Autoassociative Neural Networks for Robust Dynamic Data Reconciliation

Shuanghua Bai, David D. McLean, and Jules Thibault
Dept. of Chemical Engineering, University of Ottawa, Ottawa, Canada K1N 6N5

DOI 10.1002/aic.11080
Published online January 3, 2007 in Wiley InterScience (www.interscience.wiley.com).

Reliable estimation of process variables for plant monitoring and control is an important topic that has been studied extensively. The Kalman filter has often been used and has acquired an enviable reputation. However, use of the Kalman filter suffers from two restrictive conditions: (1) it requires state-space models and (2) it has to be tuned online to achieve its best performance. Recently, an alternative methodology based on dynamic data reconciliation has been proposed to overcome the first restriction. Although the approach of dynamic data reconciliation can incorporate any form of model, it involves online optimization that may require long computation time for complex systems. This report explores a new methodology based on a combination of autoassociative neural networks (AANNs) and dynamic data reconciliation that overcomes the need for online tuning, as required by the Kalman filter, as well as online optimization as required by conventional dynamic data reconciliation methods. Simulation examples of a distillation column demonstrate that the AANN-based dynamic data reconciliation approach is capable of effectively attenuating measurement noise and is robust to changes of the noise level in plant measurements and the loss of measurements. © 2007 American Institute of Chemical Engineers *AIChE J,* 53: 438–448, 2007
Keywords: process variable estimation, autoassociative neural network (AANN), dynamic data reconciliation, Kalman filter

Introduction

A critical concern in plant operation is estimation of the current state of a dynamic process. Unfortunately, process measurements often contain some degree of inaccurate information, such as measurement noise. The presence of measurement noise not only prevents plant operators from identifying true values of process variables, but also inhibits the performance of automatic control systems. To cope with this problem, the model-based Kalman filter has been most widely used.[1] The Kalman filter uses stochastic state-space process and measurement models. In its discrete form it is represented by

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) + \mathbf{w}_{t-1} \tag{1}$$

Correspondence concerning this article should be addressed to J. Thibault at thibault@genie.uottawa.ca.

$$\mathbf{y}_t = \mathbf{g}(\mathbf{x}_t) + \boldsymbol{\varepsilon}_t \tag{2}$$

where $\mathbf{x}_t$ is a vector of process state variables at time t, $\mathbf{y}_t$ is a vector of measured process variables, and $\mathbf{u}_{t-1}$ is a vector of process input variables. $\mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ is a functional vector representing process models. $\mathbf{g}(\mathbf{x}_t)$ is a functional vector representing measurement models that reduces to $\mathbf{x}_t$ if all the state variables are directly measured. $\mathbf{w}_{t-1}$ is a vector of random variables denoting white Gaussian process model noise, that is, $\mathbf{w}_{t-1} \sim N(\mathbf{0}, \mathbf{S})$. $\boldsymbol{\varepsilon}_t$ is a vector of random variables representing white Gaussian measurement noise, that is, $\varepsilon_t \sim N(\mathbf{0}, \mathbf{V})$. For the problem of process estimation defined by Eqs. 1 and 2, the optimal estimates, $\hat{\mathbf{x}}_t$, using the Kalman filter are given by

$$\hat{\mathbf{x}}_t = \hat{\mathbf{y}}_t + \mathbf{K}_t\left(\mathbf{y}_t - \mathbf{C}_t\hat{\mathbf{y}}_t\right) \tag{3}$$

where $\hat{\mathbf{y}}_t$ is the a priori estimate (that is, the vector of model predicted values) given by

$$\hat{\mathbf{y}}_t = \mathbf{f}(\hat{\mathbf{x}}_{t-1}, \mathbf{u}_{t-1}) \tag{4}$$

$\mathbf{K}_t$ is the gain of the Kalman filter, recursively calculated by

$$\mathbf{P}_t^- = \mathbf{A}_t \mathbf{P}_{t-1} \mathbf{A}_t^T + \mathbf{S} \tag{5}$$

$$\mathbf{K}_t = \mathbf{P}_t^- \mathbf{C}_t^T \left(\mathbf{C}_t \mathbf{P}_t^- \mathbf{C}_t^T + \mathbf{V}\right)^{-1} \tag{6}$$

$$\mathbf{P}_t = \mathbf{P}_t^- - \mathbf{K}_t \mathbf{C}_t \mathbf{P}_t^- \tag{7}$$

In Eqs. 5–7, $\mathbf{A}_t$ is a Jacobian matrix obtained by linearizing $\mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ with respect to $\mathbf{x}_{t-1}$ at the realization $\mathbf{x}_{t-1} = \hat{\mathbf{x}}_{t-1}$, $\mathbf{C}_t$ is a Jacobian matrix obtained by linearizing $\mathbf{g}(\mathbf{x}_t)$ at the realization $\mathbf{x}_t = \hat{\mathbf{y}}_t$. $\mathbf{P}_t^-$ and $\mathbf{P}_t$ are the covariance matrices of a priori and a posteriori estimation errors with initial values assumed for $\mathbf{P}_t$ (such as $\mathbf{P}_0 = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix).

The Kalman filter is a time-variant system. However, when $\mathbf{f}(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$ and $\mathbf{g}(\mathbf{x}_t)$ are linear in terms of the state variables, the gain of the filter $\mathbf{K}_t$ reaches a constant value. At this point the Kalman filter becomes time invariant. The most attractive advantage of the Kalman filter lies in its optimality in the sense of providing minimum variance estimates. However, the optimality of the Kalman filter requires two restrictive prerequisites: linear state-space models and independent white Gaussian noise for both process and measurements. Usually, it is difficult to determine the statistical properties of the process model noise, $\mathbf{w}_{t-1}$. As a result, elements in the covariance matrix of $\mathbf{w}_{t-1}$ are commonly viewed as tuning parameters rather than measurable constants in the implementation of the Kalman filters.[2]

In addition to the Kalman filter, another approach-dynamic data reconciliation (DDR)-has been developed in recent years for process variable estimation.[3–8] Mathematically, the problem of process estimation by the DDR was initially formulated as the following constrained least-squares optimization problem:

$$\begin{aligned} &\text{Minimize} \quad J(\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_t) = \sum_{i=0}^{t} \left[\left(\mathbf{y}_i - \hat{\mathbf{x}}_i\right)^T \mathbf{V}^{-1} \left(\mathbf{y}_i - \hat{\mathbf{x}}_i\right) \right] \\ &\text{subject to} \quad \mathbf{f}\left[\frac{d\hat{\mathbf{x}}}{dt}, \hat{\mathbf{x}}\right] = 0 \end{aligned} \tag{8}$$

where $\mathbf{f}$ represents a functional vector of constraints consisting of dynamic models. To solve the optimization problem for data reconciliation, the differential equations were discretized or integrated and then the DDR problem was converted to minimize the objective function constrained by algebraic equations. Subsequently, the optimization was solved analytically or by nonlinear programming (NLP). The dimensionality of the dynamic optimization problem defined by Eq. 8 increases linearly with the number of sampling periods because all measurements from time zero are considered. To reduce the magnitude of the online optimization problem, a moving window in the time horizon, $[t - H, t]$, has normally been used. At time t, only the measurements within the moving window were reconciled and only the estimates for the current time were used for process monitoring and/or control. A critical assumption in DDR, as formulated by Eq. 8, is that the models represent the true dynamics of the process and the estimates (reconciled values) exactly satisfy the model constraints. However, no mathematical model perfectly represents a real plant; consequently, model mismatch should be taken into account in the data reconciliation algorithm. Furthermore, large-scale online optimization using NLP may require excessively long computation time.

