Korean J. Chem. Eng., **17**(4), 373-392 (2000)
FEATURED REVIEW

Applications of Artificial Neural Networks in Chemical Engineering

David M. Himmelblau†

Department of Chemical Engineering, The University of Texas Austin, Texas 78731, U.S.A.

Abstract−A growing literature within the field of chemical engineering describing the use of artificial neural networks (ANN) has evolved for a diverse range of engineering applications such as fault detection, signal processing, process modeling, and control. Because ANN are nets of basis functions, they can provide good empirical models of complex nonlinear processes useful for a wide variety of purposes. This article describes certain types of neural networks that have proved to be effective in practical applications, mentions the advantages and disadvantages of using them, and presents four detailed chemical engineering applications. In the competitive field of modeling, ANN have secured a niche that now, after one decade, seems secure.

Key words: Artificial Neural Networks, Control, Data Rectification, Fault Detection, Modeling

INTRODUCTION

Traditional approaches of solving chemical engineering problems frequently have their limitations, as for example in the modeling of highly complex and nonlinear systems. Artificial neural networks (ANN) have proved to be able to solve complex tasks in a number of practical applications that should be of interest to you as a chemical engineer. This paper is not a review of the extensive literature that has been published in the last decade on artificial neural networks nor is it a general review of artificial neural networks. Instead, it focuses solely on certain kinds of ANN that have proven fruitful in solving real problems, and gives four detailed examples of applications:

1. fault detection
2. prediction of polymer quality
3. data rectification
4. modeling and control

For those who want more information, Appendix A is a partial list of the many applications of ANN to chemical engineering problems, but space prohibits a review of these and the many other articles that have been published in the last 10 years. A good start to review ANN in general would be the *Handbook of Neural Computation* [Fiesler, 1996] and *Statistics and Neural Net Users* [Kay and Titterington, 2000].

What are the advantages people see in using artificial neural networks in constrast with first principles models or other empirical models? First, ANN can be highly nonlinear, second the structure can be more complex, and hence more representative, than most other empirical models, third the structure does not have to be prespecified, and fourth, they are quite flexible models. We will mention some of the disadvantages later on!

An ANN forms a mapping F between and input space X and an output space Y. We can distinguish three different kinds of mappings:

1. Both the input and output spaces are comprised of continuous variables, a typical case of process modeling;
2. The input space is comprised of continuous variables whereas the output space is comprised of a finite set of discrete variables as in classification and fault detection;
3. Both the input space and the output space are comprised of discrete variables that are mapped in so called associative nets (that will be ignored in this article).

In what follows we first discuss the concept of artificial neural networks, and explain how their parameters are identified. Then we specifically describe feedforward nets, recursive nets, and radial basis function nets, the nets that comprise the major types of nets reported in the literature and used in practice. Finally, we give some detailed examples of the application of ANN to common chemical engineering problems.

ARTIFICIAL NEURAL NETWORKS (ANN)

As the term artificial neural networks implies, early work in the field of neural networks centered on modeling the behavior of neurons found in the human brain. Engineering systems are considerably less complex than the brain, hence from an engineering viewpoint ANN can be viewed as nonlinear empirical models that are especially useful in representing input-output data, making predictions in time, classifying data, and recognizing patterns. Appendix A lists numerous articles I selected from the literature describing applications of interest to chemical engineers.

To read the literature on the theory and application of artificial neural networks, you have to become familiar with the prevalent jargon, a jargon that is somewhat foreign to engineering.

Fig. 1 shows the basic structure of a single processing unit in an ANN which will be referred to as a *node* in this work and is analogous to a single neuron in the human brain. A node receives one or more input signals, I_j, which may come from other nodes or from some other source. Each input is weighted according to the value w_{ij} which is called a *weight*. These weights are similar to the synaptic strength between two connected neurons in the human brain. The weighted signals to the node are summed and the result-

†To whom correspondence should be addressed.
E-mail: himmelblau@che.utexas.edu

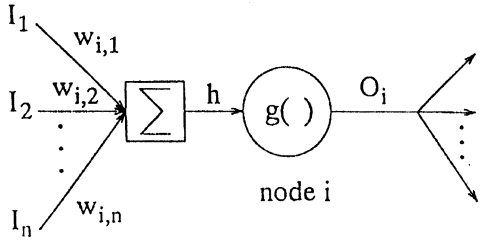

Fig. 1. Structure of a single processing node.

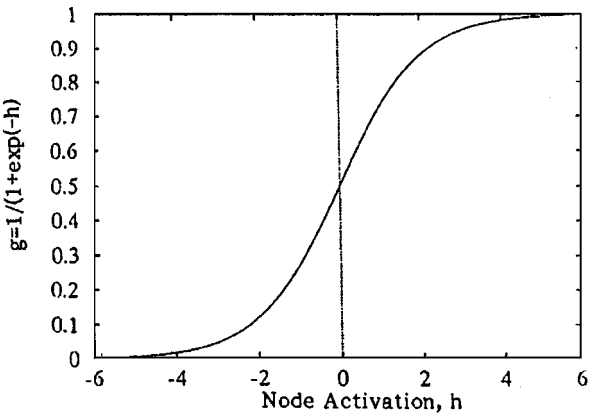

Fig. 2. Plot of the sigmoid transfer function.

ing signal, called the *activation*, *h*, is sent to a *transfer function*, *g*, which can be any type of mathematical function, but is usually taken to be a simple bounded differentiable function such as the sigmoid (Fig. 2). If the function *g* is active over the entire input space, it is termed a *global* transfer function in constrast with radial basis functions (to be described subsequently) that are *local* functions. The resulting output of the node O_j, may then be sent to one or more nodes as an input or taken as the output of an ANN model.

A collection of nodes connected to each other forms the artificial neural network. Cybenko [1987] and numerous subsequent articles have shown that various networks of such functions can approximate any input-output relation to the desired degree of accuracy (in the limit exactly). Of course, how many nodes to use cannot be prespecified, but refer to Baum and Haussler [1988] for some ideas. Fig. 3 is an example ANN. You can find numerous other architectures in the literature; Lippmann [1987] documents at least 50 other network configurations. Hybrid nets, that is nets composed of different or similar ANN, or nets connected to other types of models that are not ANN, cannot be discussed here, but a considerable literature exists for various types of architectures.

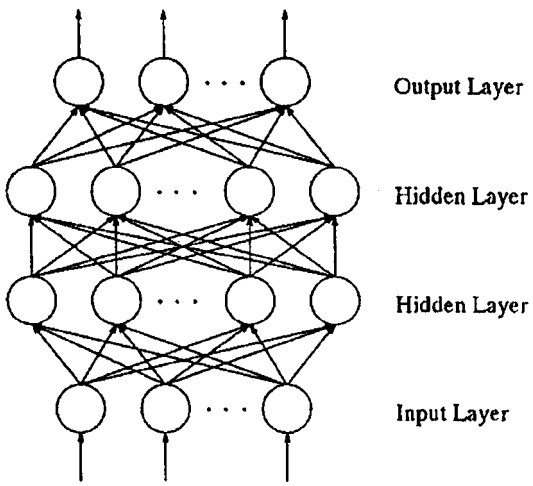

Fig. 3. Structure of a layered neural network.

A group of nodes called the *input layer* receives a signal from some external source. In general, this input layer does not process the signal unless it needs scaling. Another group of nodes, called the output layer, return signals to the external environment. The remaining nodes in the network, are called *hidden* nodes because they do not receive signals from or send a signal to an external source or location. The hidden nodes may be grouped into one or more *hidden layers*. Each of the arcs between two nodes (the lines between the circles in Fig. 3) has a weight associated with it. Fig. 3 shows a layered network in which the layers are fully connected from one layer to the next (input to hidden, hidden to hidden, hidden to output). Although this type of connectivity is frequently used, other patterns of connectivity are possible. Connections may be made between nodes in nonadjacent layers or within the same layer, or feedback connections from a node in one layer to a node in a previous layer can also be made. This latter type of connection is called a *recurrent* connection to be discussed below and, depending on the type of application for which the network is being used, such a connection may have a time delay associated with it.

Another part of the jargon associated with ANN models relates to model identification. Generally, there is no direct analytical method of calculating what the values of the weights are if a network is to model a particular behavior of a process. Instead the network must be *trained* on a set of data (called the *training set*) collected from the process to be modeled. Training is just the procedure of estimating the values of the weights and establishing the network structure, and the algorithm used to do this is called a "learning" algorithm. The learning algorithm is nothing more than some type of optimization algorithm. Once a network is trained, it provides a response with a few simple calculations, one of the advantages of using an ANN instead of a first principles model in cases for which the model equations have to be solved over and over again.

A key difficulty with optimization for neural network problems is that multiple minima occur (see Fukuoka et al. [1998]). Since most training procedures used for neural networks typically find local minima starting from randomly selected starting guesses for the parameters, it should be expected that local minima of varying quality will be found. While use of a global optimization procedure, such as genetic algorithms, branch and bound, or simulated annealing, might thus appear to be called for, the training time for such algorithms expands to an unacceptable degree. Consequently, satisfactory representation of data rests on the use of one local minimum achieved in a reasonable time.

Regardless of what training algorithm is used to calculate the values of the weights, all of the training methods go through the same general steps. First, the available data is divided into a training and test set(s). The following procedure is then used (called "supervised learning") to determine the values of weights of the network:

1. for a given ANN architecture, the values of the weights in the network are initialized as small random numbers;
2. the inputs of the training set are sent to the network and the resulting outputs are calculated;
3. some measure (an objective function) of the error between the outputs of the network and the known correct (target) values is calculated;
4. the gradients of the objective function with respect to each of the individual weights are calculated;
5. the weights are changed according to the optimization search direction and step length determined by the optimization code;
6. the procedure returns to step 2;
7. the iteration terminates when the value of the objective function calculated using the data in the test set starts to increase.

The type of objective function that is typically used in a training algorithm will be discussed subsequently below.

If target values are not known so that the learning goal is not defined in terms of specific correct examples, a procedure called "unsupervised learning" that is analogous to classification in statistics can be employed. A net will then produce output signals corresponding to the established input category, i.e., extract features from seemingly unstructured data. We will not discuss this type of training.

The purpose of partitioning the available data into the a training and test set is to evaluate how well the network *generalizes* (predicts) to domains that were not included in the training set. For non-trivial problems you probably cannot collect all of the possible input-output patterns needed to span the input-output space for a particular behavior or process to be modeled. Therefore, you have to train the network with some subset of all of the possible input-output patterns. However, the training set must be representative of the domain of interest if you expect the network to learn (interpolate among the data) the underlying relationships and correlations in the process that generated the data. If not, the net may not predict well for similar data, and may predict poorly for completely novel data (extrapolate). Noise in the data surprisingly automatically provides some smoothing, namely by adding the absolute value of the first derivative of the objective function as a penalty to the objective function. By holding some of the data back from the training phase to comprise a test set, you can evaluate how well the neural network can generalize by examining the value of the prediction error to the test set.

For three reasons you often need to carry out some type of unsupervised preprocessing of the data to be used in identifying a network so that you can

1. reduce the dimensionality of the data (feature extraction), and thus the complexity of the net used to represent it along with the correlations among variables;
2. transform the data into a more suitable format for processing by the net;
3. eliminate or reduce auto correlation for each variable.

FEED FORWARD NETWORKS

Three layer (sometimes called two layer) feed-forward artificial neural networks are commonly encountered models in the literature (see [Fine, 1999]). Computation nodes are arranged in layers and information feeds forward from layer to layer via weighted connections as illustrated in Fig. 4. While the neural network literature uses jargon such as training patterns, test sets, connections weights, and hidden layers, for modeling involving ANN, here we formulate artificial neural network models in terms of classical nonlinear system identification. Graphs of the network information flow help explain the more formidable equations.

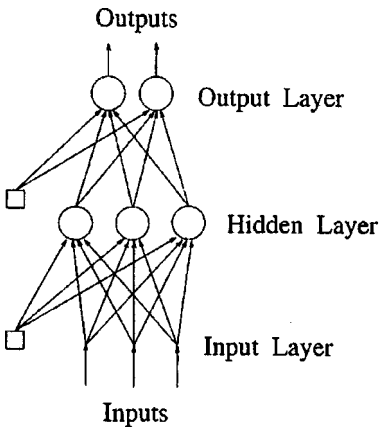

Fig. 4. Graph of the information flow in a feed-forward neural network. Circles represent computation nodes (transfer functions), and lines represent weighted connections. The bias thresholding nodes are represented by squares.

Mathematically, the typical feed-forward network can be expressed as

$$\mathbf{y}_i=\varphi_o[\mathbf{C}\varphi_h(\mathbf{B}\mathbf{u}_i+\mathbf{b}_h)+\mathbf{b}_o] \tag{1}$$

where $\mathbf{y}_i$ is the output vector corresponding to input vector $\mathbf{u}_i$, $\mathbf{C}$ is the *connection matrix* (matrix of weights) represented by arcs from the hidden layer to the output layer, $\mathbf{B}$ is the connection matrix from the input layer to the hidden layer, and $\mathbf{b}_h$ and $\mathbf{b}_o$ are the bias vectors for the hidden and output layers, respectively. $\varphi_h(\cdot)$ are $\varphi_o(\cdot)$ are the vector valued functions corresponding to the *activation* (transfer) *functions* of the nodes in the hidden and output layers, respectively. Thus, feed-forward neural network models have the general structure of

$$\mathbf{y}_i=\mathbf{f}(\mathbf{u}) \tag{2}$$

where $\mathbf{f}(\cdot)$ is a nonlinear mapping. Hence feed-forward neural networks are structurally similar to nonlinear regression models, and Eq. (2) represents a steady state process.

To use models for identification of dynamic systems or prediction of time series, a vector comprised of a moving window of past input values (*delayed coordinates*) must be introduced as inputs to the net. This procedure yields a model analogous to a nonlinear finite impulse response model where

$$\mathbf{y}_i=\mathbf{y}_t \text{ and } \mathbf{u}_i=[\mathbf{u}_t, \mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-m}] \text{ or } \mathbf{y}_t=\mathbf{f}([\mathbf{u}_t, \mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-m}]). \tag{3}$$

The lengths of the moving window must be long enough to capture the system dynamics for each variable in practice. In practice, the duration of the data windows are determined by trial and error (cross validation), and each individual input and output variable

might have a separate data window for optimal performance.

If you use windows of past inputs and outputs in feed-forward neural network models for dynamic modeling, the nets tend to be very large with the result that they include hundreds of parameters that have to be estimated. Each additional input to the neural net model adds greatly to the size of the network and the number of parameters that must be estimated. As a specific example, if the input vector at time *t* consists of 4 different variables, and the number of past values of each is selected to be 6, the net must contain 24 input nodes. If this hypothetical network were to have 12 hidden nodes and 2 output nodes, the total number of parameters to be estimated, including the bias terms, would total 326. The large number of parameters necessitates large quantities of training or identification data, and slower times for identification.

