x_i$ = mass fraction of component $i$
$y$ = output of the process
$Y_o$ = output vector of the output layer
$Y_s$ = substrate-to-biomass yield coefficient (g cell/g substrate)

### *Greek letters*

$\alpha$ = ratio of preexponential factor
$\Theta$ = input vector of dynamic neural net model
$\mu_m$ = maximum specific growth rate ($h^{-1}$)
$\sigma$ = transfer function of the motion
$\Phi$ = value of objective function

## Literature Cited


Bequette, B. W., "Nonlinear Control of Chemical Processes: A Review," *Ind. Eng. Chem. Res.*, **30**, 1391 (1991).

Bhat, N., and T. McAvoy, "Use of the Neural Nets for Dynamic Modeling and Control of Chemical Systems," *Comput. Chem. Eng.*, **38**, 573 (1990).

Chen, S., S. A. Billings, and P. M. Grant, "Nonlinear System Identification Using Neural Networks," *Int. J. Control*, **51**(6), 1191 (1990).

Chen, Q., and W. A. Weigand, "Adaptive Optimal Operation of a Bioreactor Based on a Neural Net Model," Proc. 2nd IFAC Symposium on Modeling and Control of Biotechnical Processes, N. Karim and G. Stephanopoulos, eds., Pergamon Press, Oxford, p. 193 (1992).

Cuthrell, J. E., and L. T. Beigler, "On the Optimization of Differential-Algebraic Systems," *AIChE J.*, **33**, 1257 (1987).

Eaton, J. W., and J. B. Rawlings, "Feedback Control of Chemical Processes Using On-line Optimization Techniques," *Comput. Chem. Eng.*, **14**, 469 (1990).

Gattu, G., and E. Zafiriou, "Nonlinear Quadratic Dynamic Matrix Control with State Estimation," *Ind. Eng. Chem. Res.*, **31**, 1096 (1991).

Gustavsson, L., L. Ljung, and Soderstrome, "Identification of Processes in Closed Loop-Identifiability and Accuracy Aspects," *Automatica*, **13**, 59 (1977).

Harmon, J., S. A. Svoronos, and G. Lyberator, "Adaptive Steady-State Optimization of Biomass Productivity in Continuous Fermentors," *Biotechnol. and Bioeng.*, **30**, 335 (1987).

Hornik, K., M. Stinchcombe, and H. White, "Multilayer Feedforward Neural Network are Universal Approximators," *Neural Networks*, **2**(5), 359 (1989).

Hornik, K., M. Stinchcombe, and H. White, "Universal Approximation of an Unknown Mapping and Its Derivatives Using Multilayer Feedforward Networks," *Neural Networks*, **3**(5), 551 (1990).

Jang, S., B. Joseph, and H. Mukai, "On-line Optimization of Constrained Multivariable Chemical Processes," *AIChE J.*, **33**(1), 26 (1987).

Kolvisto, H., et al., "Neural Predictive Control—A Case Study," *Proc. of IEEE International Symposium on Intelligent Control*, Arlington, VA (1991).

Lewis, J. C., "Dynamic Matrix Control for a Nonlinear CSTR Process," PhD thesis, Dept. of Chemical Engineering, Univ. of Missouri-Columbia (1989).

Lin, H. Y., "Dynamic Matrix Control of Nonlinear Processes," PhD Thesis, Dept. of Chemical Engineering, Univ. of Missouri-Columbia (1987).

Morshedi, A. M., "Universal Dynamic Matrix Control," *Chemical Process Control III*, Ashmore, CA (1986).

Narendra, K. S., and K. Parthasarthy, "Identification and Control of the Dynamical Systems Using Neural Networks," *IEEE Trans. on Neural Networks*, **1**(1), 4 (1990).

Nguyen, D. N., and B. Widrows, "Neural Networks for Self-Learning Control Systems," *IEEE Control and System Mag.*, **10**, 18 (1990).

O'Neil, D. G., and G. Lyberators, "Feedback Identification of Continuous Microbial Growth Systems," *Biotech. and Bioeng.*, **28**, 1323 (1986).

Psichogios, D. C., and L. H. Ungar, "Direct and Indirect Model-Based Control Using Artificial Neural Networks," *Ind. Eng. Chem. Res.*, **30**(12), 2564 (1991).

Rawlings, J., and L. T. Biegler, "Optimization Approaches to Nonlinear Model Predictive Control," *Chemical Process Control IV*, Houston, TX (1992).

Ray, R. H., *Advanced Process Control*, McGraw-Hill, New York.

Renfro, J. G., A. Morshedi, and A. Asbjornsen, "Simultaneous Optimization and Solution of Systems Described by Differential/Algebraic Equations," *Comput. Chem. Eng.*, **11**(5), 503 (1987).

Saint-Donat, J., N. Bhat, and T. J. McAvoy, "Neural Net Based Model Predictive Control," *Int. J. Control*, **54**(6), 1453 (1992).

Sistu, P. B., R. S. Gopinath, and B. W. Bequette, "Computational Issues in Nonlinear Predictive Control," *Comput. Chem. Eng.*, **17**(4), 361 (1993).

Sistu, P. B., and B. W. Bequette, "Nonlinear Predictive Control of Uncertain Processes: Application to a CSTR," *AIChE J.*, **37**(11), 1171 (1992).

Su, H. T., and T. J. McAvoy, "Identification of Chemical Processes Using Recurrent Networks," *Proc. Amer. Control Conf.*, **3**, 2314 (1992).

Zhou, T. J., and A. L. Tits, "User's Guide for FSQP Version 2.0B," Electrical Eng. Dept., Univ. of Maryland (1990).



*Manuscript received Apr. 29, 1993, and revision received Oct. 12, 1993.*