Accordingly, Bai et al.[9] proposed using autoassociative neural networks (AANNs) to perform DDR. An AANN for a dynamic process was used and trained offline and then implemented online in a simple process: a storage tank process. The training objective function consisted of simultaneously minimizing the weighted sum of squared measurement and model errors. Dynamic process models with various structures can be encapsulated into the training objective function. They demonstrated that an AANN can be directly implemented online to perform dynamic data reconciliation after its offline training. It neither requires online tuning nor explicit calculation of model predictions, as required in the Kalman filter, and it does not require online dynamic optimization as in the conventional DDR algorithm.

The objectives of this report are to develop methodologies of using AANN to perform dynamic data reconciliation for multivariable complex processes, where it is impossible or difficult to use phenomenological models encapsulated in the training objective function. We explore two architectures of AANNs-feedforward AANN and recurrent AANN-for dynamic processes, and develop their training strategies. Online performance of the AANN and its robustness to changes in measurement noise level and faults are evaluated. The performance of AANN is compared to that of the Kalman filter.

The structure of this article is outlined as follows. In the next section, the general concept of dynamic data reconciliation for process estimation is presented, considering errors that inevitably arise in both process measurements and models. Then an AANN-based algorithm is developed to perform dynamic data reconciliation. In the third section, the performance of the AANN-based DDR algorithm is evaluated and compared to that of the Kalman filter by simulation of multivariable control of a binary distillation column. Finally, the fourth section summarizes this article and draws some conclusions.

AANN-based DDR Algorithm

Formulation of the dynamic data reconciliation problem

In view of the uncertainties in both the process model and process measurements, a compromise is achieved by simultaneously minimizing the weighted sum of squared errors of process measurements and model, that is,

$$\text{Minimize } J(\hat{\mathbf{x}}_t) = \left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right)^T \mathbf{V}^{-1} \left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right) + \mathbf{f}^T(\hat{\mathbf{x}}_t) \mathbf{\Omega}^{-1} \mathbf{f}(\hat{\mathbf{x}}_t) \tag{9}$$

where $\mathbf{f}(\hat{\mathbf{x}}_t)$ is a functional vector of process models in discrete form and $\mathbf{\Omega}$ is the covariance matrix of model errors. The dynamic data reconciliation problem formulated by Eq. 9 requires online optimization at each sampling time. For complex processes, a large computation time may be needed. To overcome this problem, using an AANN trained with respect to Eq. 9 to perform the data reconciliation is suggested here.

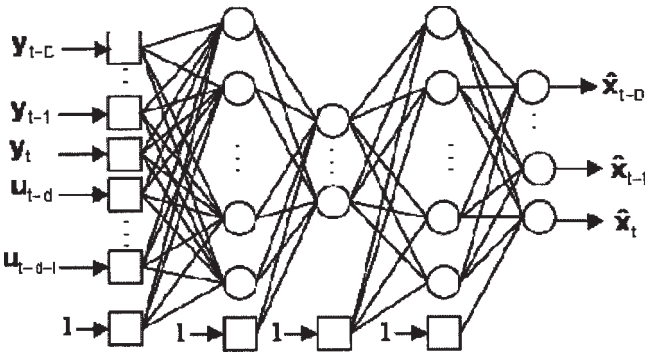

Figure 1. Architecture of feedforward autoassociative neural network (AANN) for a dynamic process.

D denotes the order of state variable, I represents the order of input variable, and d the number of time delays for the input variable.

Development of AANN-based DDR algorithm

An AANN is a network whose structure is similar to that of a conventional feedforward neural network. It normally consists of an input layer, three hidden layers, and an output layer. The input and the output layers have an identical number of neurons (excluding the bias in the input layer). The first and third hidden layers, designated *mapping* and *demapping* layers, respectively, contain a relatively larger number of neurons. The second hidden layer, referred to as the *bottleneck* layer, usually contains a lower number of neurons. The transfer functions for the mapping and demapping layers are nonlinear functions (such as sigmoid functions), whereas the transfer function for the bottleneck and output layers can be linear or nonlinear. The key feature of AANN is its data compression/regeneration by the bottleneck layer. The input, mapping, and bottleneck layers compress the input information to a lower dimension, after which the demapping and output layers recover the main underlying features of the original information. The compression/regeneration process enables the network to represent the input information in a compressed form so that it can often reveal the essence of the data. An AANN has to be trained to produce its desired outputs. In network training, the input and the target vectors presented to the network are identical and the objective function, consisting of the mean squared errors between the network outputs and its inputs, is minimized. After successful training, an AANN can be regarded as a filter used to perform steady-state data reconciliation.[10,11]

For dynamic data reconciliation, a similar architecture, shown in Figure 1, can be used to take advantage of the noise-reduction characteristic generated by the compression/regeneration of the AANN. In the dynamic case, the neural network is not fully autoassociative because additional past inputs of the process, $\mathbf{u}_{t-d},\ldots,\mathbf{u}_{t-d-I}$, necessary to capture its dynamic behavior, are required such that the vector of inputs is not identical to the vector of outputs. However, the neural network of Figure 1 is autoassociative with respect to the output variables.

Process models are encapsulated within the structure of the network to perform data reconciliation. The number of neurons in the input and output layers is determined by the structure of the process models. In Figure 1, D represents the order of the models, whereas $d,\ldots, d+I$ represent the time delays of the input variables. However, the number of neurons in the hidden layers has to be determined by trial and error. The objective function to train the dynamic AANN to perform data reconciliation can be written as

$$\text{Minimize } J(\theta) = \frac{1}{N}\sum_{t=D}^{N}\Big[\left(\mathbf{y}_{t-D}-\hat{\mathbf{x}}_{t-D}\right)^T\mathbf{V}^{-1}\left(\mathbf{y}_{t-D}-\hat{\mathbf{x}}_{t-D}\right)+\cdots$$
$$+\left(\mathbf{y}_t-\hat{\mathbf{x}}_t\right)^T\mathbf{V}^{-1}\left(\mathbf{y}_t-\hat{\mathbf{x}}_t\right)+\mathbf{f}^T(\hat{\mathbf{x}}_t,\ldots,\hat{\mathbf{x}}_{t-D},\mathbf{u}_{t-d},\ldots,\mathbf{u}_{t-d-I})\,\boldsymbol{\Omega}^{-1}$$
$$\times\,\mathbf{f}(\hat{\mathbf{x}}_t,\ldots,\hat{\mathbf{x}}_{t-D},\mathbf{u}_{t-d},\ldots,\mathbf{u}_{t-d-I})\Big] \qquad (10)$$

where θ is the vector composed of all connection weights (parameters) between the neurons in the network and N is the total number of data points in the training horizon. Among the training algorithms, backpropagation is the simplest method. However, more advanced training algorithms such as quasi-Newton and Levenberg–Marquardt methods are most often used for faster convergence. After the training, the performance of the network is validated using a fresh data set. Then the AANN can be applied online to directly perform data reconciliation.

In the AANN-based DDR algorithm formulated by Eq. 10, at each sampling time t, the raw measurements within the time window $[t-D, t]$ are reconciled. However, only the reconciled values for the current time $\hat{\mathbf{x}}_t$ are retained for process monitoring and/or control purposes. Other reconciled values for past measurements are discarded. As a result, the dimensionality of the data reconciliation problem becomes large. To overcome this problem, the architecture of the dynamic AANN is modified as presented in Figure 2. The output of the network, $\hat{\mathbf{x}}_t$, delayed a number of times, is fed back to the input layer such that both process temporal and spatial patterns are incorporated in the structure of the recurrent AANN. For this case, the neural network is autoassociative only for the current output variables, and obviously only the current measurements are reconciled at each sampling time.