RECURRENT NETWORKS

Recurrent Neural Networks (RNN) have architectures similar to standard feed-forward Artificial Neural Networks with layers of nodes connected via weighted feed-forward connections, but also include time delayed feedback or recurrent connections in the network architecture. Examine Fig. 5.

Recurrent neural network models have the same relationship to feed-forward neural network models as autoregressive/infinite impulse response models have to moving average/finite impulse response models. RNN provide a more parsimonious model structure of reduced complexity because the feedback connections largely obviate the necessity of data windows of time lagged inputs. RNN also have a direct nonlinear state space interpretation useful in optimal estimation as discussed below.

Two individual variations of recurrent neural network architectures are commonly employed. The first is called an Internally Recurrent Network (IRN), that is characterized by time delayed feedback connections to the hidden nodes. Examine the connections in the hidden layer in Fig. 5. The remainder of the network comprise a standard feed-forward architecture. This structure is also known as an Elman network [Elman, 1990].

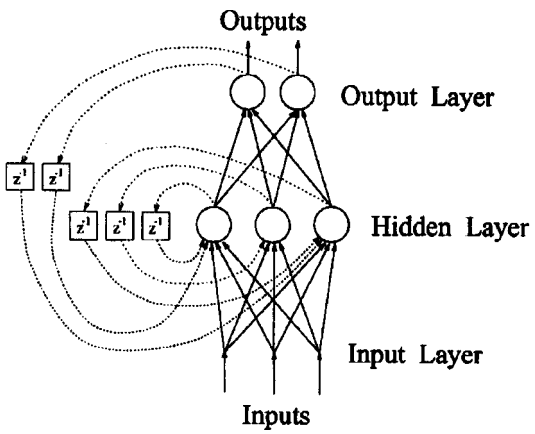

Fig. 5. Representation of internally/externally recurrent neural networks. Circles represent computation nodes, lines represent weighted connections, z^{-1} indicates time delay. For clarity not all recurrent connections are shown and bias nodes are omitted.

Externally Recurrent Networks (ERN), on the other hand, contain time delayed feedback connections from the output layer to the hidden layer. You can also envision a hybrid recurrent network which contains both types of recurrent connections, and might be described as an Internal-External Recurrent Network (IERN) such as the network pictured in Fig. 5. Simulation studies, both published and unpublished, have indicated no clear advantage of using an IRN versus an ERN, or even an IERN, for dynamic modeling. Both IRN and ERN models seem to be equally satisfactory in most process modeling applications.

Another possibility is to include a moving window of past outputs along with the past inputs to the network

$$\mathbf{y}_t=\mathbf{f}([\mathbf{y}_{t-1}, \mathbf{y}_{t-2}, \ldots, \mathbf{y}_{t-n}; \mathbf{u}_t, \mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-m}]) \tag{4}$$

analogous to a more general nonlinear time series model.

If we allow the vectors $\mathbf{u}_t$, $\mathbf{x}_t$, and $\mathbf{y}_t$ to denote the vector outputs of the input, hidden, and output nodes, respectively, at time *t*, we can formulate an IRN network as a discrete time model

$$\mathbf{x}_{t+1}=\varphi_h(\mathbf{A}\mathrm{x}_t+\mathbf{B}\mathbf{u}_t+\mathbf{b}_k) \tag{5}$$

$$\mathbf{y}_{t+1}=\varphi_o(\mathbf{C}\mathbf{x}_{t+1}+\mathbf{b}_o) \tag{6}$$

where and $\varphi_h(\cdot)$ are $\varphi_o(\cdot)$ are the vector valued functions corresponding to the activation functions in the hidden and output layers, respectively. In most applications the scalar elements of the Gaussian activation function for each hidden node are

$$\varphi_i(\mathrm{v}_i)=\exp\left(\frac{-\mathrm{v}_i^2}{2}\right) \tag{7}$$

where v_i is the total input to each node. Usually all the elements are made identical for simplicity. Linear activation functions are typically used in the output layer. The matrices **A**, **B**, and **C** are the matrices of connection weights for the hidden to hidden recurrent connections, input to hidden, and hidden to output connections, respectively, and the vectors $\mathbf{b}_h$ and $\mathbf{b}_o$ are the bias vectors for the hidden and output layers. By posing the IRN model in the above form we see that this type of recurrent neural network is a nonlinear extension of the standard linear state-space model in which the outputs of the hidden layer nodes, $\mathbf{x}_t$ are the states of the model.

In a similar fashion we can write nonlinear state space equations for the ERN. Whereas in the IRN model the states are the outputs of the hidden nodes, in the ERN model the states are the outputs of the nodes in the output layer so that the state space equations are

$$\mathbf{x}_{t+1}=\varphi_o[\mathbf{C}\varphi_h(\mathbf{D}\mathbf{x}_t+\mathbf{B}\mathbf{u}_t+\mathbf{b}_h)+\mathbf{b}_o] \tag{8}$$

$$\mathbf{y}_{t+1}=\mathbf{x}_{t+1} \tag{9}$$

where the matrices **B** and **C** and vectors $\mathbf{b}_h$ and $\mathbf{b}_o$ have the same meaning for the ERN as the IRN, and the matrix **D** is the matrix of weights for the recurrent connections from the output layer at time t−1 to the inputs of the hidden layer at time t.

Although the ERN and IRN can exhibit comparable modeling performance, they have different features that may make one more desirable than the other for a particular process. Just like the conventional linear state space model, the IRN does not have any structural limit on the number of model states because the number of hidden nodes can be freely varied. The ERN, however, can only

have the same number of states as model outputs because the outputs are the states. The IRN thus tends to be more flexible in modeling. The ERN has the advantage that the model states have a clear physical interpretation in that they are the variables of the process itself, whereas the states of the IRN are hypothetical and neither unique nor canonical.

Since both types of models have been posed as difference equations (rather than differential equations), to complete the model, a vector of initial values of the model states must be specified. Initialization of ERN models is simple because the user can observe the current values of the process output and use those values to initialize the states. Just as with linear state space models, IRN models are more difficult to initialize as the states lack physical meaning. In applications you usually initialize the states of IRN models with the median value of the activation function of the hidden nodes (0.5 if the activation function ranges from 0 to 1.0). Inaccuracies in the state initialization typically result in initial inaccuracies in the model predictions, but these die out in a time of the order of the dominant time constant of the process being modeled. Such startup transients can be minimized by holding the network inputs constant using the initial input vector and cycling the IRN model until the states and hence the output of the network becomes constant. This is equivalent to assuming that $u_t=u_o$ for all $t<0$.

SELECTION OF THE SPECIFIC ARCHITECTURE OF AN ANN

Once you decide on a particular category from which to select an ANN for your application, you still must determine the specific details concerning the structure of the nodes (transfer functions) and the connections between them. No general theoretically based strategy exists to carry out this task, but numerous strategies have been proposed. Refer to van de Laer and Henkes [1999] and the references therein or to Reed [1993]. An appropriate size network should exhibit:

1. Good "generalization", i.e., prediction for new data, by avoiding under- and over-fitting
2. Computational efficiency; the smaller the network, the fewer the parameters, less data is needed, and the identification time is less.
3. Interpretation of the input-output relation is so far as possible.

Because ANN are not unique, that is many nets can produce identical outputs from prespecified inputs, and many different goals can be deemed "best", searching for the "best" net in some sense is rarely an efficient use of your time. A "satisfactory" net is all that you need to make predictions or classify data.

If you choose to start the training (identification) with more nodes and connections than you eventually plan to end up with, the net will contain considerable redundant information after the training terminates. What you should do then is prune the nodes and/or links from the network without significantly degrading performance. Pruning techniques can be categorized into two classes. One is the sensitivity method [Lee, 1991]. The sensitivity of the error function is estimated after the network is trained. Then the weights or nodes which relate to the lowest sensitivity are pruned. The other class is to add terms to the objective function that prune the network by driving some weights to zero during training [Kamruzzom, 1992; Reed, 1993]. These techniques require some parameter tuning which is problem dependent to obtain good performance. An alternate approach to building a net (the "growing" technique) is to start with a small number of hidden nodes and add new nodes or split existing nodes if the performance of the network is not satisfactory. Pruning is identical to backward elimination and growing to forward selection in regression.

You can apply principal component analysis, or the Karhunen-Loeve transformation, to your data set to reduce the number of inputs to a net, and hence reduce the size and structure of the net. The transformed coordinates can be arranged in order of their significance, with the first being the components corresponding to the major eigenvectors of the correlation matrix (largest eigenvalues). A major weakness of these methods is that they are not invariant under a transformation of the variables. For example a linear scaling of the input variables (that may be caused by a change of units for the measurements or by scaling needed for identification) is sufficient to modify the PCA results. Feature selection methods that are sufficient for simple distributions of the patterns belonging to different classes can fail in classification tasks with complex decision boundaries. In addition, methods based on a linear dependence (such as *correlation*) cannot take care of arbitrary relations between the pattern coordinates and the different classes.

PARAMETER IDENTIFICATION

If you choose one of the Recurrent Neural Network (RNN) structures as a model, Eqs. (5) and (6), or (8) and (9), how do you estimate the values of the parameters (the weights) of the network? The standard way from the perspective of investigators using neural networks is to train the networks to reproduce the desired dynamic behavior using the backpropagation-through-time algorithm [Hertz, 1991]. Closer examination of this technique reveals that what is really being carried out is conventional prediction error estimation [Lyung, 1987] which will be briefly described here.

Let the parameters vector in the RNN nonlinear state-space model be denoted by $\boldsymbol{\theta}$ where

$$\boldsymbol{\theta}=\{\mathbf{A};\ \mathbf{B};\ \mathbf{C};\ \mathbf{b}_h;\ \mathbf{b}_o\} \tag{10}$$

for the IRN model and

$$\boldsymbol{\theta}=\{\mathbf{B};\ \mathbf{C};\ \mathbf{D};\ \mathbf{b}_h;\ \mathbf{b}_o\} \tag{11}$$

for the ERN model. Let the vector of prediction errors of either model be

$$\boldsymbol{\varepsilon}_t(\boldsymbol{\theta})=\mathbf{y}_t-\hat{\mathbf{y}}_t(\boldsymbol{\theta}) \tag{12}$$

where $\mathbf{y}_t$ is the vector of observed outputs and $\hat{\mathbf{y}}_t(\boldsymbol{\theta})$ is the vector of predictions from the model. The observed data from the process being modeled is the set of input-output vector pairs

$$\mathbf{Z}^N=\{\mathbf{y}_1,\ \mathbf{u}_1;\ \mathbf{y}_2,\ \mathbf{u}_2;\ \ldots,\ \mathbf{y}_t,\ \mathbf{u}_t;\ \ldots,\ \mathbf{y}_N,\ \mathbf{u}_N\} \tag{13}$$

where N is the number of data samples and $\mathbf{u}_t$ is the process input vector. The goal in prediction error identification is to minimize the prediction error of the model for the data set $\mathbf{Z}^N$ by adjusting the

parameter vector $\boldsymbol{\theta}$, i.e.

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}, \mathbf{Z}^N) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon_t^T(\boldsymbol{\theta})\varepsilon_t(\boldsymbol{\theta}) \tag{14}$$

Eq. (14) is the standard unweighted least squares objective function. When working with data containing outliers it is often more robust to use

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}, \mathbf{Z}^N) = \frac{1}{N}\sum_{t=1}^{N} \zeta(\varepsilon_t(\boldsymbol{\theta})) \tag{15}$$

where the function $\zeta(\varepsilon)$ is a positive, scaler function such as

$$\zeta(\varepsilon) = |\varepsilon| \text{ or } \zeta(\varepsilon) = \log\left(1 + \frac{1}{2}\varepsilon^2\right)$$

Because ANN models are nonlinear in the coefficients, iterative methods must be used to minimize Eq. (14). The backpropagation algorithm is a gradient descent scheme that is well suited for parallel implementation in hardware as each stage uses only local information about the inputs and outputs of each activation node. For calculations on a serial computer, more efficient optimization techniques such as the BFGS or conjugate gradient algorithms are preferred. Although analytic formulation of the gradients of $J(\boldsymbol{\theta}, \mathbf{Z}^N)$ with respect to $\boldsymbol{\theta}$ given the specific equations for the ANN is quite complex because of the existence of state feedback in recurrent nets, use of the gradient calculation as done in the BP algorithm [Hertz, 1991; Werbus, 1990] is both intuitive and computationally efficient. Analytical gradients of the objective function can be combined with an efficient quasi-Newton optimization code such as NPSOL in MATLAB or GRG2 in Excel to yield rapid parameter identification. We do not recommend trying to program the parameter identification code; instead use a commercial code focusing on ANN.

The parameter estimation scheme described above is known as prediction error estimation. An inherent assumption underlying this strategy is that the process output measurements, $\mathbf{y}_t$, only contain additive white noise (noise uncorrelated in time) while the process inputs are assumed to be deterministic. In reality, these assumptions are rarely met, and it can be shown that even when simple linear regression is used to model a steady-state process, the presence of noise in the independent variable will yield biased parameter estimates and biased predictions. Noise in the inputs is also a serious problem in the identification of linear dynamic models because when the effect of input noise is neglected, and it exists, prediction error methods cannot give consistent parameter estimates. If the noise characteristics of the process measurements are known, this problem can be ameliorated to a degree, but in general how to resolve the problem is still open. For nonlinear, nonparametric system identification such as for ERN or IRN, the problem of bias similary exists, and is further complicated by the nonlinearity of the model. In the case of nonlinear systems modeled by parametric models, various types of linearization based error-in-the-variables methods have been proposed [Kim et al., 1990]. Similar methods could be applied to neural network models if model bias became a serious problem.

Another problem with using the prediction error method has to do with the uncertainty associated with predicted output values. You cannot assume the values are not autocorrelated even if the residual errors are normally distributed, hence any confidence limits you place on the outputs must be developed with care.