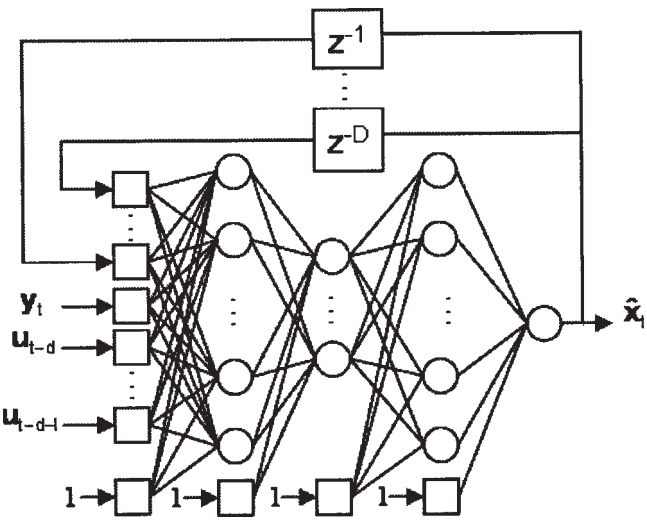

Figure 2. Architecture of recurrent AANN for a dynamic process.

z^{-1} denotes the backshift operator such that $z^{-1}\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t-1}$.

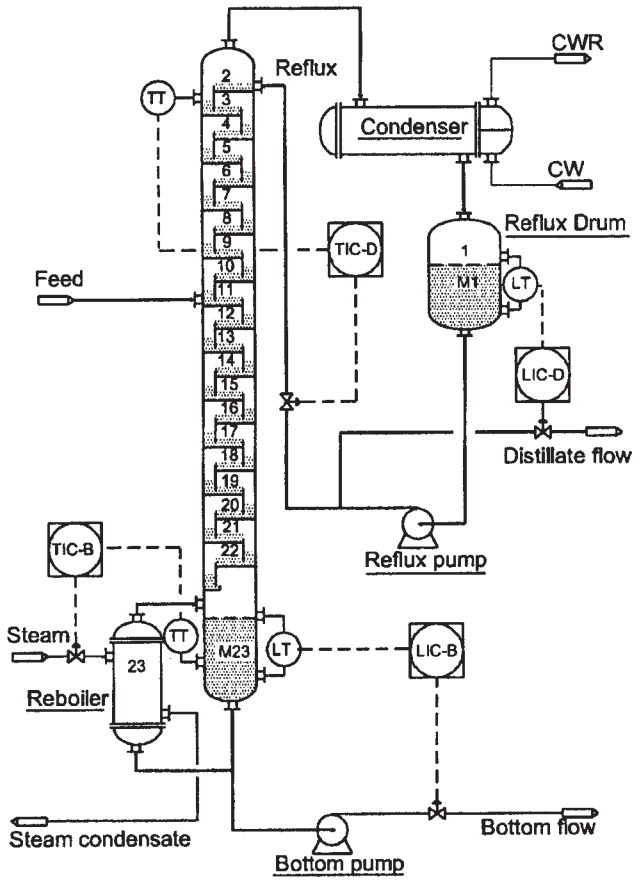

Figure 3. Control scheme of a binary distillation column.

By means of the recurrent architecture of the AANN, the training objective function for data reconciliation becomes

$$\text{Minimize } J(\theta) = \frac{1}{N}\sum_{t=D}^{N}\Big[\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right)^T \mathbf{V}^{-1}\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right) + \mathbf{f}^T\left(\hat{\mathbf{x}}_t, \ldots, \hat{\mathbf{x}}_{t-D}, \mathbf{u}_{t-d}, \ldots, \mathbf{u}_{t-d-I}\right)\boldsymbol{\Omega}^{-1} \times \mathbf{f}\left(\hat{\mathbf{x}}_t, \ldots, \hat{\mathbf{x}}_{t-D}, \mathbf{u}_{t-d}, \ldots, \mathbf{u}_{t-d-I}\right)\Big] \quad (11)$$

In the network training, it is important to note that the output vector, $\hat{\mathbf{x}}_t$, fed back to the input layer is not known initially; that is, the values of $\hat{\mathbf{x}}_{t-D},\ldots,\hat{\mathbf{x}}_{t-1}$ as inputs to the network are unknown. Accordingly, an iterative training methodology is developed to manage this problem. To begin the training, the values for $\hat{\mathbf{x}}_{t-D},\ldots,\hat{\mathbf{x}}_{t-1}$ are assigned the values of their associated raw measurements, $\mathbf{y}_{t-D},\ldots,\mathbf{y}_{t-1}$. Then, the network is trained with respect to Eq. 11 until satisfactory convergence is met. After the first iteration, the network output, $\hat{\mathbf{x}}_t$, is delayed a number of times and fed back as inputs $\hat{\mathbf{x}}_{t-D},\ldots,\hat{\mathbf{x}}_{t-1}$, and then the network is trained again. After several iterations, the feedback vectors and the objective function will not change, indicating the training of the neural network has been completed. Comparing the two training objective functions for the feedforward and recurrent AANNs, it appears that in the feedforward AANN, past and current measurements are reconciled, providing more degrees of freedom in minimizing the objective function such that process models impose less restriction on the raw measurements. On the contrary, in the recurrent AANN, only the current measurements are reconciled so that the raw measurements can be constrained more tightly by the process models. The recurrent structure of the AANN presented in Figure 2 is expected to be more effective for dynamic data reconciliation than the feedforward structure presented in Figure 1. This statement is verified in the next section.

The covariance matrix of the model residuals ($\boldsymbol{\Omega}$) in the training objective function is often difficult to determine. Consequently, $\boldsymbol{\Omega}$ is considered diagonal and the elements in $\boldsymbol{\Omega}$ are treated as tuning parameters in network training. If elements of $\boldsymbol{\Omega}$ are given larger numbers relative to the variances of the raw measurements, meaning that more confidence is put on the raw measurements, the outputs of the network (that is, the reconciled values) will be close to the raw measurements. On the other hand, if the elements of $\boldsymbol{\Omega}$ are given relatively smaller values, then more confidence is given to the models and, as a result, model mismatch may bias the reconciled values. The matrix $\boldsymbol{\Omega}$ provides a measure for offline tuning for the AANN to perform dynamic data reconciliation. The AANN-based DDR provides the advantage that it does not need online tuning as required in the Kalman filter. Furthermore, the use of AANN-based DDR is straightforward without explicitly calculating model predictions or performing online dynamic optimization because information about process dynamics has been encoded into the neural network during its offline training.

An Illustrative Example

The AANN-based DDR algorithm developed in this work was evaluated through simulation of multiloop control of a binary (benzene/toluene) distillation column. This distillation column was selected to demonstrate an application of the AANN-based DDR algorithm for control of a relatively complex process. The use of AANN-based DDR algorithm for simple processes with one controlled variable can be found in Bai et al.[9] The distillation column having four PI (proportional integral) control loops is presented in Figure 3. Controllers TIC-D and TIC-B are used to control the top and bottom temperatures by manipulating the reflux flow rate and the flow of steam to the reboiler, respectively. Controllers LIC-D and LIC-B are used to control the reflux drum and column base liquid levels by manipulating the distillate flow rate and the bottom product flow rate, respectively. A sampling time of 30 s was used for all the measurements. White Gaussian noise with standard deviation σ, presented in Table 1, was added to the true values of the four controlled variables to simulate their measured values. The dynamic distillation simulator, based on rigorous distillation models (that is, mass and heat balances, vapor–liquid equilibrium, and tray hydraulics) was developed by Bai.[12]

Table 1. Nominal Noise Levels of Controlled Variables for the Distillation Column

Measured Variables	Unit	σ
Reflux drum level, H_D	m	0.02
Top temperature, T_D	°C	0.25
Column base level, H_B	m	0.02
Bottom temperature, T_B	°C	0.25

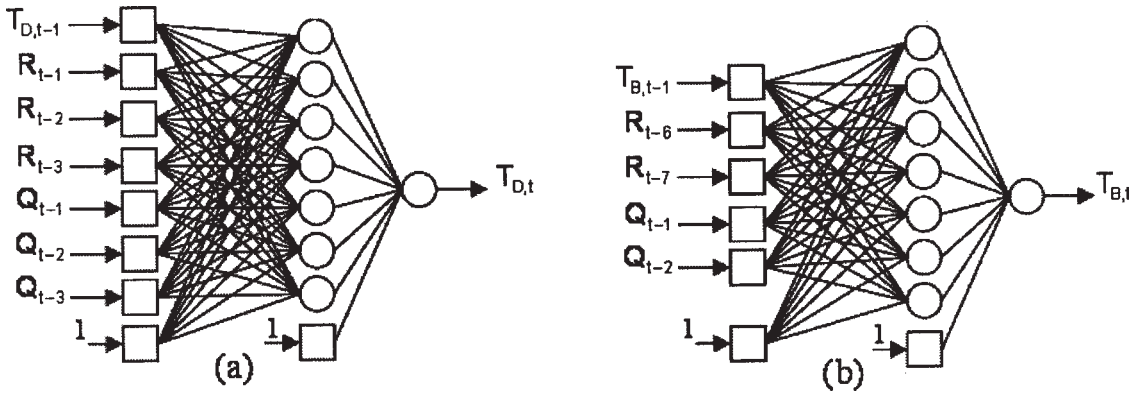

Figure 4. Feedforward neural network models.

(a) Top temperature; (b) bottom temperature.

Offline training AANNs

Dynamic Models. Because of the complexity of the distillation process, it is impractical to develop phenomenological models for use in data reconciliation. Consequently, input–output empirical models were identified for the four controlled variables. It was found that the dynamic behavior of the two liquid levels can be adequately described by linear models, whereas nonlinear models were required for the two temperatures. The nonlinearity of the two temperatures lies in the fact that the top temperature cannot decrease below the boiling point of pure benzene and the bottom temperature cannot exceed the boiling point of pure toluene under the column pressures. The linear models for the two liquid levels were given in terms of deviation variables (denoted by primes) by

$$H'_{D,t} = H'_{D,t-1} - 9.058 \times 10^{-4} D'_{t-1} - 9.058 \times 10^{-4} R'_{t-1} + 2.112 \times 10^{-5} Q'_{t-1} + 1.06 \times 10^{-5} Q'_{t-2} \quad (12)$$

$$H'_{B,t} = H'_{B,t-1} + 1.023 \times 10^{-3} R'_{t-2} - 1.152 \times 10^{-3} B'_{t-1} - 2.056 \times 10^{-5} Q'_{t-1} - 1.028 \times 10^{-5} Q'_{t-2} \quad (13)$$

where $H_{D,t}$ is the reflux drum level at sampling time t, $H_{B,t}$ is the column base level, D_t is the distillate flow rate, R_t is the reflux flow rate, B_t is the bottom flow rate, and Q_t is the reboiler heat duty. A feedforward neural network model, illustrated in Figure 4a, was developed for the top temperature, $T_{D,t}$. This neural network model can be concisely expressed as

$$T_{D,t} = f_{NN}(T_{D,t-1}, R_{t-1}, R_{t-2}, R_{t-3}, Q_{t-1}, Q_{t-2}, Q_{t-3}) \quad (14)$$

where f_{NN} represents the neural model used to predict the column top temperature at time t. A similar feedforward neural network, illustrated in Figure 4b, was developed for the column bottom temperature, $T_{B,t}$. It can be expressed as

$$T_{B,t} = f_{NN}(T_{B,t-1}, R_{t-6}, R_{t-7}, Q_{t-1}, Q_{t-2}) \quad (15)$$

Structures of AANNs. For the distillation column having multiple inputs and multiple outputs, it is possible to develop an AANN that contains all the variables collectively to perform dynamic data reconciliation. However, this approach results in an extremely large network that is difficult to handle. Consequently, individual AANN was developed for each controlled variable. The use of the recurrent AANN was considered. The architectures of the AANNs for the reflux drum level and top temperature are presented in Figure 5, where the superscript "m" for the controlled variables in the inputs of the networks denotes the raw measurements and "r" in outputs of the networks denotes the reconciled values. Akin to the reflux drum level and top temperature, the AANNs for the column base level and bottom temperature have similar architectures.

The cross-correlations of the model errors for the four controlled variables were found to be negligible. Therefore, the four objective functions corresponding to the four controlled variables used to train the four AANNs are given by the following equations:

$$J(\theta_1) = \frac{1}{N}\sum_{t=2}^{N}\left\{\frac{1}{\sigma^2_{H_D}}\left(H^m_{D,t} - H^r_{D,t}\right)^2 + \frac{1}{\sigma^2_{H_D,R}}\left[H^r_{D,t} - H^r_{D,t-1} + 9.058 \times 10^{-4}(D_{t-1} - D_0) + 9.058 \times 10^{-4}(R_{t-1} - R_0) - 2.112 \times 10^{-5}(Q_{t-1} - Q_0) - 1.06 \times 10^{-5}(Q_{t-2} - Q_0)\right]^2\right\} \quad (16)$$

$$J(\theta_2) = \frac{1}{N}\sum_{t=3}^{N}\left\{\frac{1}{\sigma^2_{T_D}}\left(T^m_{D,t} - T^r_{D,t}\right)^2 + \frac{1}{\sigma^2_{T_D,R}}\left[T^r_{D,t} - f_{NN}\left(T^r_{D,t-1}, R_{t-1}, R_{t-2}, R_{t-3}, Q_{t-1}, Q_{t-2}, Q_{t-3}\right)\right]^2\right\} \quad (17)$$

$$J(\theta_3) = \frac{1}{N}\sum_{t=2}^{N}\left\{\frac{1}{\sigma^2_{H_B}}\left(H^m_{B,t} - H^r_{B,t}\right)^2 + \frac{1}{\sigma^2_{H_B,R}}\left[H^r_{B,t} - H^r_{B,t-1} - 1.023 \times 10^{-3}(R_{t-2} - R_0) + 1.152 \times 10^{-3}(B_{t-1} - B_0) + 2.056 \times 10^{-5}(Q_{t-1} - Q_0) + 1.028 \times 10^{-5}(Q_{t-2} - Q_0)\right]^2\right\} \quad (18)$$

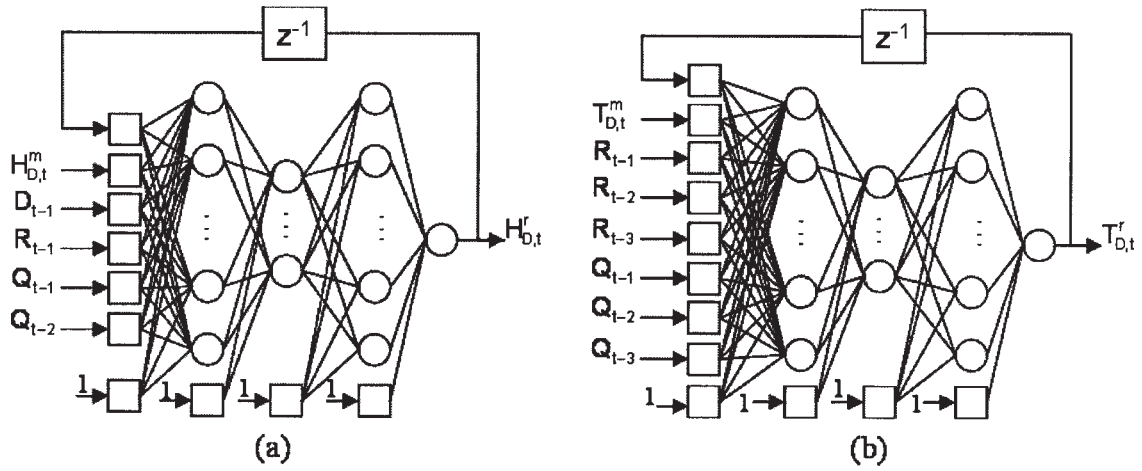

Figure 5. Architectures of recurrent AANNs for dynamic data reconciliation.