THE BIAS/VARIANCE ISSUE

The great strength of neural networks, in general, is their ability to "learn" (represent) arbitrary mappings through their role as nonparametric estimators. This strength is also a weakness because in fitting input-output data, a large number of weights must be adjusted during training. If we consider the problem to be one of forming an estimate $y = f(\mathbf{x}; D)$, of an unknown model, $E[y|\mathbf{x}]$, given a training set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, the mean square estimation error between the function we create and the actual model is

$$E[(f(\mathbf{x}; D) - E[y|\mathbf{x}])^2] = (E[(f(\mathbf{x}; D)] - E[y|\mathbf{x}])^2 + E[(f(\mathbf{x}; D) - E[f(\mathbf{x}; D)])^2] \tag{16}$$

for any arbitrary $\mathbf{x}$ and all possible realizations of D. The first term on the right hand side of the equality sign is the square of the bias between our estimate and the unknown model, and the second term is the variance of our estimate, i.e.

$$(\text{estimation error})^2 = (\text{bias})^2 + \text{variance} \tag{17}$$

thus decomposing the estimation error into bias and variance components. A trade-off exists between reducing bias and variance in estimation theory [Goman et al., 1992; Moody, 1994]. A simple parametric model with few parameters may show low variance in the estimation error but intolerable bias in its predictions due to an inability to capture the complexity of the system being modeled. A traditional feed-forward neural network with hundreds or thousands of weights may have very low bias but high variance due to over-fitting of the noisy training data. The goal is to minimize both bias and variance. You may be able to reduce variance by using larger and larger training sets, and to reduce bias by increasing size of the network, making a large optimization problem quite difficult to solve. But a more common approach to the control of estimation bias and variance in modeling feedforward ANN is that of periodic stopping during training and using cross validation to evaluate the residual error. When the residual error no longer decreases, training is stopped and the weights (coefficients) are fixed. This procedure is a form of regularization and is discussed from a system identification perspective in [Sjoberg and Ljang, 1992]. Other methods of controlling both bias and variance in neural network models include reducing the number of weights through pruning or slowly allowing the network to grow while training to prevent over-parameterization.

Recurrent networks alleviate many of the problems of over-fitting and the need for large training sets characteristic of feedforward networks when applied to modeling dynamic processes. The absence of a need for a history window for each input variable as well as fewer hidden nodes translates into significantly fewer weights and less chance of over-fitting for a given data set. Incorporation of prior knowledge about the process to be modeled into the neural net as in Ungar's work [Psichogious and Unger, 1992] may allow the parameter count to be reduced even further.

MODEL VALIDATION

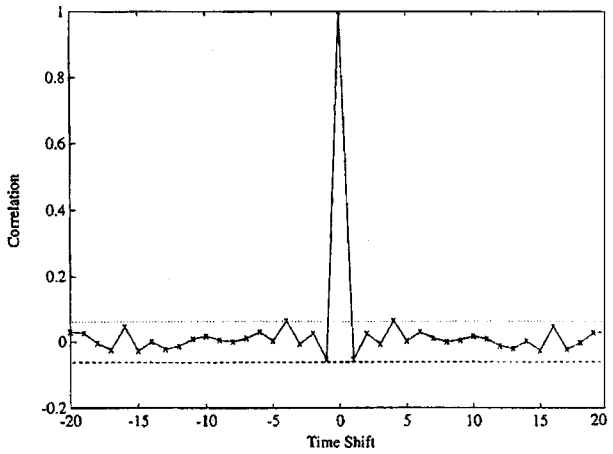

Fig. 6. The autocorrelation function for white noise.

Model validation is an important part of system identification. Although a large number of statistical hypothesis tests and evaluation criteria have been developed for linear, steady-state systems, the problem is much more complicated for nonlinear, dynamic systems. A simple criterion of model validity is the value of the objective function $J(\boldsymbol{\theta}, \mathbf{Z}^N)$ when the model is applied to a data set ($\mathbf{Z}^M$) different than the data set ($\mathbf{Z}^N$) used for system identification.

However, such a criterion does not distinguish between error caused by model mismatch (bias) and the error due to data corruption. More sophisticated tests are based on correlational analysis in which you examine the prediction errors, $\varepsilon_t(\boldsymbol{\theta})$. If a non-linear, non-parametric model is adequate and unbiased, then the prediction errors should be uncorrelated with all linear and nonlinear combinations of past inputs and outputs [Billings and Varn, 1986]. This outcome can be determined using the normalized cross-correlation function

$$\hat{\phi}_{\psi_1\psi_2}(\tau)=\frac{\sum_{t=1}^{N-\tau}\psi_1(t)\psi_2(t-\tau)}{[\sum_{t=1}^{N}\psi_1^2(t)\sum_{t=1}^{N}\psi_2^2(t)]} \tag{18}$$

Here $\hat{\phi}_{\psi_1\psi_2}$ is the normalized cross-correlation between two variables (time series ψ_1 and ψ_2), τ is the time shift, and t is the time index. You can plot the $\hat{\phi}_{\psi_1\psi_2}(\tau)$ as a function of τ for both positive and negative time lags. Examine Fig. 6 which is an example of $\hat{\phi}_{\psi_1\psi_1}(\tau)$ when ψ_1 is a white noise sequence, and exhibits negligible autocorrelation. Because the estimated correlations will never be exactly zero, approximate 95% confidence bands can be drawn as $\pm 1.96/\sqrt{N}$ for large N to indicate if the correlations are significant. For multivariate, nonlinear models it is of course impractical to check every possible cross-correlation, but the auto and cross-correlations should be calculated for the residuals as a minimal check on model validity.

RADIAL BASIS FUNCTION NETWORKS

If you view ANN such as shown in Figs. 4 and 5 as providing system outputs that result from a finite sum of weighted outputs of nonlinear functions forming the hidden nodes, then numerous networks are analogous to ANN. One type is the radial basis function network (RBFN) which was first used for process modeling by Chen et al. [1990]. Lee et al. [1999] and Gurumoorthy and Kosanovich [1998] review some of the theory (existence, uniqueness, stability etc). The structure of a RBFN (Fig. 7) differs from an ANN in that the inputs to the network are fed directly into the hidden nodes through connections with unity fixed weights. Each node represents only a limited range of the total range of an input variable, hence is a local function. The transfer function of each hidden node is a radial basis function (RBF) usually Gaussian or ellipsoidal:

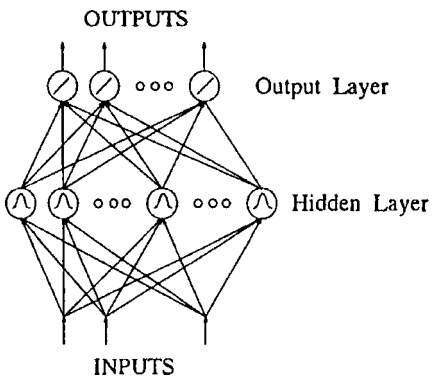

Fig. 7. Structure of the radial basis functions network.

$$g(\mathbf{I})=\exp\left(-\frac{\|\mathbf{I}-\mathbf{c}\|^2}{\sigma^2}\right) \tag{19}$$

where $\mathbf{I}$ denotes the vector of inputs to the node, $\mathbf{c}$ is a vector which centers each function of the RBF in the input space, σ is a "span" parameter, and $\|\cdot\|$ a vector norm. Note that the output of this radial basis function is 1 when $\mathbf{I}=\mathbf{c}$ and drops off to zero as $\mathbf{I}$ moves away from $\mathbf{c}$ (figuratively shown by the sketches in the circular node symbols in Fig. 7). The outputs of the hidden nodes are then sent to an output layer through a layer of weighted connections. The weighted signals are summed, and the sum forms the output of the network, i.e. the transfer functions in the output layer are linear. The sum of overlapping functions $g(\mathbf{I})$ form a smoothed representation of data as do ANN, and have been shown to be capable of universal approximation [Frombe, 1988].

Two major advantages exist for this type of network structure. First, using established numerical methods for clustering (grouping) data, the values of c can be calculated for each hidden node. Selection of the number of hidden nodes is a complex problem in clustering. The values of σ can be arbitrary or evaluated (separately from c) by simple optimization. The weights to the output layer can be calculated directly during training by linear regression because of the linear relations between output nodes and the outputs of the radial basis functions. Thus, the time to train the network is much shorter than for ANNs. Second, if a unique new input vector is encountered in testing, the network output will go to zero because of the local properties of the RBF in Eq. (19). This is a major advantage over most other network types since neural networks generally extrapolate for new inputs very poorly. While RBFN suffer from this same problem, at least they are capable of detecting when the network is being asked to extrapolate.

One major disadvantage of RBFN is that like ANN time must be explicitly incorporated into the structure by using a window of past process inputs and outputs as the inputs to the network. Un-

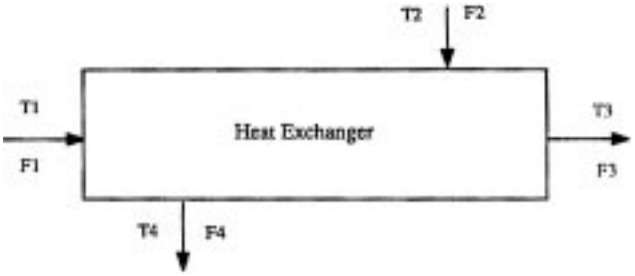

Fig. 8. A typical heat exchanger. $F_1=F_3$ and $F_2=F_4$ for the example.

Table 1. Prespecified data for the normally operating heat exchanger

	Stream 1	Stream 2	Stream 3	Stream 4
Temperature, T (K)	533	977	708.2	599.8
Flow rate, F (kg/hr)	9080	4086	9080	4086
Pressure, P (kPa)	207	345	148	236

fortunately, with RBFN, the size of the network scales exponentially with the number of inputs to the network. Each of the hidden nodes represents a bump in the input space, and the number of bumps required to model a process over the entire space rises exponentially with the dimension of the input space. Even for steady state models, RBFN become impractical with more than 4 or 5 input variables.

We now present four examples of applications of ANN in typical chemical engineering processes.

EXAMPLE 1. FAULT DETECTION USING AN ANN

This example from Suewatanakul [1993] demonstrates the use of a feedforward ANN to detect faults in a heat exchanger. Fig. 8 is a sketch showing the input and output measurements of an exchanger. The temperature and flow rate deviations from normal were deemed to be symptoms of the physical causes of faults. The two physical causes considered faults here were tube plugging and partial fouling in the heat exchanger internals. The diagnostic ability of a neural network is compared with Bayesian and KNN classifiers to detect the internal faults.

Rather than using data from an operating heat exchanger, the Simulation Sciences code PROCESS was used to generate data for clean and fouled conditions for a steady state counter-current heat exchanger. To generate the data for both the clean and faulty conditions, a data file for each faulty (and normal) condition was prepared. Information about the thermodynamic properties of the streams, the fluid and exchanger physical properties, the configuration of the heat exchanger, the number of tubes, the size of the tubes and shell, and the fouling layer thickness in the tube and shell sides (for fouled conditions) was prespecified. Table 1 lists the physical data and the normal parameters for the heat exchanger.

Tube side:

Feed: mixture of water: ethyl benzene: styrene of composition (weight percent)

55: 25: 20

Number of tubes: 108; length: 4.88 m; outside tube diameter: 3.18 cm; thickness: 0.26 cm; tube arrangements: square tube pitch: 3.97 cm; tube-side fouling layer thickness: 0

Shell side:

Feed: water
inside diameter: 54 cm
shell-side fouling layer thickness: 0 cm

Baffles:

Number of cuts (segments) for each baffle: single

For the fault of tube plugging (tube-sided only), the degree of the fault was classified into 4 cases-the number of tubes plugged was 5, 3, 2, and 1. In the study of fouling (both for the tube-side and the shell-side), the degree of fouling was expressed as a function of the decreased cross-sectional area which was also classified into 4 cases, namely a decrease of 8%, 5%, 3%, and 1%, respectively. To make the simulated measurements more realistic, two different levels of normally distributed noise were added to the deterministic flows, pressure, and temperatures so that the coefficients of variation of the noise were 0.02 and 0.01. Fig. 9 illustrates a typical feedforward ANN used to classify the respective faults.

To train the ANN, each measurement, namely the temperatures of all four streams (T_1, T_2, T_3, and T_4), the two flow rates (F_1 and F_2), and the pressure drops in the tube and shell size (ΔP_{tube} and ΔP_{shell}), for both the normal (clean) and faulty states had to be linearly scaled to be between the range of −1 and 1. The network was trained so that 0.9 represented the normal state (yes) while 0.1 represented the faulty state (no). For example, a target output pattern of 0.9 0.9 0.9 represented a pattern in which the heat exchanger was clean. The target pattern from a state in which there

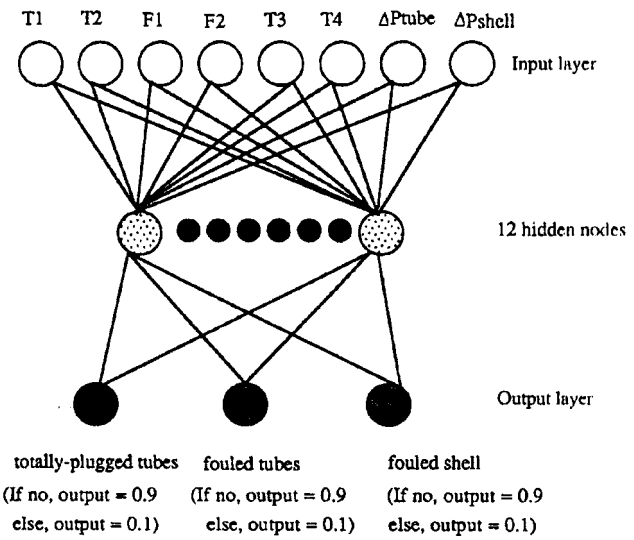

Fig. 9. The network architecture used in the training for internal fault detection (bias nodes are not shown).

was one or more totally-plugged tube was 0.1 0.9 0.9. The target output pattern of 0.9 0.1 0.9 represented a state in which fouled tubes existed while a target output pattern of 0.9 0.9 0.1 represented a state in which fouling existed on the shell side. The training data set contained 80 patterns each for the four different classes (a total of 320 training patterns). Another set of 20 patterns for each class was used for testing the classification capability of the ANN (a total of 80 patterns). NPSOL was the optimization code used in the training of the ANN.

In testing the ANN, a threshold value of 0.5 for the output of an output node was used as the discrimination criterion for classification. If the activation of an output node was greater than 0.5, then that node was deemed activated and represented a faulty state of the exchanger. If the node value was less than 0.5, then the node was deemed to be not activated, and the fault was said not to occur. Table 2 lists the results for one set of runs from the training and testing of the net.

By way of comparison, the two tradition classifiers, Bayesian and k-nearest neighbors (KNN, K=5 in Table 3), were also applied to the data sets. Table 3 lists the results.

Multivariate hypothesis tests on the means of the measurements gave much larger rates of misclassification. The conclusion is that ANN for this type of analysis are no worse than traditional methods of classification, and may have some edge.