(a) Reflux drum level; (b) top temperature.

$$J(\theta_4) = \frac{1}{N}\sum_{t=7}^{N}\left\{\frac{1}{\sigma^2_{T_B}}\left(T^m_{B,t} - T^r_{B,t}\right)^2 + \frac{1}{\sigma^2_{T_B,R}}\left[T^r_{B,t} - f_{NN}(T^r_{B,t-1}, R_{t-6}, R_{t-7}, Q_{t-1}, Q_{t-2})\right]^2\right\} \quad (19)$$

where D_0, R_0, B_0, and Q_0 are steady-state values for the distillate flow rate, reflux flow rate, bottom flow rate, and reboiler heat duty, respectively. $\sigma^2_{H_D}$, $\sigma^2_{T_D}$, $\sigma^2_{H_B}$, and $\sigma^2_{T_B}$ are the associated variances of the raw measurements. $\sigma^2_{H_D,R}$, $\sigma^2_{T_D,R}$, $\sigma^2_{H_B,R}$, and $\sigma^2_{T_B,R}$ are the associated variances of the model errors.

Training Data Sets. Under open-loop operation, the distillation column was sufficiently excited by a series of random step changes in the four manipulated variables. For the distillate flow rate and bottom flow rate, the random step changes between two consecutive sampling periods had a switching probability of 0.2, whereas for the reflux flow rate and reboiler heat duty, a switching probability of 0.1 was used. Further, to effectively capture the nonlinearity of the process, the amplitudes of the random step changes had a uniform distribution around the nominal steady state. In all, 1450 data points for the responses of the four controlled variables along with the variations of the four manipulated variables were obtained and 80% of the data points were used for network training and the remaining 20% were used for network validation.

Network Training. From Figure 5 and Eqs. 16–19, it can be seen that, for each AANN, the number of neurons in the input layer was determined by the structure of the dynamic model encapsulated in the training objective function. The number of neurons in the three hidden layers for each AANN was determined by trail and error. The same number of neurons was assigned to the mapping and demapping layers. The training of the neural networks was carried out using a quasi-Newton method.

To assess the performance of each AANN, the mean squared error (MSE) between the reconciled values and the true values from the validation data set was used as the objective criterion. The MSE is defined as

$$\text{MSE} = \frac{1}{K}\sum_{t=0}^{K}(\hat{\mathbf{x}}_t - \mathbf{x}_t)^2 \quad (20)$$

where K is the total number of data points in the validation data set. It is very important to note that the calculation of the MSE criterion is achievable only in the simulated case study because the true values of the controlled variables are exactly known in the simulation, so that it is possible to compare the various algorithms and assess their performance. However, for implementation in a real plant, one could not assess the reconciliation performance using the criterion of Eq. 20, although it does not impede the use of the AANN-based DDR because the training objective functions (see Eqs. 16–19) do not use these true values. In practice, location and dispersion statistics of residuals between the reconciled and the measured values could be used in hypothesis tests as criteria to evaluate the reconciliation performance. White noise with mean zero and variance close to that of the raw measurements for the residuals would constitute a sign of good performance of the AANN-based DDR algorithm.

A group of AANNs was selected in the trial-and-error examination for each controlled variable. For the column base level and bottom temperature, the AANNs are presented in Table 2. The number of neurons in the bottleneck layer was chosen from 2 to 4, whereas the number of neurons in the mapping and demapping layers was chosen from 3 to 8. The number of neurons in each layer included a bias neuron, except for the output layer. For each AANN, it was completely trained and validated 20 times, each time with the network having different initial weights. The MSE value standardized by the associated variance of the raw measurements, MSE/σ^2, was calculated for each trial. The mean of the MSE/σ^2 values and its standard error are also presented in Table 2. For each group of AANNs having the same number of neurons in the bottleneck layer, the mean of the MSE/σ^2 values decreased, and then increased slightly, with the increased number of neurons in the mapping and demapping

Table 2. Performance of AANNs for the Column Base Level and Bottom Temperature*

Number of Neurons in Each Layer	Total Number of Neurons	Total Number of Weights	MSE/σ^2 ± SE	
			H_B	T_B
{7, 3, 2, 3, 1}	16	24	0.921 ± 0.046	0.284 ± 0.005
{7, 4, 2, 4, 1}	18	35	0.802 ± 0.043	0.275 ± 0.005
{7, 5, 2, 5, 1}	20	46	0.779 ± 0.039	0.269 ± 0.002
{7, 6, 2, 6, 1}	22	57	0.630 ± 0.026	0.273 ± 0.007
{7, 7, 2, 7, 1}	24	68	0.653 ± 0.017	0.276 ± 0.003
{7, 8, 2, 8, 1}	26	79	0.657 ± 0.023	0.281 ± 0.006
{7, 3, 3, 3, 1}	17	29	0.896 ± 0.030	0.283 ± 0.005
{7, 4, 3, 4, 1}	19	42	0.741 ± 0.056	0.275 ± 0.005
{7, 5, 3, 5, 1}	21	55	0.622 ± 0.040	0.269 ± 0.001
{7, 6, 3, 6, 1}	23	68	0.610 ± 0.053	0.270 ± 0.001
{7, 7, 3, 7, 1}	25	81	0.674 ± 0.043	0.271 ± 0.001
{7, 8, 3, 8, 1}	27	94	0.671 ± 0.031	0.272 ± 0.001
{7, 3, 4, 3, 1}	18	34	0.882 ± 0.032	0.286 ± 0.007
{7, 4, 4, 4, 1}	20	49	0.709 ± 0.051	0.275 ± 0.005
{7, 5, 4, 5, 1}	22	64	0.672 ± 0.048	0.272 ± 0.002
{7, 6, 4, 6, 1}	24	79	0.612 ± 0.031	0.269 ± 0.001
{7, 7, 4, 7, 1}	26	94	0.647 ± 0.041	0.270 ± 0.001
{7, 8, 4, 8, 1}	28	109	0.662 ± 0.027	0.271 ± 0.001

*Results are based on 20 training/validation trials. SE, standard error.

layers. Having smaller number of neurons in the mapping and demapping layers, the structures of the AANNs were insufficient for data reconciliation and consequently resulted in larger MSE/σ^2 values. On the other hand, having larger numbers of neurons in the mapping and demapping layers, the AANNs were overparameterized and, consequently, the noise attenuation performed by the AANNs was deteriorated. Similar results were obtained for the reflux drum level and top temperature. The AANNs having the structures as {7, 8, 3, 8, 1}, {9, 8, 2, 8, 1}, {7, 6, 3, 6, 1}, and {7, 6, 4, 6, 1} for the reflux drum level, top temperature, column base level, and bottom temperature, respectively, were selected for online implementation.

As mentioned, the variance of model errors provides a measure for offline tuning an AANN to perform dynamic data reconciliation. Next, the performance of the recurrent AANN selected for each controlled variable as a function of the variance of model error in the training objective function was investigated. Given a value of the variance of model error, the AANN was trained and then validated. The standardized MSE values as a function of the tuning parameter are plotted in Figure 6. The minimum MSE/σ^2 values for the four controlled variables were found at $\sigma^2_{H_D,R} = 2.5 \times 10^{-3}$, $\sigma^2_{T_D,R} = 0.36$, $\sigma^2_{H_B,R} = 1.8 \times 10^{-4}$, and $\sigma^2_{T_B,R} = 0.22$, respectively, which represent the optimal tuning parameters for the AANN-based DDR algorithm. Increasing the variance of the model error means that a greater level of confidence level was placed on the raw measurements. On the other hand, decreasing the variance of the model error means that greater confidence was placed on the process model. As shown in Figure 6, if the variance of the model error in the training objective function assumed a different value of its optimal value, a larger MSE/σ^2 value resulted.

Using the optimal variances of model errors to train the AANNs, representative samples of raw, reconciled, and true values for the four controlled variables in the network training and validation are presented in Figure 7. This figure shows that the reconciled values were very close to the true values. The noise contained in the measurements was significantly reduced after being reconciled by the neural network. Using the representative data shown in Figure 7, the residuals between the reconciled and raw measurements in the validation data set were calculated for each variable. The mean values of the residuals evaluated for the reflux drum level, top temperature, base level, and bottom temperature were −0.002 m, 0.088°C, −0.004 m, and −0.032°C, respectively. These results corroborate the results shown in Figure 7, indicating that the reconciled values are unbiased based on the analysis of the residuals.

Comparison of Recurrent AANN to Feedforward AANN. As stated in the previous section, the AANNs having recur-

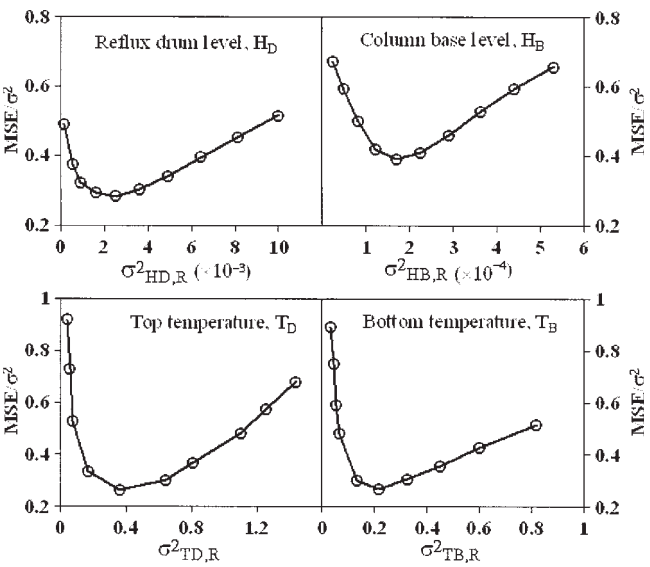

Figure 6. Values of MSE/σ^2 as a function of the variances of model errors in the training objective functions of recurrent AANNs.

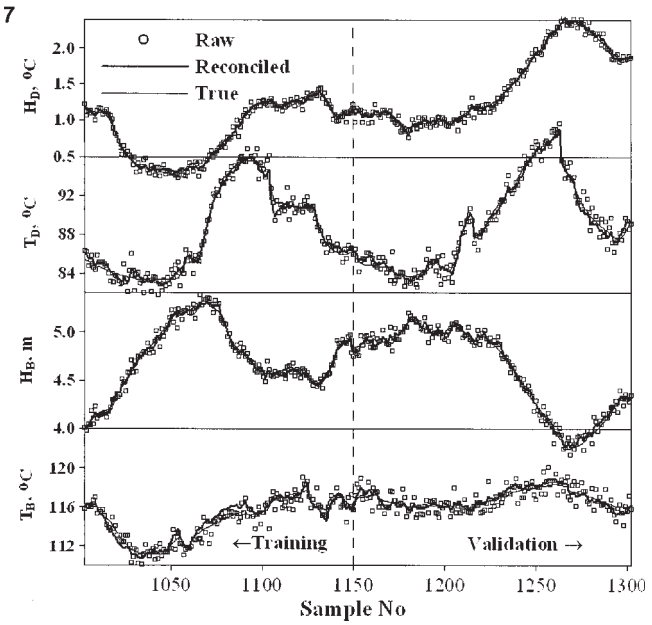

Figure 7. Representative samples of raw, reconciled, and true values for the four controlled variables in network training and validation.

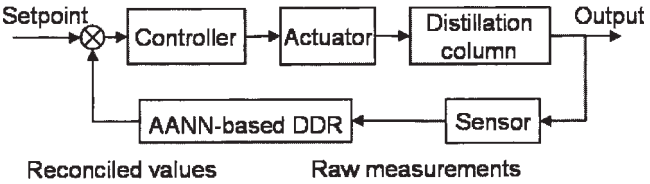

Figure 8. Feedback control scheme using AANN-based DDR.

rent structures are expected to have better performance than those having feedforward structures. This hypothesis was verified by examining the values of MSE/σ^2 obtained using each structure. For each controlled variable, a feedforward AANN was developed. It had the same number of neurons in each layer as in the recurrent AANN, except that it was necessary that the network output layer had two neurons for the reconciled values at times t and $t-1$ (see Figure 2 for the structure of a feedforward AANN). Feedforward AANNs as well as each recurrent AANN were trained and validated 20 times, each time with different initial weights. The average values of MSE/σ^2 and their standard errors obtained by the AANNs are presented in Table 3. On average, the values of MSE/σ^2 achieved by the feedforward AANNs were 1.5- to twofold larger than those achieved by the recurrent AANNs. The recurrent AANNs performed significantly better than the feedforward AANNs for dynamic data reconciliation.

Online implementations of AANNs

After the offline training, the recurrent AANNs were implanted inside PI feedback control loops in the distillation column, as illustrated in Figure 8, where the raw measurements were first reconciled using the AANNs, and then the outputs of the AANNs were used by the controllers to calculate control moves.

The closed-loop performance of the AANN-based DDR for each controlled variable was first evaluated for the nominal noise levels listed in Table 1. To obtain significant dynamic change in the process, the set point of the bottom temperature was lowered by 2.5°C at 30 min and returned to its nominal set point at 210 min, whereas the set point of the top temperature was increased by 3.0°C at 120 min and returned to its nominal value at 300 min. The raw measurements, reconciled, and true values for the four controlled variables along with the control moves are presented in Figure 9. A set-point change in either of the two temperatures had an

Table 3. Comparison of Performance of Recurrent and Feedforward AANNs*

	MSE/σ^2 ± SE			
AANN	H_D	T_D	H_B	T_B
Feedforward	0.44 ± 0.01	0.49 ± 0.01	0.61 ± 0.01	0.56 ± 0.01
Recurrent	0.27 ± 0.02	0.25 ± 0.01	0.40 ± 0.05	0.27 ± 0.01

*Results are based on 20 training/validation trials. SE, standard error.

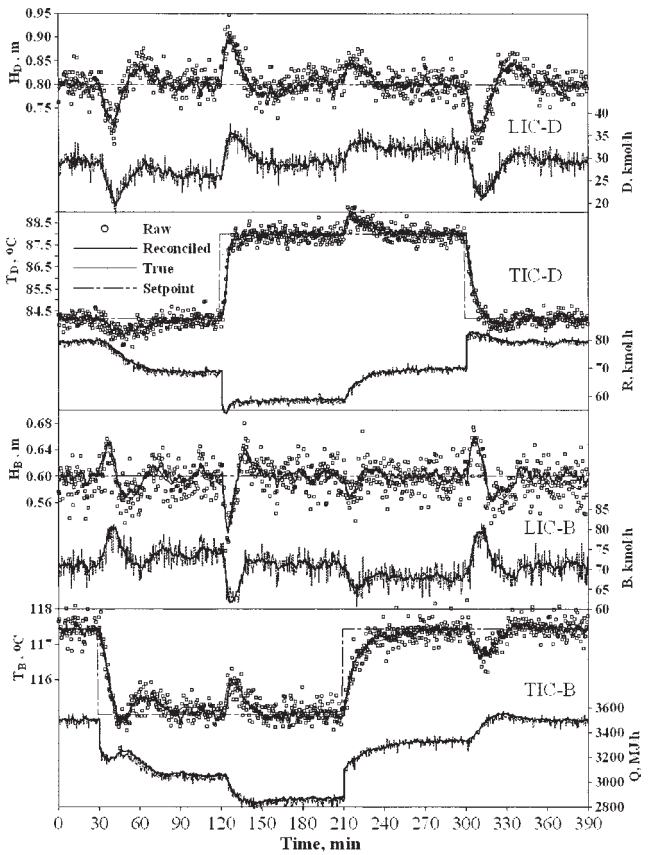

Figure 9. Closed-loop performance of AANN-based DDR for measurements corrupted by the nominal noise level.