EXAMPLE 2. PREDICTION OF POLYMER QUALITY USING AN ANN

This example from Barton [1997] illustrates the use of a recurrent ANN to predict polymer quality in an industrial reactor unit. Operation and control of industrial polymerization reactors is difficult because of the lack of reliable and timely measurements of key polymer product quality variables close to the reactor. Often product samples must be collected hours downstream from the reactor, after the polymer finishing operations. Measurement of these quality variables is typically performed off-line in a laboratory and

Table 2. Classification rates for the neural network (when the noise coefficient of variation was 0.02)

Tube plugging:		
Number of totally-plugged tubes	% correctly classified	
	Training	Testing
5	100	100
3	100	100
2	100	100
1	100	95
Tube-side fouling:		
% area was decreased	% correctly classified	
	Training	Testing
5	100	95.00
3	100	93.75
2	98.75	92.50
1	97.50	86.25
Shell side fouling:		
% area was decreased	% correctly classified	
	Training	Testing
5	100	96.25
3	100	95.00
2	98.75	93.75
1	97.50	92.50

Table 3. Classification rates yielded by traditional methods (when the noise coefficient of variation was 0.02)

Tube plugging				
Number of totally plugged tubes	Bayes procedure		5-NN procedure	
	Training % correct	Testing % correct	Training % correct	Testing % correct
5	100	100	100	98.00
3	100	100	100	95.00
2	100	98.00	100	92.50
1	100	93.00	100	89.75
Tube-side fouling				
% area was decreased	Bayes procedure		5-NN procedure	
	Training % correct	Testing % correct	Training % correct	Testing % correct
5	100	96.00	100	93.00
3	100	91.50	100	89.00
2	98.75	90.00	98.75	83.00
1	97.50	88.00	97.50	80.50
Shell-side fouling				
% area was decreased	Bayes procedure		5-NN procedure	
	Training % correct	Testing % correct	Training % correct	Testing % correct
5	100	97.00	100	96.00
3	100	95.00	100	93.50
2	98.75	91.00	98.75	90.00
1	98.75	89.00	98.75	80.50

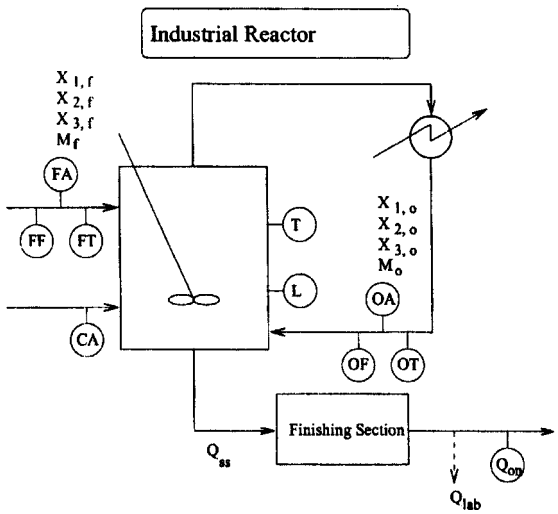

Fig. 10. Schematic of an industrial polymerization reactor.

may not be available for several hours after the sample is delivered to the lab. Thus, the measurements of the quality variables arrive too late to provide useful feedback for control.

Fig. 10 is a schematic of an industrial polymerization reactor and subsequent finishing section. The reactor feed flowrate, FF, and feed temperature, FT, are measured and the feed stream is analyzed for the monomer concentration, M_f, which can be manipulated, and several key impurities, $X_{i,f}$, which are disturbances that affect the product quality. The feed rate for the polymerization catalyst, CA, is a manipulated variable, and is also measured. The temperature in the reactor, T, is measured along with the liquid level, L. A recycle stream from the reactor is analyzed for unreacted monomer, M_0, and the key impurities, $X_{i,o}$. The recycle flowrate, OF, and the recycle temperature, OT, are measured, but not directly manipulated.

After the polymer product leaves the reactor, it must be processed another two to six hours in the finishing section. After the finishing section, the final polymer product is sampled every four hours and analyzed in the laboratory which takes another four hours to return the quality measurement, Q_{lab}. The product quality is also measured on-line Q_{on}, but the on-line instrument was unreliable and only sporadically available, and when in operation was difficult to keep calibrated.

The reactor in Fig. 10 was used to manufacture several different polymer grades spanning a wide operating range over which the reactor is highly nonlinear. The polymer finishing section imparts dynamics to the response Q_{lab} to changes in reactor conditions due to mixing.

An internally recurrent net (IRN) shown in Fig. 11 was used to model the dynamic process, particularly when changes occurred on transition from one grade of polymer to another.

The model corresponding to Fig. 11 is

$$\mathbf{x}_{t+1}=\sigma(\mathbf{A}\mathbf{x}_t+\mathbf{B}\mathbf{u}_t)$$
$$\mathbf{y}_{t+1}=\mathbf{C}\mathbf{x}_{t+1} \qquad (20)$$

where $\mathbf{x}_{t+1}$ is the network's internal state vector prediction (outputs of the hidden nodes) at time t+1; $\mathbf{y}_{t+1}$ is the network prediction from the vector of process outputs; in this case the network output is the product quality prediction, Q_{irn}; and $\mathbf{u}_t$ is the vector of network inputs at time t. The input vector to the IRN model consisted of the measurements FF, CA, M_f, $X_{1,f}$, $X_{2,f}$, $X_{3,f}$, T, and M_o, as indicated in Fig. 10. $\sigma(\cdot)$ is a vector-valued nonlinear (sigmoidal) activation function; and **A**, **B**, and **C** are weight matrices that are trained using historical data. Q_{ss} below represents the company's prediction from a previously developed steady state model.

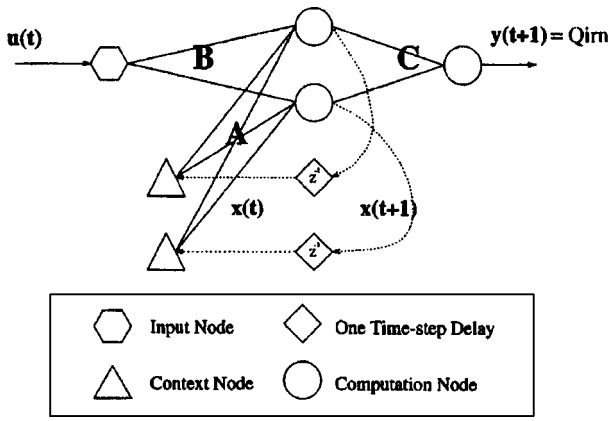

Fig. 11. The IRN structure used in modeling the reactor.

The IRN in this work was trained using approximately 10,000 hours of data collected at one hour sample intervals over several months, and tested using approximately 2,000 hours of data. A pure time delay of 3 hours for the product quality was found to yield the best IRN model, and was incorporated directly into the training and testing data. The difference between the 6 hour pure delay estimated for Q_{ss} and the 3 hour delay used for Q_{irn} is explained by the dynamic lag associated with product mixing in the finishing section. For this problem, an IRN with 4 hidden nodes gave the best performance. The measurements for Q_{lab} were interpolated to fill in the three hours of missing data between the Q_{lab} samples.

For this problem the IRN yielded better long term predictions than feed forward networks with feedback of past outputs because potentially erroneous old process output measurements did not have to be used in the network input vector. Long term predictions were necessary for this model because the goal was to use the model to develop strategies for directly controlling Q_{lab} during grade transitions.

The criterion for evaluating models for predicting Q_{lab} was the standard prediction error (SPE):

$$\mathrm{SPE}=\sqrt{\frac{1}{N}\sum_{t=1}^{N}(Q_{lab,t}-\hat{Q}_{mod,t})^2}$$

where Q_{lab} is the quality measurement, $\hat{Q}_{mod}$ is the model prediction (either Q_{irn} or Q_{ss}), t is the sample time, and N is the number of samples in the data set. In this work the product quality was scaled to lie between 0 and 10.

Fig. 12 compares the predictions from the linear steady-state model developed by the company operating the reactor with the predictions from the IRN for the test data set, and the (delayed) lab data (which may not necessarily be correct).

The IRN is clearly better able to capture the dynamic characteristics of the reactor, and does an excellent job of predicting the product quality several hours before the laboratory measurement become available. In Fig. 12 two polymer grades are shown and four

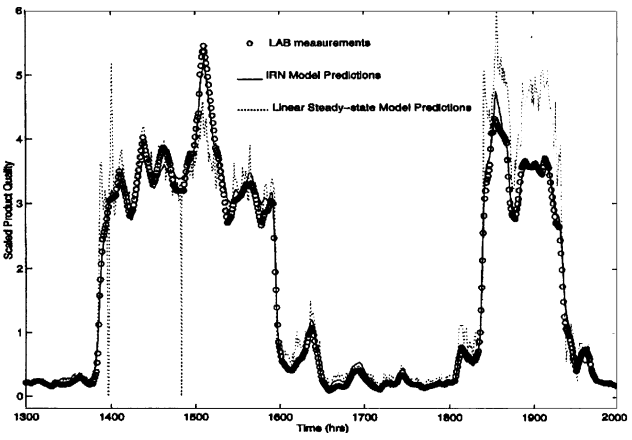

Fig. 12. Comparison of polymer product quality predictions for testing data set.

Table 4. Comparison of the Standard Prediction Errors (Scaled SPEs) for Quality Prediction by the Steady-State and IRN Models

Model	Number of parameters	Train	Test
Q_{ss}	12	0.458	0.466
Q_{irn}	57	0.191	0.210

grade transitions. Quality above 3 represents a "high" grade polymer, while quality below 1 represents a "low" grade polymer. Quality between 1 and 3 is in a grade transition region, and the IRN predictions are shown to be excellent in these regions. The large discrepancy between Q_{irn} and Q_{ss} in the high grade region between t=1800 and t=2000 in Fig. 12 is caused by the failure of the steady-state linear model to include the effects of impurity X_2.

Table 4 lists the SPE results for the IRN model versus the steady-state model when used to predict the product quality. The IRN model represents a marked improvement over the linear steady-state model in terms of SPE due to the modeling of process dynamics that are imparted by the polymer finishing section, and because the effects of measured impurities are included in the IRN model. The IRN model accurately predicts the quality variable over the entire reactor operating region, even between grades, eliminating the need for switching between different models in different operating regions.

EXAMPLE 3. DATA RECTIFICATION IN A DYNAMIC PROCESS USING AN ANN

This example from Karjala [1995] shows how nonparametric models can be used to adjust process data. The goal of data rectification is to compensate for random and nonrandom measurement errors by making suitable adjustments to the measured values of the process variables in order to provide the best (in some sense) estimates of the "true" values. This example focuses on detecting and eliminating gross and random errors, but does not address other important problems such as bias, correlations, nonperiodic data, missing measurements, and nonsymmetric probability distributions for the process measurements.

The term *data reconciliation* usually refers to the adjustment of process measurements to conform to some prescribed model. Since the model used in this example is nonparametric, that is the structure is built from the process data itself, we refer to the adjustment as *data rectification.* Furthermore, as explained below, the model involved does not necessarily use current data in the adjustment, but uses predicted values of the variables. Consequently, the term rectification in the sense of "making the data right" seems to be the appropriate word to use.

An unfortunate aspect of the literature on the rectification of data collected in a dynamic process is that the reported results of rectification always appear to be favorable, because the authors of the papers usually assume a known model and probability distribution for the noise in the data. To demonstrate how well rectification works, the authors simulate deterministic process data and corrupt the data by adding the known noise (almost always white Gaussian), and perhaps gross errors. Then, the noise and gross errors are removed using the known, exact model by the strategy proposed by the authors of the paper.

But how good is the rectification if the process model is not known exactly as is usually the case in practice? Furthermore, how good is the rectification if the probability distribution of the noise is different in practice than the assumed distribution? In this example we describe how internal recurrent neural nets can be used for the rectification of data from dynamic process whose true mathematical description is unknown and uncertain.

Dynamic data rectification can be posed as a general optimization problem in which the equations representing the model, the ANN, form part of the constraints:

$$\underset{\hat{\mathbf{m}}}{\text{Minimize:}}\ \Phi(\mathbf{m}_t, \hat{\mathbf{m}}_t, \mathbf{m}_{t-1}, \hat{\mathbf{m}}_{t-1}, \Lambda) \tag{21}$$

$$\text{Subject to: } \mathbf{f}(\dot{\mathbf{x}}_t, \mathbf{x}_t, \mathbf{u}_t, t)=\mathbf{0}$$
$$\mathbf{h}(\mathbf{x}_t, t)=\mathbf{0}$$
$$\mathbf{g}(\mathbf{x}_t, t)\geq\mathbf{0}$$

where Φ is a generalized objective function (normally the sum of squares or absolute values), $\mathbf{x}_t$ is a vector of state variables at time t (not all of which are measured), $\mathbf{m}_t$ and $\hat{\mathbf{m}}_t$ are the measurements and rectified measurements, respectively, $\mathbf{f}$ is the dynamic process model, $\mathbf{g}$ is a vector of inequality constraints (including bounds on the variables), and $\mathbf{h}$ is a vector of known equality constraints. For state space models the model constraint equations $\mathbf{f}$ are typically expressed as dynamic differential equations which are solved via orthogonal collocation on finite elements. The above minimization problem is usually performed for a moving window of past measurements. This window must be long enough to capture relevant process dynamics, but kept to a minimum to keep the nonlinear programming problem size tractable.

In data rectification using an IRN as the model $\mathbf{f}$ in Eq. (21), the idea is to build a model of the process in which one step ahead predictions can be made of both the input and the output variables. The input variables may need special treatment (see Barton [1996]). The mathematical model for rectification is

$$\hat{\mathbf{m}}_t=\mathbf{G}(\mathbf{m}_{t-1}, \mathbf{m}_{t-2}, \mathbf{m}_{t-3}, \Lambda, \mathbf{m}_0) \tag{22}$$

where $\hat{\mathbf{m}}_t$ is the estimate of the process measurement vector at time t, $\mathbf{m}_t$ is the actual measurement vector, and $\mathbf{G}(\cdot)$ is the nonlinear mapping we seek to identify. Note that with this model the current $\hat{\mathbf{m}}_t$ is not calculated from $\mathbf{m}_t$, the current measurement. You cannot use $\mathbf{m}_t$ as an independent variable in the nonparametric representation $\mathbf{G}$ because then system identification would yield the trivial identity mapping of $\hat{\mathbf{m}}_t=\mathbf{m}_t$. Because the noise in each measurement is assumed to be uncorrelated with the noise in previous measurements, it is possible to identify a system that describes the evolution of the measurement vectors in time using a prediction error model. The problems encountered when the noise in the measurements is autocorrelated is beyond our scope here. Because the model input vector $\mathbf{m}_{t-1}$ is not deterministic and contains measurement noise, the parameter estimates from the IRN model will be biased. Nevertheless, good results have been obtained for nonlinear processes in which the process measurements are corrupted by Gaussian and spike type measurement errors. In "training" the IRN, i.e., estimating the values of the coefficients, the targets are the measurements. The states of the net are the outputs of the "hidden" nodes, but these variables have no physical meaning, and the

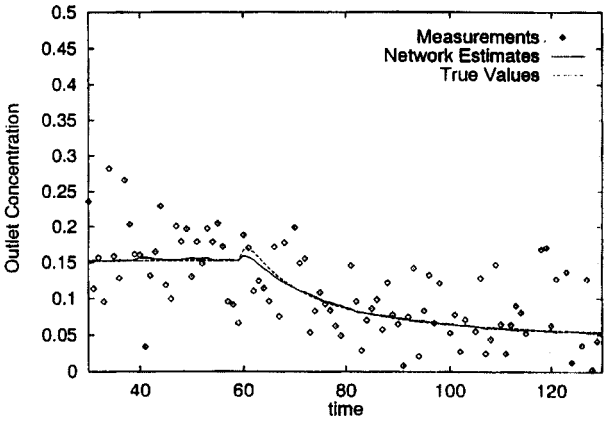

Fig. 13. Rectification of the outlet concentration.