The thinner dashed lines for the manipulated variables represent control moves without the AANN-based DDR.

Table 4. MSE/σ^2, Values of Reconciled Data for the Four Controlled Variables Achieved by the AANN-based DDR*

	$MSE_i/\sigma_i^2 \pm$ SE			
Noise Level	H_D	T_D	H_B	T_B
Nominal, $\sigma_1 = \sigma$	0.169 ± 0.006	0.245 ± 0.006	0.182 ± 0.006	0.163 ± 0.004
Doubled, $\sigma_2 = 2\sigma$	0.121 ± 0.003	0.152 ± 0.004	0.110 ± 0.004	0.128 ± 0.003
Tripled, $\sigma_3 = 3\sigma$	0.112 ± 0.003	0.134 ± 0.004	0.097 ± 0.003	0.121 ± 0.003

*Results are based on 20 white noise sequences. SE, standard error.

impact on the other three controlled variables arising from inherent interactions among the four control loops. Despite the complex dynamics, the AANN-based DDR performed very well in tracking the true values. The reconciled data were significantly less noisy than the raw measurements. Moreover, the high-frequency oscillations of the control moves were also reduced.

To quantitatively assess the performance of AANN-based DDR, the MSE/σ^2 value for each controlled variable was evaluated for 20 simulation runs, each run with a different white noise sequence. The mean of the MSE/σ^2 values and its standard error for each controlled variable are presented in the first row of Table 4. Compared to the variances of the raw measurements, the MSE values for the reconciled data were much smaller, indicating that noise reduction for the four controlled variables was significant.

Without retuning the AANNs, the performance of the identical AANNs was evaluated when the noise level of the measurements was two- and threefold the nominal noise level. For the identical sequences of top and bottom temperature set point changes, the process was simulated 20 times for each noise level, each time with a different white noise sequence. The mean of the MSE/σ^2 values and its standard error for each controlled variable for the two noise level cases are presented in the second and third rows of Table 4, respectively. On average, the MSE/σ^2 values decreased despite the increase of noise level, indicating the performance of the AANN-based DDR for noise reduction became more effective, thereby demonstrating robustness to the change in the magnitude of the measurement noise. For threefold the nominal noise level, the raw measurements, reconciled, and true values for the four controlled variables along with the control moves are presented in Figure 10. The dynamic responses of the four controlled variables without data reconciliation were considerably masked by the measurement noise. The noise-to-signal ratios of the two liquid levels were very high, such that it was difficult to observe any dynamic trend. However, with the AANN-based DDR, the dynamic responses were clearly observed and the high magnitudes of oscillations of the control moves were significantly reduced. The benefits of using the AANN-based DDR inside the control loops became more obvious. However, it is important to point out that when process conditions change, such as when the tray efficiency decreases, the models used in training the AANN may not be able to represent as well the dynamics of the process. Therefore, new models have to be identified.

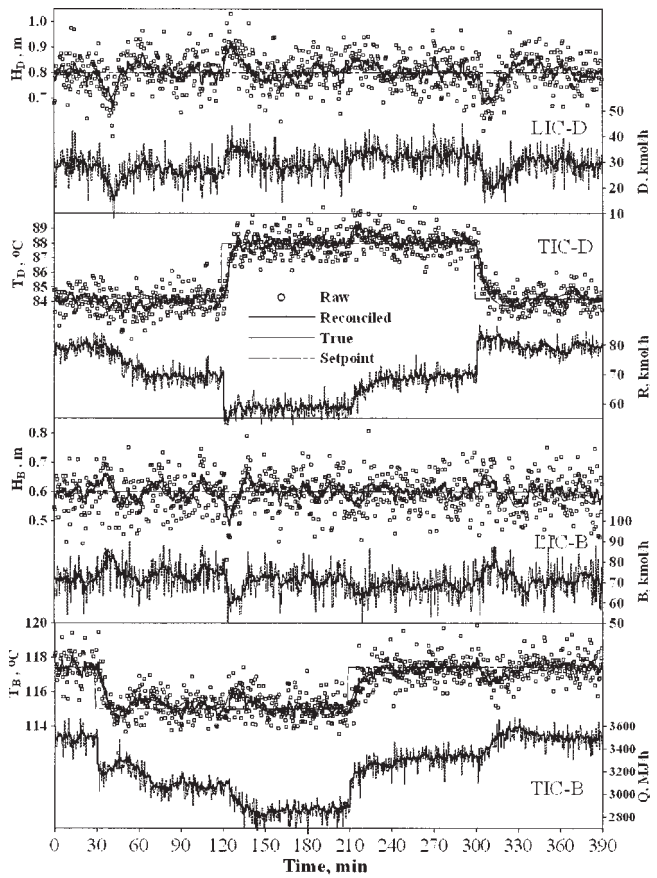

Figure 10. Closed-loop performance of AANN-based DDR for measurements corrupted by thrice the nominal noise level.

The thinner dashed lines for the manipulated variables represent control moves without the AANN-based DDR.

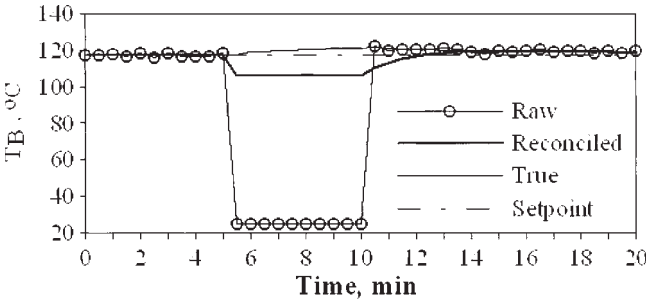

Figure 11. Closed-loop performance of AANN-based DDR for bottom temperature when there was a sudden failure in measurement device.

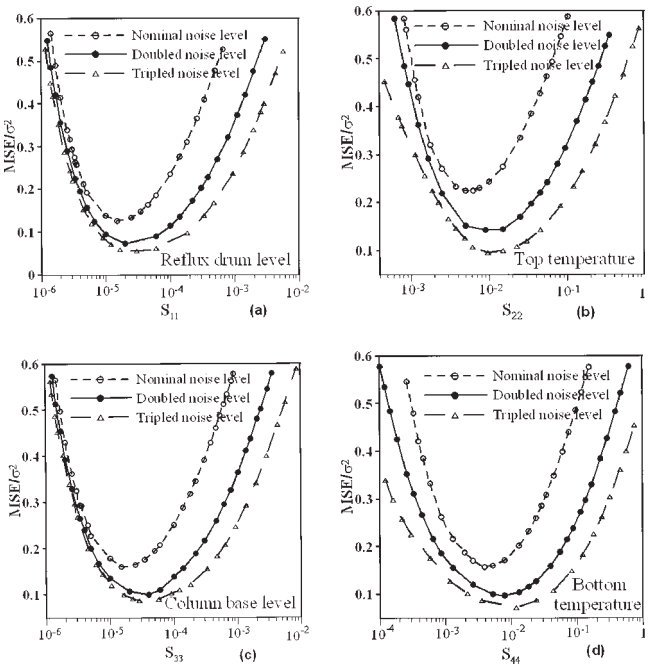

Figure 12. Performance of Kalman filter for the four controlled variables for nominal, twice, and thrice the nominal noise level as a function of variances of process model noise.