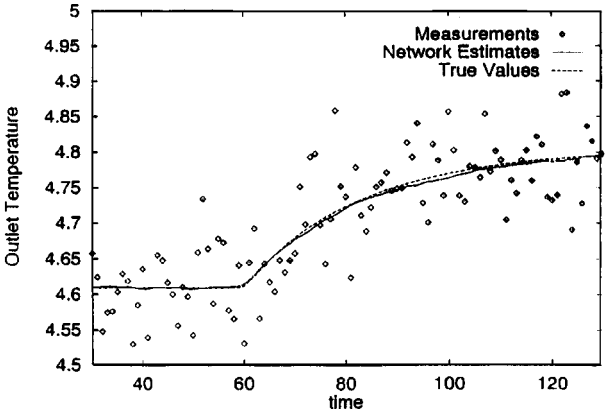

Fig. 14. Rectification of the outlet temperature.

number of nodes was determined by trial and error.

To compare rectified measurements with their "true" values, we used a model published by Seinfeld [1970] to develop simulated measurements (You cannot use actual process data and know what the "true" values of the measurements are.). The process consisted of a continuous nonlinear stirred tank reactor (CSTR) with a first-order exothermic reaction and heat removal by a coil or jacket. Jang et al. [1986] used this example to compare extended Kalman filtering to a nonlinear programming approach for state and parameter estimation. We added Gaussian noise to the simulated deterministic process measurements, and used the simulated noisy measurements in identifying an IRN model for rectification based on step changes in the inputs to the reactor.

Figs. 13 and 14 show the results using the trained net for rectification of the outlet concentration and temperature (the solid lines), respectively, together with the simulated test data (the diamonds), and the respective true values (the dashed line). You can see that the rectified values are excellent. Keep in mind that the results shown above did not require that the true model be used as a constraint. An important but usually ignored factor in process modeling is that variable delay occurs in a response from a process so that modeling the delay successfully in a theoretically based model is quite formidable. The IRN model automatically accommodates delay.

EXAMPLE 4. PROCESS CONTROL USING AN ANN

This example from MacMurray [1993] explains how an ANN model can be used for model predictive control. Model predictive control (MPC) involves using a process model to predict future process behavior so that the controlled variables can be manipulated such that the process will meet some desired future state(s), e.g., set-point, trajectory of set-points, maximize a yield, minimize the operating cost, or any combination of these. The performance of MPC relies heavily on the quality of the process model, and developing or identifying a valid process model is a major part of the work required to implement MPC.

Models based on first principles represent the process by a set of equations (linear and/or nonlinear ordinary and partial differential equations, and algebraic equations) derived from conservation laws and knowledge of the process. The model form is well established, but values of a few parameters have to be either estimated from data or derived from physical laws in order to make predictions using such models. Models constructed in this fashion are known as parametric models.

In contrast, nonparametric models comprise an arbitrary but usually very flexible model structure involving numerous parameters which must be estimated from ample process data. The key advantage of nonparametric models are that little or no *a priori* knowledge of the process is required, the development time for the model can be quite short compared to the first principles approach, and prediction using the identified model is rapid because sets of equations do not have to be solved for each new input vector. A process can not wait several seconds for the optimization problem to be solved if it requires controller action immediately.

When posed as an optimization problem, MPC uses the process model as a constraint in evaluating the trajectory of the process according to some objective function. Traditionally, the objective function is expressed as:

$$\Phi=\sum_{j=1}^{P}[y_{sp}(t_{k+j})-\hat{y}(t_{k+j})]^T Q_j[y_{sp}(t_{k+j})-\hat{y}(t_{k+j})] + \sum_{j=1}^{N_{cont}}\Delta u^T(t_{k+j})R_j\Delta u(t_{k+j}) \quad (23)$$

subject to:

$$\hat{y}=F(y,u) \quad (24)$$

$$u_{min}\le\Delta u(t_{k+j})\le u_{max} \quad (25)$$

$$|\Delta u(t_{k+j})|\le\Delta u_{max} \quad (26)$$

where $y_{sp}(t_{k+j})$ is the vector of set-points of the controlled variables and $\hat{y}(t_{k+j})$ is the vector of model predictions of the controlled variables at time t_{k+j} (t_k is the current time) which is generated by the model of the process represented by *F*. *P* is the prediction horizon; it defines the time interval for optimizing the process trajectory and how far into the future the process model will be called on to predict.

N_{cont} is the number of control moves in a control horizon, C, which will be made by the controller (C≤P). If C is shorter than the prediction horizon, the manipulated variables, **u**, are assumed to remain constant at their last computed values, $\mathbf{u}(t_{k+c})$, for the remainder of the precition horizon. Q_j and R_j are weighting matrices for each time step along the prediction and control horizons respectively, and are usually diagonal matrices.

Eqs. (25) and (26) restrict the changes which can be made in the values of the manipulated variables, since for a real process the manipulated variables may be limited in range and rate of change by physical limitations of the operating equipment.

As posed in Eqs. (23)-(26), the tuning parameters of MPC are *P*, *C*, and the weighting matrices, Q_j and R_j. Generally, a longer control horizon will make the controller more aggressive as will a shorter prediction horizon and *vice versa*. R_j penalizes control moves and hence large elements in R_j will make the controller less aggressive. There are no generally accepted rules of thumb for selecting the tuning parameters, but the computation time required to perform the optimization will increase as either P or C is increased. Depending on the complexity of the model used to describe the process, the computation time to perform the optimization may be the limiting factor in applying MPC, and hence influence the selections of the values of *P*, *C*, and the type of model itself. For stability of such a system when the model is an ANN refer to Kulawski and Brdys [2000].

To demonstrate the use of ERN in control, MacMurray developed an ERN model based on the work of Patwardhan [1991] who modeled a pilot plant packed distillation column (see Fig. 15). A feed stream enters the column (with flow rate, *feed*, and composition, *xf*) between two packed sections (a rectifying section and a stripping section) that contain a structured or unstructured packing material which is used to produce and support the liquid-vapor interface inside of the column. Mass transfer occurs between the vapor flowing up and the liquid flowing down the column. The vapor exiting at the top of the column is condensed, and part of the resulting liquid flow is returned to the column at the top (the reflux, *rr*); the remainder is taken as the distillate product (*dist*) with composition *xd*. Part of the liquid flow out of the bottom of the column is vaporized (*vbr*) in a reboiler and sent back to the bottom of the column. The remainder is taken as the bottoms product with a composition of *xb*.

The reason that this packed column is of interest is that the process gain changes sign over the operating region shown in Fig. 16.

Patwardhan's model for the separation of a binary mixture of cyclohexane and n-heptane contained two partial differential equations, three ordinary differential equations, and eight algebraic equations, and was used in lieu of data from the actual pilot plant column to simulate data for identification of the ERN. The model assumed:

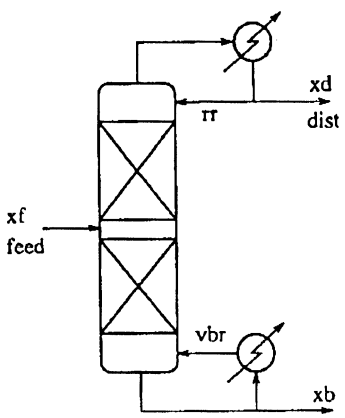

Fig. 15. Diagram of a packed distillation column.

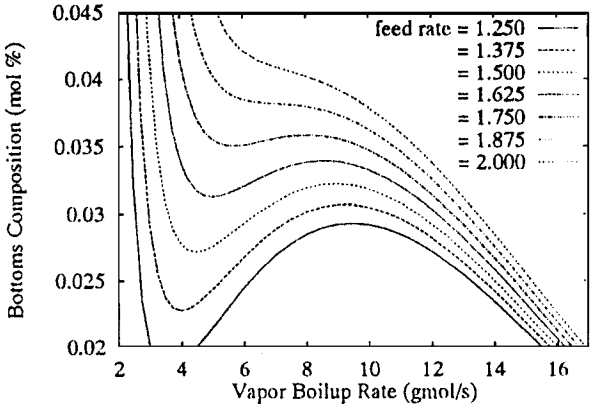

Fig. 16. Steady-state output of the VE model with Sulzer packing, xf=0.5 and xd=1−xb.

- equimolar counter diffusion
- negligible liquid phase mass transfer resistance
- vapor boilup used as an input.

In the identification of the column, the feed composition (*xf*) and flow rate (*feed*) were assumed to be the disturbances, and the manipulated variables were the vapor boilup rate (*vbr*) and either the reflux rate (*rr*) or the distillate rate (*dist*). Since the level in the condenser drum was held constant, the value of the distillate rate determined the value of the reflux rate and *vice versa*. Various ERN models were used to predict the distillate and bottoms compositions, *xd* and *xb*, respectively, the controlled variables. Patwardhan's VE model was deemed to be the "true" process, and his code was used to generate the deterministic training and test data for the ANN modeling. The operation and design parameters of the column used were identical to those used by Patwardhan.

Data for the training and data sets used to estimate the weights in the ERN were generated by making step changes in the feed variables and recording the response of the exit compositions every 60 s (the time constant in this region of operations was approximately 4000 s). Two PI controllers were used to change the manipulated variables to states such that the distillate and bottoms compositions would return to their set points after the disturbances in the feed had been introduced. Because of the relatively low volumetric hold up in the packed sections of the column and the manner in which the "true" model was implemented mathematically, there was virtually no dead time associated with the response of the outputs of the column when changes were made in any of the disturbances or manipulated inputs.

Gaussian random noise with a coefficient of variation of 0.01 for each of the four input and two output variables was added to the deterministic measurements to represent measurement noise. All of the resulting data were scaled into the range of zero to one in order to prevent scaling problems during training of the ERN, and also to insure a fair influence of each of the output variables when the objective function was computed. To give the precision needed, the final sizes of the training and test sets comprised 8333 data points each.

Because Patwardhan's model (which represented the actual column well) took too long to solve for the purpose of control, one of his simpler approximate models developed for control (the Z model) comprised of one ordinary differential equation and three nonlinear algebraic equations was compared to various ERN. The result of control using the ERN shown below are for a model with 2 recurrent inputs to the hidden nodes, 15 hidden nodes, and a total of 197 weights. This particular model represented the best balance between representing the steady state and unsteady state data simultaneously. In practice it is possible that a network which modeled the dynamics very well and the steady state very poorly (or vice versa) could be selected over a network which modeled both the dynamics and steady state moderately well.

Fig. 17 shows the response of one column output, the distillate composition, to two changes in the setpoint. Some slight steady state offset occurred with the ERN network for the bottoms composition as shown in Fig. 18 as might be expected to occur in practice because of mismatch between the model used by the controller and the actual process (e.g. the parameters for the mass transfer equations will not be exactly correct).

To reduce the steady state offset, a simple error feedback scheme was added to the model predictive controller. Use of the Z and ANN models for MPC was also compared when measured disturbances were introduced into the feed flowrate. Fig. 19 shows that

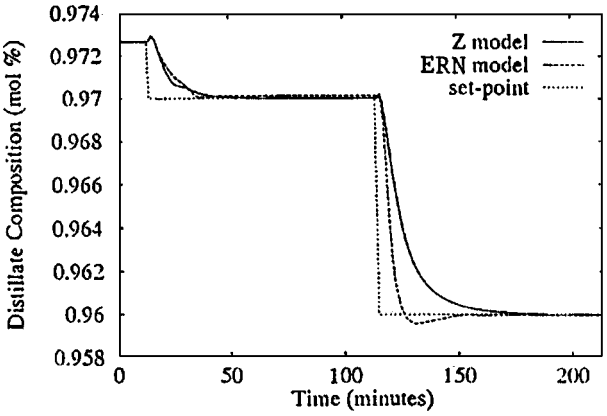

Fig. 17. Response of the distillate composition under MPC to setpoint changes using the Z and ERN models.

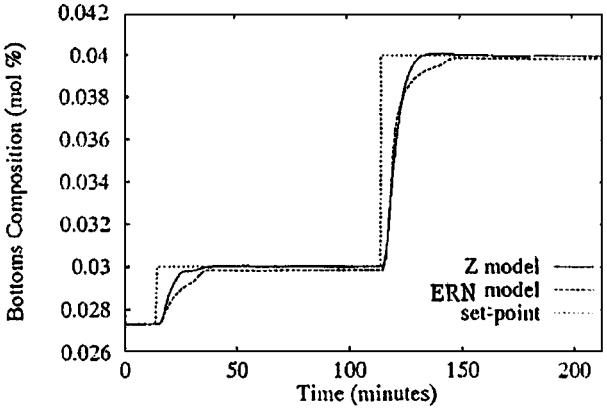

Fig. 18. Response of the bottoms composition under MPC to setpoint changes using the Z and ERN models.

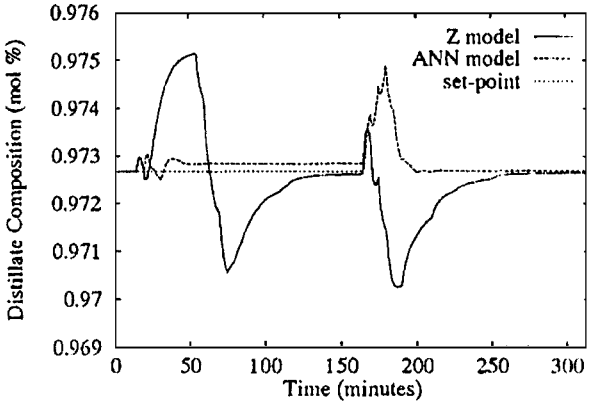

Fig. 19. Response of the distillate composition for MPC using the Z and ERN models after step disturbances in the feed flowrate.

the ERN model (with the error feedback scheme implemented) provided substantially better control performance than did the Z model.

WHAT ARE THE FUTURE PROSPECTS FOR ANN?

Many competing types of models exist for process modeling besides ANN. A number of the advantages and disadvantages of using ANN have been discussed in previous sections. For processes too complex to be modeled by a first principles model, not well understood, or that take too long to model with various empirical models, an ANN model might be a very effective choice for a model. Because ANN work best when interpolating, the need to collect suitable data and the time required to train the nets represent the two disadvantages in using ANN. On the other hand, in principle ANN would involve less model mismatch for a real process and would reduce the computation time to predict outputs from inputs. Some process design and operation simulators, such as those by Pavilion Technologies, Inc., already incorporate ANN based on the argument that an engineer can model a process quickly from the process data alone. Probably the arrival of more validated software that use ANN as part of a state and/or dynamic process flowsheeting code will be needed if practicing engineers are to use ANN extensively in the next decade.

REFERENCES

Barton, R., "Analysis and Rectification of Data from Dynamic Chemical Processes via Artificial Neural Networks," Ph.D. Dissertation, Univ. of Texas, Austin, TX (1996).

Billings, S. A. and Voon, W. S. F., "Correlation Based Model Validity Tests for Nonlinear Models," *Intl. J. Control*, **44**, 235 (1986).

Baum, E. B. and Haussler, D., "What Size Net Gives Valid Generalization?," *Neural Comput.*, **1**, 151 (1988).

Chen, S., Billings, S. A., Cowan, C. F. N. and Grant, P. M., "Nonlinear System Identification using Radial Basic Functions," *Intl. J. Systems Sci.*, **21**, 2513 (1990).

Cybenko, G., "Approximation by Superpositions of a Sigmoidal Function," *Math. Control Signal Syst.*, **2**, 303 (1987).

Draper, N., "Straight Line Regression when Both Variables are Sub-

jects to Error," Proceed. 1991 Kansas State Univ. Conf. Appld. Statistics in Agriculture, 1 (1991).
Elman, J. L., "Finding Structure in Time," *Cognitive Science*, **14**, 179 (1990).
Fiesler, E., "Handbook of Neural Computation," Oxford Univ. Press, N.Y. (1996).
Franke, R., "Convergence Properties of Radial Basis Functions," *Constr. Approx.*, **4**, 243 (1988).
Fine, T. L., "Feed Forward Neural Network Methodology," Springer, N.Y. (1999).
Fukuoka, Y., Matsuki, H., Minamitani, H. and Iskida, A., "A Modified Backpropagation Method to Avoid False Local Minima," *Neural Networks*, **11**, 1059 (1998).
Geman, S., Bienonstock, E. and Doursat, R., "Neural Networks and the Bias/Variance Dilema," *Neural Computation* **4**, 1 (1992).
Hertz, J. A., Krogh, A. S. and Palmer, R. G., "Introduction to the Theory of Neural Computation," Addison Wesley (1991).
Jang, S. S., Joseph, B. and Mukai, H., "Comparison of Two Approaches to One-line Parameters and State Estimation of Nonlinear Systems," *Ind. Engng. Chem. Process Dev.*, **25**, 809 (1986).
Kamruzzaman, J., Kumagai, Y. and Hikitu, H., "Study on Minimal Net Size, Convergence Behavior and Generalization Ability of Heterogeneous Backpropagation Network," *Artificial Neural Networks*, **2**, 203 (1992).
Karjala, T. W., "Dynamic Data Rectification via Recurrent Neural Networks," Ph.D. Dissertation, Univ. of Texas, Austin, TX (1995).
Kay, J. W. and Titterington, D. M., "Statistics and Neural Networks," Oxford Univ. Press, Oxford (2000).
Kim, I. W., Liebman, M. J. and Edgar, T. F., "Robust Error in Variables Estimation using Nonlinear Programming," *AIChE Journal*, **36**, 405 (1990).
Kulawski, G. J. and Brdys, N. A., "Stable Adaptive Control with Recurrent Networks," *Automatica*, **36**, 5 (2000).
Lee, C. C., Chung, P. C., Tsai, J.-R. and Chang, C. I., "Robust Radial Basic Function Networks," *IEEE Trans. SMC-Part B, Cybernetics*, **29**, 674 (1999).
Lee, T. C., "Structure Level Adaptation for Artificial Neural Networks," Kluwer Acad. Publ. (1991).
Lippmann, R. P., "An Introduction to Computing with Neural Nets," *IEEE ASSP Magn*, **4**, 4 (1987).
Ljung, L., "System Identification, Theory for the User," Prentice Hall (1987).
Ljung, L. and Sjoberg, J., "A System Identification Perspective on Neural Nets," *Proceed. 1992 IEEE Workshop on Neural Nets for Signal Processing*, IEEE (1992).
MacMurray, J. C., "Modeling and Control of a Packed Distillation Column using Artificial Neural Networks," M.S. Thesis, Univ. of Texas, Austin, TX (1993).
Moody, J., "Prediction Risk and Architecture Selection for Neural Networks," in From Statistics to Neural Networks, eds. V. Cherkassky, J. H. Friedman, and H. Wechsler, Springer Verlag, Berlin, 147 (1994).
Patwardhan, A. A., "Modeling and Control of a Packed Distillation Column," Ph.D. Dissertation, Univ. of Texas, Austin, TX (1991).
Psichogious, D. and Unger, L., "A Hybrid Neural Network First Principles Approach to Process Modeling," *AIChE Journal*, **38**, 1499 (1992).
Reed, R., "Pruning Algorithms-A Survey," *IEEE Trans. Neural Net.*, **4**, 740 (1993).
Rumerhart, D. E. and McClelland, J. L., "Parallel Distributed Processing," MIT Press, **1** (1986).
Seinfeld, J. H., "Optimal Stochastic Control of Nonlinear Systems," *AIChE J.*, **16**, 1016 (1970).
Sjoberg, J. and Ljung, L., "Overtraining, Regularization, and Searching for a Minimum in Neural Networks," in Proceed. IFAC Symp. Adaptive Systems in Control and Signal Processing, IFAC, 669 (1992).
Soderstrom, T., "Identification of Stochastic Linear Systems in Presence of Input Noise," *Automatica*, **17**, 713 (1981).
Suewatanakal, W., "A Comparison of Fault Detection and Classification using ANN with Traditional Methods," Ph.D. Dissertation, Univ. of Texas, Austin, TX (1993).
van de Laar, P. and Heskes, T., "Pruning Using Parameter and Neuronal Matrices," *Neural Computation*, **11**, 977 (1999).
Werbos, P. J., "Backpropagation through Time: What it Does and How to Do It," *Proceed. IEEE*, **78**, 1550 (1990).

APPENDIX A

Aldrich, C. and Deventer, J., "Artificial Intelligence Methods for Process Monitoring and Modelling of Metallurgical Plants," Proceedings of PSE '94, 1297 (1994).
Aguiar, H., and Filho, R., "Modeling and Optimization of Pulp and Paper Processing Using Neural Networks," *Computers Chem. Engng*, **22**, S981-S984.
Aldrich, C. and Deventer, J., "Modelling of Induced Aeration in Turbine Aerators by Use of Radial Basis Function Neural Networks," *The Canadian Journal of Chemical Engineering*, **73**, 808 (1995).
Aldrich, C., Moolman, D. and Deventer, J., "Monitoring and Control of Hydrometallurgical Processes With Self-Organizing and Adaptive Neural Net Systems," *Computers in Chemical Engineering*, **19**, S803 (1995).
Alvarez, E., Riverol, C., Correa, J. M. and Navaza, J. M., "Design of a Combined Mixing Rule for the Prediction of Vapor-Liquid Equilibria Using Neural Networks," *Industrial Engineering Chemical Resource*, **38**, 1706 (1999).
Alvarez, E., Riverol, C., Navaza, J. M., "Control of Chemical Processes using Neural Networks: Implementation in a Plant for Xylose Production," *ISA Transactions*, **38**, 375 (1999).
Aly, A. and Peralta, R., "Optimal Design of Aquifer Cleanup Systems under Uncertainty using a Neural Network and a Genetic Algorithm," *Water Resources Research*, **8**, 2523 (1999).
Anderson, J. S., Kevrekidis, I. G. and Rico-Martinez, R., "A Comparison of Recurrent Training Algorithms for Time Series Analysis and System Identification," *Computers in Chemical Engineering*, **20**, S751 (1996).
Arena, P., Baglio, S., Castorina, C., Fortuna, L. and Nunnari, G., "A Neural Architecture to Predict Pollution in Industrial Areas," Proceedings of ICNN Wash. D.C. June 2, 1996, 2107 (1996).
Baratti, R., Vacca, G. and Servida, A., "Neural Network Modeling of Distillation Columns," *Hydrocarbon Processing*, June, 35 (1995).
Barsamian, A. and Macias, J., "Inferential Property Predictors using Neural Networks," *Hydrocarbon Processing*, October, 107 (1998).
Barton, R. S. and Himmelblau, D. M., "On-Line Prediction of Poly-

mer Product Quality in an Industrial Reactor Using Recurrent Neural Networks," Proc. International Neural Networks, Houston, TX June, 1997, 111 (1997).

Baughman, D., "Neural Networks in Bioprocessing and Chemical Engineering," Ph.D. Dissertation, Virginia Polytechnic Institute and State University (1995).

Bawazeer, K. and Zilouchian, A., "Prediction of Products Quality Parameters of a Crude Fractionation Section of an Oil Refinery Using Neural Networks," Proceedings of ICNN Houston, 157 (1997).

Bhat, N., Minderman, P. and McAvoy T., "Use of Neural Nets for Modeling of Chemical Process Systems," IFAC Dynamics and Control of Chemical Reactors (DYCORD 89), 169 (1989).

Biegler-Konig, F. and Barmann, F.,"Neuronal Networks for Data Correlating Illustrated for the Example of a Distillation Column," *Chem. Eng. Tech.*, **63**, 64 (1991).

Birky, G. and McAvoy, T., "A Neural Net to Learn the Design of Distillation Controls," IFAC Dynamics and Control of Chemical Reactors (DYCORD 89), 205 (1989).

Bittanti, S. and Piroddi, L., "Nonlinear Identification and Control of a Heat Exchanger: A Neural Network Approach," *Journal of Franklin Institute*, **3343**, 135 (1997).

Blum, J., Villard, P., Leuba, A., Karjala, T. and Himmelblau, D. M., "Practical Issues in Applying Artificial Neural Network for Identification in Model Based Predictive Control," Proc. World Conf. Neural Networks, San Diego, CA, June, **II**, 135 (1994).

Blum, J., Villard, P., Leuba, A., Karjala, T. and Himmelblau, D. M., "Practical Issues in Applying Artificial Neural Networks for Identification in Model Based Predictive Control," Proceed. World Conference on Neural Networks, San Diego, CA, June (1994).

Bochereau, L., Bourgine, P., Bouyer, F. and Muratet, G., "Using Recurrent Multilayer Neural Networks for Simulating Batch Reactors," Proceedings of IJCNN Singapore, 1510 (1991).

Bogdan, S., Gosak, D. and Vasic-Racki, D., "Mathematical Modeling of Liquid-Liquid Equilibria in Aqueous Polymer Solution Containing Neutral Proteinase and Oxytetracycline Using Artificial Neural Network," *Computers in Chemical Engineering*, **19**, S791 (1995).

Bohr, H., "Neural Networks for Protein Structure Prediction," In Scientific Applications of Neural Nets, J. W. Clark, ed., Springer, Berlin, 189 (1999).

Borges, L. and Castier, M., "Evaluation of the Thermodynamic Models *Uniquac* and *Unifac* Using Artificial Neural Networks," Proceedings of 1993 International Joint Conference on Neural Networks, 999 (1993).

Brambilla, A. and Trivella, F., "Estimate Product Quality with ANNs," *Hydrocarbon Processing*, Sept., 61 (1996).

Bravi, M., Chianese, A., Mustacchi, C., Sed, M. and Zhang, J., "A Neural Network for Crystallizer Dynamics," Chaos Fractals Chem. Eng., G. Biardi ed., World Scien. Publ. Singapore, 284 (1995).

Bulsari, A. and Palosaari, S., "System Identification of an Adsorption Process Using Neural Networks," Proceedings of Ad Chem-Kyoto 1994, 53 (1994).

Bulsari, A. and Saxen, H., "Application of Feed-forward Neural Networks for System Identification of a Biochemical Process," Proceedings of IJCNN Singapore 1991, 1222 (1991).

Bulsari, A. and Saxen, H., "Implementation of Heuristics for Chemical Reactor Selection in a Feed-Forward Neural Network," Artificial Neural Networks, T. Kohoren ed., Elsevier, 435 (1991).

Bungay, H. and Clark, J., "Neural Network for Classifying Flow in a Tank," *Chemical Engineering Communications*, **125**, 105 (1993).

Burns, J. A., "Application of Feed-forward Neural Networks to Problems in Chemistry," PhD. Dissertation, Harvard University (1993).

Cai, S. and Toral, H., "Flow Rate Measurement in Air-Water Horizontal Pipeline by Neural Networks," Proceedings of 1993 International Joint Conference on Neural Networks, 2013 (1993).

Cai, S., Toral, H., Qiu, J. and Archer, J., "Neural Network Based Objective Flow Regime Identification in Air-Water Two Phase Flow," *The Canadian Journal of Chemical Engineering*, **72**, 440 (1994).

Calderon, Z., Espuna, A. and Puigjaner, L., "Waste Analysis and Minimization in Batch and Semibatch Reactor Operation Through Dynamic Simulation and Neural Networks," *Computers Chem. Engng.*, **22**, S977 (1998).

Chen, F. Z. and Wang, X. Z., "Software Sensor Design Using Bayesian Automatic Classification and Back-Propagation Neural Networks," *Ind. Eng. Chem. Res.*, **37**, 3985 (1998).

Chen, Y. and Himmelblau, D. M., "Determination of Nonlinear Dynamic Model Order by False Nearest Neighbor Method," Proceed. World Congress on Neural Networks, Washington, DC, July (1995).

Cheng, Y. and Himmelblau, D. M., "Identification of Nonlinear Process with Dead Time by Recurrent Neural Netwroks," Proceed. 1995 Amer. Control Conf., Seattle, WA, June (1995).

Cheng, Y., Karjala, T. W. and Himmelblau, D. M., "Closed Loop Nonlinear Process Identification Using Internally Recurrent Neural Nets," *Neural Networks*, **10**, 573 (1997).

Cheng, Y., Karjala, T. W. and Himmelblau, D. M., "Identification of Nonlinear Dynamic Processes with Unknown and Variable Dead Time Using An Internal Recurrent Neural Network," *Ind. Eng. Chem. Res.*, **34**, 1735 (1995).

Cheng, Y., Karjala, T. W. and Himmelblau, D. M., "Resolving Problems in Closed Loop Nonlinear Process Identification Using IRN," *Computers Chem. Engnr.*, **20**, 1159 (1996).

Cheng, Y., Karjala, T. W. and Himmelblau, D. M., "Identification of Nonlinear Dynamic Processes with Unknown and Variable Deat Time Using an Internal Recurrent Neural Network," *Ind. Eng. Chem. Res.*, **54**, 1735 (1995).

Chessari, C., McKay, B., Agamennoni, O., Barton, G. and Romagnoli, J., "The Application of Neural Networks in the Development of an Online Model for a Semi-Regenerative Catalytic Reformer," Proceedings of World Congress on Neural Networks-San Diego 1994, I-173 (1994).

Christo, F. C., Masri, A. R. and Nebot E. M., "Utilising Artificial Neural Network and Repro-Modelling in Turbulent Combustion," Proceedings of ICNN, 1995, 911 (1995).