(a) Reflux drum level; (b) top temperature; (c) column base level; (d) bottom temperature.

Next, the performance of the AANN-based DDR was tested when a fault occurred in the measurement device. The measurement of the bottom temperature was taken as an illustrative example. At process nominal steady state, this measurement had a sudden failure at 5 min, decreasing to 25°C from its 118.43°C nominal value. Then, after 5 min, it suddenly returned to its normal condition. The raw, reconciled, and true values of the bottom temperature are presented in Figure 11. Upon failure of the measurements, the reconciled data gradually displayed deviations from the true values and then remained at a constant value of 106°C. However, during this failure, the AANN was able to provide relatively more reliable data to the controller than the raw measurements. When the faulty measurements were corrected and returned to their normal condition, the AANN predictions returned to the true values within 4 min, thus indicating the AANN-based DDR was robust and offered a means to prevent the process from complete failure when a measurement fault occurs.

Comparison to Kalman filter

The performance of the AANN-based DDR to attenuate the impact of measurement noise embedded inside the control loops was also compared to that of an extended Kalman filter (EKF). To use the Kalman filter, process models (Eqs. 12–15) were used directly. Because all four controlled variables were directly measured, the measurement model was

$$\mathbf{y}_t = \mathbf{x}_t + \boldsymbol{\varepsilon}_t \tag{21}$$

Because the covariance matrix of the process model noise **S**, required by the Kalman filter, was difficult to determine, **S** was assumed diagonal with elements treated as tuning parameters in calculating the Kalman gain $\mathbf{K}_t$. In addition, the two neural models for the top and bottom temperatures were linearized at each sampling time to calculate the gain of the Kalman filter.

Performance of the Kalman filter was evaluated when the distillation column was subjected to the same sequence of temperature set-point changes and for the three noise levels that were used to assess the AANN-based DDR. The MSE values standardized by their respective variances of raw measurements, MSE/σ^2, for each controlled variable as a function of the tuning parameters are plotted in Figure 12. It can be observed that, using the Kalman filter for each noise level, the optimal tuning parameters were obtained for minimum MSE/σ^2 values. By increasing or decreasing the tuning parameters away from their optimal values, the values of MSE/σ^2 increased rapidly. When the noise level of the measurements changed, the Kalman filter was retuned to regain its optimality; otherwise, suboptimal performance resulted.

The performance of the extended Kalman filter using its optimal tuning parameters was compared to that of the AANN-based DDR when the measurements were corrupted by the three noise levels, respectively. Results of the mean of the MSE/σ^2 values and its standard error obtained from 20 white noise sequences for each controlled variable and for each noise level are presented in Table 5. Compared to the values shown in Table 4, the extended Kalman filter performed slightly better than the AANN-based DDR in the vicinity of its optimal regions. The better performance of the extended Kalman filter is attributed to the fact that process models directly participated in the data reconciliation, whereas process models indirectly participated in the AANN-based DDR. Information about the process dynamics contained in the models was partly lost during offline training of the AANNs. Nevertheless, the better performance of the extended Kalman filter achieved was dependent on the proper online tuning of the variance of the process models and required considerable computational effort to calculate Jacobian

Table 5. MSE/σ^2 Values of Reconciled Data for the Four Controlled Variables Achieved by the Extended Kalman Filter*

	$\text{MSE}_i/\sigma_i^2 \pm$ SE			
Noise Level	H_D	T_D	H_B	T_B
Nominal, $\sigma_1 = \sigma$	0.161 ± 0.005	0.188 ± 0.005	0.151 ± 0.005	0.153 ± 0.004
Doubled, $\sigma_2 = 2\sigma$	0.111 ± 0.004	0.113 ± 0.004	0.104 ± 0.004	0.091 ± 0.003
Tripled, $\sigma_3 = 3\sigma$	0.086 ± 0.004	0.085 ± 0.003	0.083 ± 0.004	0.067 ± 0.002

*Results are based on 20 white noise sequences. SE, standard error.

matrices of the process models and the Kalman gain at each sampling time.

Conclusion

This article developed an AANN-based dynamic data reconciliation algorithm and its performance for process estimation was quantitatively assessed when it was embedded inside feedback control loops. Simulation results from the distillation column demonstrated the AANN-based DDR can efficiently attenuate the impact of measurement noise and consequently resulted in improved process monitoring and/or control. Two AANN structures for dynamic data reconciliation-a feedforward AANN and a recurrent AANN-were proposed. The performance of the recurrent AANN proved more effective than that of the feedforward AANN. Models with various structures, such as linear or nonlinear, can be encapsulated into the training objective function of an AANN to perform dynamic data reconciliation. The covariance matrix of the model residuals provides a measure for offline tuning for the AANN-based DDR. After successful offline training, the AANN can be directly implemented online to perform DDR, without explicitly calculating model predictions as required in the Kalman filter and without resorting to an optimization algorithm as required in the conventional DDR algorithm. The Kalman filter has slightly better performance than the AANN-based DDR in the vicinity of its optimal region, at the cost of appropriate online tuning and considerable computational effort to calculate the Kalman gain at each sampling time. Both the AANN-based DDR and the Kalman filter are robust to the changes of noise level in plant measurements. Nevertheless, the AANN-based DDR needs only offline training and then it is easier to implement online and can provide a measure to prevent the process from complete failure when measurement fault occurs.

Literature Cited

1. Kamen EW, Su JK. Introduction to Optimal Estimation. London: Springer; 1999.
2. Wilson DI, Agarwal M, Rippin DWT. Experiences implementing the extended Kalman filter on an industrial batch reactor. *Comput Chem Eng*. 1998;22:1653–1672.
3. Liebman MJ, Edgar TF, Lasdon LS. Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. *Comput Chem Eng*. 1992;16:963–986.
4. Ramamurthi Y, Sistu PB, Bequette BW. Control relevant dynamic data reconciliation and parameter estimation. *Comput Chem Eng*. 1993;17:41–59.
5. Albuquerque J, Biegler L. Data reconciliation and gross error detection for dynamic systems. *AIChE J*. 1996;42:2841–2856.
6. Hodouin D, Makni S. Real-time reconciliation of mineral processing plant data using bilinear material balance equations coupled to empirical dynamic models. *Int J Miner Process*. 1996;48:245–264.
7. Bagajewicz M, Jiang Q. Integral approach to plant linear dynamic reconciliation. *AIChE J*. 1997;43:2546–2558.
8. Binder T, Blank L, Dahmen W, Marquardt W. On the regularization of dynamic data reconciliation problems. *J Process Control*. 2002;12:557–567.
9. Bai S, Thibault J, McLean DD. Use of autoassociative neural network for dynamic data reconciliation. Proceedings of the 16th IFAC World Congress, Prague, Czech Republic, July 4–8; 2005.
10. Kramer MA. Autoassociative neural networks. *Comput Chem Eng*. 1992;16:313–328.
11. Du Y, Hodouin D, Thibault J. Use of novel autoassociative neural network for nonlinear steady-state data reconciliation, *AIChE J*. 1997;43:1785–1796.
12. Bai S. Assessment of Controller Performance via Embedded Data Reconciliation. M.A.Sc. Thesis. Ottawa, Canada: University of Ottawa; 2003.

Manuscript received Apr. 4, 2006, and revision received Nov. 22, 2006.