Cristina, H., Aguiar, L. and Filho, R. M., "Modeling and Optimization of Pulp and Paper Processes using Neural Networks," *Computers Chem. Engng.*, **22**, S981 (1998).

Dutta, P. and Rhinehart, R., "Application of Neural Network Control to Distillation and an Experimental Comparison with Other Advanced Controllers," *ISA Transactions*, **38**, 251 (1999).

Fakhr-Eddine, K., Cabassud, M., Duverneuil, P., Lann M. V. and Le, C., "Use of Neural Networks for LPCVD Reactors Modelling," *Computers chem. Engng.*, **20**, S521 (1996).

Fileti, A. and Pereira, A. F., "Adaptive and Predictive Control Strategies for Batch Distillation: Development and Experimental Testing," *Computers chem. Engng.*, **21**, S1227 (1997).

Fujiwara, Takeshi, Sotowa, Kenichiro, Hashimoto, Iori, Torimoto, Yoshifumi, Nakamura, J. and Tsuto, K., "Development of Operation Support System for a Hydrolysis Reactor of Fat or Oil - A Neural Network Approach," Proceedings of PSE 1991 Montebello, Quebec, 9.1 (1991).

Fullana, M., Trabelsi, F. and Recasens, F., "Use of Neural Net Computing for Statistical and Kinetic Modelling and Simulation of Supercritical Fluid Extractors," *Chemical Engineering Science*, **55**, 79 (2000).

Gemmingen, U., "Neuronal Networks for Data Correlation, Illustrated for the Example of a Distillation Column," *Chem.-Ing.-Tech.*, **63**, 64 (1991).

Gonzalez-Garcia, R., "Identification of Distributed Parameter Systems: A Neural Net Based Approach," *Computers chem. Engng.*, **22**, S965 (1998).

Grondin-Perez, B., Defaye, G., Regnier, N. and Caralp, L., "Modelisation par Reseaux de Neurones et Conduite de Reacteurs Chimiques," *Prog. Gence Procedes*, **9**, 39 (1995).

Gu, Z., Lam H. and Dhurjati, P., "Feature Correlation Method for Enhancing Fermentation Development: A Comparison of Quadratic Regression With Artificial Neural Networks," *Computers chem. Engng.*, **20**, S407 (1996).

Guimaraes, P. R. B. and McGreavy, C., "Flow of Information Through an Artificial Neural Network," *Computers chem. Engng.*, **19**, S741 (1995).

Hadjuski, M., Elenkov, G., Hadjuski, L. and Mohadjer, A., "Neural Network Based Steady-State Observation and Reference Control of Distillation Columns," Proceedings of IFAC Symposium on Dynamics and Control of Chemical Reactors, Distillation Columns and Batch Processes College Park, MD 1992, 393 (1992).

Hanzevack, E. L., Long, T. W. and Menousek, J. F., "Neural Nets Speed Flow Calculation," *Chem. Eng. Commun.*, 41 (1993).

Himmelblau, D. M and Hsiung, J. T., "Development of Control Strategies via Artificial Neural Networks and Reinforcement Learning," Proceed 1991 ACC, Boston, MA. (1991).

Himmelblau, D. M., "Fault Detection and Diagnosis Via Neural Networks," Neural Networks in Process Engineering. Eds. G A. Montague and M. J. Willis, IOP Publ., Philadelphia, PA (1994).

Himmelblau, D. M., "Fault Detection in Heat Exchangers," Proceed Amer. Control Conf. Chicago, IL, June 23-26 (1992).

Himmelblau, D. M., "Neural Networks and Chemical Engineering," Proceedings of 15th Intl. Conf. Chemical Engineering Industry, Paris, France (1989).

Himmelblau, D. M., "Rectification of Data in Dynamic Processes Using Artificial Neural Networks," Proc. Conf. Process Systems Engr., Kyongju, Korea, May, 623 (1994).

Himmelblau, D. M., "Use of Artificial Neural Networks to Monitor Faults and For Troubleshooting in the Process Industries," Proceed. IFAC Conference on On-Line Fault Detection and Supervision in the Chem. Process Industries, University of Delaware, Newark DE, April, 22-24 (1992).

Himmelblau, D. M. and Bhalodia, M., "On-Line Sensor Validation of Single Sensors Using Artificial Neural Network," Proceedings of American Control Conference, Seattle, WA, June (1995).

Himmelblau, D. M. and Bhalodia, M., "On-Line Validation of Single Sensors Using Artificial Neural Networks," Proceed 1995 Amer. Control Conference, Seattle, WA, June (1995).

Himmelblau, D. M. and Karjala, T. W., "Rectification of Data in a Dynamic Process Using Artificial Neural Networks," *Computers Chem. Engnr.*, **20**, 805 (1996).

Himmelblau, D. M., Barker, R. W. and Suewatanakul, W., "Fault Classification With the Aid of Artificial Neural Networks," Proceed IFAC Conf. Fault Detection, Supervision & Safety, Baden-Baden, Sept. 10-13 (1991).

Hoskins, J. C. and Himmelblau, D. M., "Artificial Neural Network Models of Knowledge Representation in Chemical Engineering," *Computers and Chem. Engr.*, **12**, 610 (1988).

Hoskins, J. C. and Himmelblau, D. M., "Automatic Process Control Using Reinforcement Learning in Artificial Neural Networks," Proceed. 1st Annual Meet. Intl. Neural Network Soc. (1988).

Hoskins, J. C. and Himmelblau, D. M., "Fault Detection and Diagnosis Using Artificial Neural Networks," Artificial Intelligence in Process Engineering, Ed. M. Mavrovouniotis, Academic Press, NY (1991).

Hoskins, J. C. and Himmelblau, D. M., "Fault Detection and Diagnosis Using Artificial Neural Networks," Artificial Intelligence in Process Engineering, Academic Press, San Diego, CA (1992).

Hoskins, J. C. and Himmelblau, D. M., "Fault Detection via Artificial Neural Networks," Computer Applns. For the Chemical Industry, R. Eckermann, ed., Dechema, Frankfurt, **116**, 227 (1989).

Hoskins, J. C. and Himmelblau, D. M., "Process Control via Artificial Neural Networking and Reinforcement Learning," *Computers Chem. Engng.*, **16**, 241 (1992).

Hoskins, J. C. and Himmelblau, D. M., "Automatic Process Control Using Reinforcement Learning in Artificial Neural Networks," Proceedings of First Annual Meeting International Neural Network Society, Boston, MA (1988).

Hoskins, J. C. and Mavrovouniotis, M. L., "Fault Detection and Diagnosis Using Artificial Neural Networks," Artificial Intelligence in Process Engineering, Academic Press, 123 (1990).

Hoskins, J. C., Kaliyur, J. C. and Himmelblau, D. M., "Fault Diagnosis in Complex Chemical Plants Using Artificial Neural Networks," *AIChE Journal.*, **37**, 137 (1991).

Hoskins, J. C., Kaliyur, K. M. and Himmelblau, D. M., "Incipient Fault Detection and Diagnosis Using Artificial Neural Networks," Proceedings of IJCNN90, San Diego (1990).

Huang, Y. L., Edgar, T. F., Himmelblau, D. M. and Trachtenberg, I., "Constructing A Reliable Neural Network Model For A Plasma Etching Process Using Limited Experimental Data," *IEEE Trans., Semiconductor Manufacturing*, **7**, 333 (1994).

Hwarng, H. and Chong, C., "ART-based Control Chart Pattern Recognizers," Proceedings of WCNN Korea, 1994, I-496 (1994).

Iordache, O., Corriou, J. P., Garrido-Sanchez, L., Fonteix, C. and Tondeur, D., "Neural Network Frames Application to Biochemical Kinetic Diagnosis," *Computers chem. Engng.*, **17**, 1101 (1993).

Ishida, M. and Zhan, J., "A Policy-and Experience-Driven Neural Network (Penn) and its Application to SISO and MIMO Process Control," Proceedings of International Joint Conference on Neural Networks, 681 (1993).

Ishida, M. and Zhan, J., "Neural Model-Predictive Control of Distributed Parameter Crystal Growth Process," *AIChE Journal*, **41**, 2333 (1995).

Jaramillo-Mendez, N., Embid-Droz, S., Fernandez-Rodriguez, B. and Lake, L., "Estimation of Capillary Pressure and Relative Permeability Using Artificial Neural Networks," Proceedings Annie, 1998, **8**,

521 (1998).

Jeong, S. and Lee, K. S., "A Study on Interpolating Behavior of Neural Networks for Nonlinear Engineering Problems," *HWAHAK KONGHAK*, **31**, 54 (1993).

Joseph, B. and Hanratty, F., "Predictive Control of Quality in a Batch Manufacturing Process Using Artificial Neural Network Models," *Ind. Eng. Chem. Res.*, **32**, 1951 (1993).

Kargupta, H. and Ray, S., "Temporal Sequence Processing Based on the Biological Reaction-Diffusion Process," Proceedings WCNN Korea, 1994, 150 (1994).

Karim, M. N. and Rivera, S. L., "Application of Neural Networks in Bioprocess State Estimation," Proceedings of ACC 1992, WA14, 495 (1992).

Karjala, T. and Himmelblau, D. M., "Dynamic Data Rectification Using the Extended Kalman Filter and Recurrent Neural Networks," Proc. IEEE Intl. Conf. Neural Networks, Orlando, FL, June 25, 3244 (1994).

Karjala, T. W. and Himmelblau, D. M., "Dynamic Rectification of Data via Recurrent Neural Nets and the Extended Kalman Filter," *AIChE Journal*, **42**, 2225 (1996).

Karjala, T. W. and Himmelblau, D. M., "Dynamic Rectification of Process Measurements Containing Either Uncorrelated or Autocorrelated Errors Using Externally Recurrent Networks," Proceed. First Intl. Chemometrics Internet Conference, August (1994).

Karjala, T. W. and Himmelblau, D. M., "Estimation of Measurement Biases Using Recurrent Neural Networks and the Extended Kalman Filter," Proceed. World Congress Neural Networks, Washington DC, July (1995).

Karjala, T. W. and Himmelblau, D. M., "Dynamic Data Rectification by Recurrent Neural Networks vs Traditional Methods," *AIChE J.*, **40**, 1865 (1994).

Karjala, T. W. and Himmelblau, D. M., "Estimation of Measurement Biases Using Recurrent Neural Networks and Extended Kalman Filter," Proceed. of International Federation of Automatic Control, **M265**, San Francisco, CA (1996).

Karjala, T. W., Himmelblau, D. M. and Miikkulainen, R., "Data Rectification Using Recurrent (Elman) Neural Networks," Proceed. Intl. Joint Conference on Neural Networks, Baltimore, MD, June 8-10 (1992).

Keeler, J., "Prediction and Control of Chaotic Chemical Reactions Via Neural Network Models," Proceedings of Conference on Artificial Intelligence in Petroleum Exploration and Production Plano, TX May 1993, 1 (1993).

Keeler, J., "Vision of Neural Networks and Fuzzy Logic for Prediction and Optimization of Manufacturing Processes," *SPIE*, **1709**, 447 (1992).

Keller, P., Kouzes, R. and Kangas, L., "Neural Network Based Chemical Sensor Systems for Environmental Monitoring," Proceedings of WCNN Korea 1994, 269 (1994).

Keshavaraj, R., Tock, R. and Narayan, R., "Prediction of Solvent Activity in Polymer Systems with Neural Networks," *Ind. Eng. Chem. Res.*, **34**, 3974 (1995).

Kiparissides, C. and Morris, J., "Intelligent Manufacturing of Polymers," **20**, S1113 (1996).

Koiranen, T., Virkki-Hatakka, T., Kraslawski, A. and Nystrom, L., "Hybrid, Fuzzy and Neural Adaptation in Case-Based Reasioning System for Procase Equipment Selection," *Computers chem. Engng.*, **22**, S997 (1998).

Kraslawski, A., Pedrycz, W., Koiranen, T. and Nystrom, L., "Hybrid Knowledge-Based Tool for Mixing Equipment Selection- the Fuzzy Neural Network," *Chem. Eng. Technol.*, **19**, 233 (1996).

Krishnapura, V. and Jutan, A., "ARMA Neuron Networks for Modeling Nonlinear Dynamical Systems," *The Canadian Journal of Chemical Engineering*, **75**, 574 (1997).

Krothapally, M. and Palanki, S., "A Neural Network Strategy for End-Point Optimization of Batch Processes," *ISA Transactions*, **38**, 383 (1999).

Kuroda, C., "Uses of Neural Networks-Data Processing for Nonlinear Processes," *Kagaku Kogaku*, **59**, 408 (1995).

Kuroe, Y. and Kimura, I., "Modeling of Unsteady Heat Conduction Field by Using Composite Recurrent Neural Networks," Proceedings of ICNN, 1995, 323 (1995).

Kurtanjek, Z., "Modeling and Control by Artificial Neural Networks in Biotechnology," *Computers chem. Engng.*, **18**, S627 (1994).

Lambert, J. and Hecht-Nielsen, R., "Application of Feedforward and Recurrent Neural Networks to Chemical Plant Predictive Modeling," Proceedings of IJCNN Seattle IEEE, I-373 (1991).

Langonnet, P., "Process Control with Neural Networks: an Example," *SPIE*, **1709**, 468 (1992).

Latrille, E., Corrieu, G. and Thibault, J., "Neural Network Models for Final Process Time Determination in Fermented Milk Production," *Computers chem. Engng.*, **18**, 1171 (1994).

Lee, J., Yum, C. and Kim, W., "Neural Network Based Judgmental Adjustment for Time Series Forecasting," Proceedings of EANN Helsinbu August 1995, 299 (1995).

Lei Jia, J., "The Model Reference Adaptive Control Based on the Genetic Algorithm," Proceedings of ICNN Houston, 1997, 783 (1995).

Lei, J., He, G. and Jiang, J., "The State Estimation of the CSTR System Based on a Recurrent Neural Network Trained by HGAs," Proceedings of IEEE, 779 (1997).

Lin, J. and Jang, S., "Nonlinear Dynamic Artificial Neural Network Modeling Using an Information Theory Based Experimental Design Approach," *Ind. Eng. Chem. Res.*, **37**, 3640 (1998).

Lin, J., Jang, S., Shieh, S. and Subramaniam, M., "Generalized Multivariable Dynamic Artificial Neural Network Modeling for Chemical Processes," *Ind. Eng. Chem. Res.*, **38**, 4700 (1999).

Liu, H., Xiaoming, X. and Zhang, Z., "CMAC-Based Learning Control for a Batch Reactor," Proceedings of IFAC Triennial World Congress San Francisco 1996, 385 (1996).

Loh, A., Looi, K. and Fong, K., "Neural Network Modelling and Control Strategies for a pH Process," *J. Proc. Control*, **5**, 355 (1995).

MacMurray, J. C. and Himmelblau, D. M., "Modeling and Control of a Packed Distillation Column Using Artificial Neural Networks," *Computers Chem. Engnr.*, **19**, 1077 (1995).

Madison, M., "Application of Neural Networks to Structure Elucidation," PhD. Dissertation, Arizona State University (1992).

Martinetz, T. and Poppe, T., "A Neural Network Approach to Estimating Material Properties," Proceedings WCNN Korea, 1994, 541 (1994).

Martinez, E. and Wilson, J., "A Hybrid Neural Network-First Principles Approach to Batch Unit Optimisation," *Computers chem. Engng.*, **22**, S893 (1998).

Martinez, R., Anderson, J. and Kevrekidis, I., "Self-Consistency in Neural Network-Based NLPC Analysis with Applications to Time-

Series Processing," *Computers chem. Engng*., **20**, S1089 (1996).
Matsumoto, H., Kuroda, C., Palosaari, S. and Ogawa, K., "Neural Network Modeling of Serum Protein Fractionation Using Gel Filtration Chromatography," *Journal of Chemical Engineering of Japan*, **32**, 1 (1999).
McAnany, D., "Practical Applications of Artificial Neural Networks in Chemical Process Development," Proceedings of ISA 1993, Paper #93-412, 1721 (1993).
McAvoy, T. and Pan, D., "Application of Olfactory Neural Networks to Chemical Processes," Proceedings of PSE 1994, 1025 (1994).
Meert, K. and Rijckaert S., "Intelligent Modelling in the Chemical Process Industry With Neural Networks: a Case Study," *Computers Chem. Engng*., **22**, S587 (1998).
Megan, L. and Cooper, D., "Pattern Recognition Based Adaptive Control of Two-Input/Two-Output Systems Using ART2-A Neural Networks," *Ind. Eng. Chem. Res*., **33**, 1510 (1994).
Meghlaoui, A., Thibault, J., Bui, R., Tikasz, L. and Santerre, R., "Neural Networks for the Identification of the Aluminum Electrolysis Process," *Computers Chem. Engng*., **22**, 1419 (1998).
Milanic, S., Hvala, N., Strmcnik, S. and Karba, R., "A Semi-Empirical and a Neural Network Model of a Batch Plant Dynamics for Flexible Recipes Control," DYCOPS-5 Greece June 1998, 292 (1998).
Montague, G., Tham, M., Willis, M. and Morris, A., "Predictive Control of Distillation Columns Using Dynamic Neural Networks," Proceedings of 3rd IFAC Symposium Dynamics Control Chem. Reactors, Distillation Batch Process, College Park, MD April 1992, 231 (1992).
Munsif, H. P., "Applications of Neural Networks for Distillation Control," PhD. Dissertation, Texas Tech University (1996).
Nikravesh, M., Soroush, M. and Johnston, R., "Nonlinear Control of an Oil Well," Proceedings of the American Control Conference Albuquerque NM June 1997, 739 (1997).
Noid, D., Darsey, J. and Sumpter, B., "Applications of Neural Networks in Polymer Simulations," Soc. Plastic Eng. Meeting New Orleans March 1993, 22 (1993).
Otto, M., "Neural Networks for Calibration of Unspecific Chemical Sensors," Proceedings of WCNN Portland, July 11-15, 1993, 346 (1993).
Palau, A., Delgado, A., Velo, E. and Puigjaner, L., "Use of Neural Networks for Predicting the Performance of Discontinuous Gas-Solid Chilling Machines," *Computers chem. Engng*., **20**, S297 (1996).
Park, J. K., "Modeling of Distillation Column and Reactor Dynamics Using Artificial Neural Networks," PhD Dissertation. Northwestern University (1993).
Petersen, R., Fredenslund, A. and Rasmussen, P., "Artificial Neural Networks as a Predictive Tool for Vapor-Liquid Equilibrium," *Computers chem. Engng*., S63 (1994).
Piron, E., Latrille, E. and Rene, F., "Application of Artificial Neural Networks for Crossflow Microfiltration Modelling: "Black-Box" and Semi-Physical Approaches," *Computers chem. Engng*., **21**, 1021 (1997).
Prevost, P., Isambert, A., Depeyre, D., Donadille, C. and Perisse, R., "Practical Insights Into Neural Network Implementation in Metallurgical Industry," *Computers chem. Engng*., **18**, 1157 (1994).
Qian, Y., "Artificial Neural Network for Dynamic Simulation of a CTMP Process," Proceedings of the 3rd International Conference on Fuzzy Logic Iizuka Japan, August, 1994, 107 (1994).
Qin, S. and Rajagopal, B., "Combining Statistics and Expert Systems with Neural Networks for Empirical Process Modeling," Proceedings of ISA 1993, 93 (1993).
Raju, G. and Cooney, C., "Active Learning From Process Data," *AIChE J*., **44**, 2199 (1998).
Ramasamy, S. and Deshpande, P., "Consider Neural Networks for Process Identification," Hydrocarbon Processing, 59-62 June (1995).
Ramchandran, B., "Neural Network Model-Based Control of Distillation Columns," PhD. Dissertation, Texas Tech University (1994).
Ramchandran, S. and Rhinehart, R., "A Very Simple Structure for Neural Network Control of Distillation," *J. Proc. Cont*., **5**, 115 (1995).
Reddy, V., "Analysis of Plant Measurements Through Input-Training Neural Networks," *Computers chem. Engng*., **20**, S889 (1996).
Rehbein, D., Maze, S. and Havener, J., "The Application of Neural Networks in the Process Industry," *ISA Transactions*, **31**, 7 (1992).
Reifman, J., Vitela, J., Feldman, E. and Wei, T., "Recurrent Neural Networks for NO_x Prediction in Fossil Plants," Proceedings of Society of Computer Simulation, April (1993).
Reuter, M., "A Generalized Neural-Net Kinetic Rate Equation," *Chemical Engineering Science*, **48**, 1281 (1993).
Riddle, A., Bhat, N. and Hopper, J., "Neural Networks Help Optimize Solvent Extraction," Hydrocarbon Processing, Nov., 45 (1998).
Rivals, I., Personnaz, P., Dreyfus, G. and Ploix, J., "Modelisation, Classification et Commande Par Reseaux de Neurones: Principes Fondamentaux, Methodologie de Conception et Illustrations Industrielles," *Prog. Genie Procedes*, **9**, 1 (1995).
Rivera, S., "Neural Networks and Micro-Genetic Algorithms for State Estimation and Optimization of Bioprocesses," PhD Dissertation, Colorado State University (1992).
Roj, E. and Wilk, M., "Simulation of an Absorption Column Performance Using Feed-Forward Neural Networks in Nitric Acid Production," *Computers chem. Engng*., **22**, S909 (1998).
Savkovic-Stevanovic, J., "Neural Networks in Dynamic Simulation and Control of Chemical Systems," Proceed EANN Helsinki 1995, 489 (1995).
Savkovic-Stevanovic, J., "A Neural Network Model for Analysis and Optimization of Processes," *Computers Chem. Engng*., **47**, S411 (1993).
Savkovic-Stevanovic, J., "Neural Networks For Process Analysis and Optimization: Modeling and Applications," *Computers Chem. Engng*., **18**, 1149 (1994).
Saynatjoki, P. and Hammarstrom, L., "Identification of a Pilot Plant Reactive Distillation Process Using RBF Networks," 4th IFAC Symposium Proc. Dyn. Control Chem React. Distill. Col. Batch Processes, 105 (1995).
Schenker, B. and Agarwal, M., "Predictive Control of a Bench-Scale Chemical Reactor Based on Neural-Network Models," *IEEE Transactions*, **6**, 388 (1998).
Schenker, B. and Agarwal, M., "State Prediction for Chemical Reactors Using Feedback Neural Networks," DYCORD Maryland 1992, 69 (1992).
Schenker, B. and Agarwal, M., "State Prediction for Chemical Reactors Using Feedback Neural Networks," 3rd IFAC Symposium on Dynamics and Control of Chemical Reactors Distillations and Batch Processes, 171 (1992).
Schmitz, G. and Aldrich, C., "Neurofuzzy Modeling of Chemical Pro-

cess Systems With Ellipsoidal Radial Basis Function Neural Networks and Genetic Algorithms," *Computers chem. Engng*., **22**, S1001 (1998).

Shahrokhi, M. and Pishvaie, M., "Artificial Neural Network Feedforward/Feedback Control of a Batch Polymerization Reactor," Proceedings of the American Control Conference June 1994, 3391 (1998).

Shene, C., Diez, C. and Bravo, S., "Neural Networks for the Prediction of the State of Zymomonas Mobilis CP4 Batch Fermentations," *Computers and Chem. Engnr*., **23**, 1097 (1999).

Shimizu, Y., "Multi-Objective Optimization for Site Location Problems through Hybrid Genetic Algorithm with Neural Networks," *Journal of Chemical Engineerng Japan*, **32**, 51 (1999).

Song, J. and Park, S., "Neural Model Predictive Control for Nonlinear Chemical Processes," *J. Chem. Eng. Jpn*., **26**, 347 (1993).

Souza, M., "Neural Net Based Model Predictive Control of a Chaotic Continuous Solution Polymerization Reactor," Proceedings of International Joint Conference on Neural Networks 1993, 1777 (1993).

Su, H., Fan, L. and Schlup, J., "On-Line Determination of the Degree of Cure of Epoxy/Graphite Composites With Neural Networks," ISPE Snowmass July 1995, 1 (1995).

Suewatanakul, W. and Himmelblau, D. M., "Fault Detection via Artificial Neural Network," *Engineering Simulation*, **13**, 967 (1996).

Terry, P. A. and Himmelblau, D. M. "Date Rectification and Gross Error Detection in a Steady State Process Via Artificial Neural Networks," *Ind. Eng. Chem. Res*., **32**, 3020 (1993).

Tholudur, A. and Ramirez, W., "Neural-Network Modeling and Optimization of Induced Foreign Protein Production," *AIChE J*., **45**, 1660 (1999).

Tojima, M., Suzuki, T., Kobayashi, G. and Minami, Y., "Pattern Recognition of Microstructures Using Neural Networks," *Tetsu to Hagane*, **80**, 551 (1994).

Tsai, C. and Chang, C., "Dynamic Process Diagnosis Via Integrated Neural Networks," *Computers Chem. Engng*., **19**, S747 (1995).

Tsaptsinos, D. and Leigh, J., "A Step by Step Approach for the Construction of a Fermentation Process Estimator," Proceedings WCNN Portland, July 11-15, 1993, 216 (1993).

Tsen, A., Jang, S., Wong, D. and Joseph, B., "Predictive Control of Quality in Batch Polymerization Using Hybrid ANN Models," *AIChE J*., **42**, 455 (1996).

Van Deventer, J., "The Tracking of Changes in Chemical Processes Using Computer Vision and Self-Organizing Maps," Proceedings of ICNN, 1995, 3068 (1995).

Van Deventer, J., "Visualisation of Plant Disturbances Using Self-Organising Maps," *Computers Chem. Engng*., **20**, S1095 (1996).

Wang, H. and Yoon, E., "Adaptive Control of Nonlinear Chemical Processes Using Neural Networks," DYCOPS-5 Cortu Greece IFAC, 286 (1998).

Wang, H., Oh, Y. and Yoon, E., "Strategies for Modeling and Control of Nonlinear Chemical Processes Using Neural Networks," *Computers chem. Engng*., **22**, S823 (1998).

Watanabe, K., Hirota, S., Hou, L. and Himmelblau, D. M., "Diagnosis of Multiple Simultaneous Faults Via Hierarchal Artificial Neural Networks," *AIChE Journal*, **40**, 839 (1994).

Watanabe, K., Matsuura, I., Abe, M., Kubota, M. and Himmelblau, D. M. "Incipient Fault Diagnosis of Chemical Processes Via Artificial Neural Networks," *AIChE J*., **35**, 1803 (1989).

Whaley, A., Bode, C., Ghosh, J. and Eldridge, R., "HETP and Pressure Drop Prediction for Structured Packing Distillation Columns Using a Neural Network Model," Proceedings of Annie., **8**, 669 (1998).

Willis, M., Massimo, C., Montague, G., Tham, M. and Morris, A., "Artificial Neural Networks in Process Engineering," *IEE Proceedings Part D*., **138**, 256 (1991).

Wilson, J. and Martinez, E., "Neuro-Fuzzy Modeling and Control of a Batch Process Involving Simultaneous Reaction and Distillation," *Computers chem. Engng*., **21**, S1233 (1997).

Wong, I., Lam, D., Storey, A., Fong, P. and Swayne, D., "A Neural Network Approach to Predict Missing Environmental Data," Proceedings WCNN Korea, 1994, 490 (1994).

Xiarong, H., Xiaoguang, Z. and Bingzhen, C., "On-Line Estimation of Vapour Pressure of Stabilized Gasoline via ANNs," Proceed. EANN Helsinki August 1995, 609 (1995).

Yang, S., Wang, J. and Wang S., "Optimal Blending System of Gasoline Basing Artificial Neural Network," Proceedings of PSE 1994, 1021 (1994).

Zbicinski, I., Strumillo, P. and Kaminski, W., "Hybrid Neural Model of Thermal Drying in a Fluidized Bed," *Computers Chem. Engng.*, **20**, S695 (1996).

Zexin, H., Xiwen, L. and Huaqing, M., "Adaptive Observer and Noninear Control Strategy for Chemical Reactors via Neural Networks," *Huagong Xuebao*, **46**, 144 (1995).

Zhang, B. and May, G., "Towards Real-Time Fault Identification in Plasma Etching Using Neural Networks," Proceedings ANNIE 1998, **8**, 803 (1998).

Zhang, J. and Morris, A., "Fuzzy Neural Networks for Nonlinear Systems Modelling," *IEE Proc-Control Theory*, **142**, 551 (1995).

Zhang, J., Martin, E., Morris, A. and Kiparissides, C., "Inferential Estimation of Polymer Quality Using Stacked Neural Networks," *Computers Chem. Engng.*, **21**, S1025 (1997).

Zhang, J., Morris, A. and Martin, E., "Long-term Prediction Models Based on Mixed Order Locally Recurrent Neural Networks," *Computers chem. Engng*., **22**, 1051 (1998).

Zhao, H. and McAvoy, T., "Modeling of Activated Sludge Wastewater Treatment Processes Using Integrated Neural Networks and a First Principle Model," 1996 IFAC 13 Triennial World Congress San Francisco, 455 (1996).

Zhong, W. and Yu, J., "Modeling Jet Fuel Endpoint Using Neural Networks with Genetic Learning," *Hydrocarbon Processing*, 77, Dec. (1999).

Zhou, J., Zhihong, X., Leming, S., Zhang, Y. and Cai, S., "Application of Artificial Neural Networks in Prediction of Material Properties and Problem Diagnosis of Chemical Process," *Huagong Yejin*, **14**, 57 (1993).

Zhu, J., Zhengzhi, H., Xiao, J. and Rao, M., "Modeling Liquid Phase Oxygenation Process Using Neural Network," Proceedings of ICNN, Houston, 1997, 844 (1997).