DYNAMIC DATA RECONCILIATION FOR IMPROVED PROCESS PERFORMANCE

Shuanghua Bai, Jules Thibault* and David D. McLean
Department of Chemical Engineering, University of Ottawa
161 Louis Pasteur St., Ottawa, Ontario, Canada K1N 6N5
* Corresponding author. Fax: 1-613-562-5172, email: thibault@genie.uottawa.ca

Key Words: Measurement noise, Data reconciliation, Controller performance.

1. Introduction

One of main thrusts of modern plant operation is to improve the quality of online information in distributed control system (DCS). Accurate information about the current state of a process is paramount for plant monitoring and control. Unfortunately, process measurements are often corrupted by measurement noise. The presence of measurement noise not only prevents plant operators from identifying true values of process variables, but also deteriorates controller performance when the raw measurements are directly transmitted to controllers to calculate control moves.

To cope with these problems, digital filters, such as exponentially weighted moving average (EWMA) or moving average (MA) filters, have been widely applied. The EWMA and MA filters use measurement temporal redundancy where past and current measurements are averaged to give estimates for the current values of the process variables. These classical filters provide satisfactory performance for steady-state or slow dynamic processes. However, under process transient conditions, they inevitably introduce larger time delay in order to effectively attenuate the noise. Consequently, advanced filters, such as model-based filters, have been developed for dynamic processes. The model-based filters employ both measurement temporal and spatial redundancy, where information from both past measurements and other variables in different places of a plant via process models are used for the state estimation. A well known model-based filter is the Kalman filter developed in 1960. It employs stochastic linear-state-space process and measurement models. The most attractive advantage of the Kalman filter lies in its optimal estimation in the sense of minimum mean squared prediction errors, and it has acquired a reputation as a panacea for process state estimation and prediction [1]. The optimality of the Kalman filter requires two restrictive prerequisites, linear-state-space models and independent white Gaussian noise for both process and measurements. Although applications of Kalman filters in tracking moving objects such as aircraft and missiles have been common, their applications in chemical engineering are relatively infrequent [2, 3]. As Brosilow and Joseph [3] pointed out, the difficulties in implementing Kalman filters in chemical industries are associated with identifying reliable dynamic models for the processes as well as specifying stochastic properties of process model and measurement noise. Kalman filters are usually tuned online during their implementations. However, their applications have met some problems, such as divergent and unreliable results and difficulty to tune [4].

Dynamic data reconciliation, as an alternative to the Kalman filter, was studied by Bai et al. [5]. The principle of data reconciliation lies in its integration of information from both measurements and process models to provide more reliable estimates of process variables. More importantly, the reconciled data are consistent with known relationships between process variables, such as mass and heat balances. The concept of data reconciliation was initially developed to calculate mass balances for steady-state processes [6]. The interest in applying data reconciliation in plants started in the 1980s when plant managers realized that data reconciliation can turn process raw data into consistent and reliable information, and such information is critical for effective plant operation and management. During the past decade, the benefits of applying data reconciliation to steady-state processes have triggered some researchers to develop data reconciliation for dynamic processes, for example, [5, 7-14].

This paper introduces the methodologies of dynamic data reconciliation (DDR) developed by Bai et al. [5, 7, 15, 16]. The DDR, viewed as a filter, makes use of discrete dynamic models that can be phenomenological or empirical as constraints in reconciling raw measurements, so that the “best” estimates for the process variable are obtained at each sampling time, and more importantly, the reconciled values are used for process control. In

this paper, the impact of measurement noise on controller performance is investigated. Then, the use of DDR to improve controller performance is conducted. The performance of DDR in improving controller performance is compared to that of EWMA filter. Finally, a method to use autoassociative neural networks to perform DDR is studied. All these studies are carried out via simulation of a binary distillation column with four PI control loops.

2. Methodology

2.1. Predictor-corrector form for DDR algorithm

Consider a plant with M measured variables, and assume the measurements are contaminated only by white Gaussian noise, at time t, we can write

$$\mathbf{y}_t = \mathbf{x}_t + \boldsymbol{\varepsilon}_t \quad (1)$$

where $\mathbf{y}_t$ is a M×1 vector of the measured values for the M variables, $\mathbf{x}_t$ is a M×1 vector of the true values of the M variables, and $\boldsymbol{\varepsilon}_t$ is a M×1 vector of random variables representing the white Gaussian noise, $\boldsymbol{\varepsilon}_t \sim \mathrm{N}(\mathbf{0}, \mathbf{V})$. In order to obtain more accurate knowledge about the current state of the process, prior information is required in addition to the measured values. This prior information can originate from process models. In other words, $\mathbf{x}_t$ can be estimated from model predictions (e.g., one-step-ahead predictions), $\hat{\mathbf{y}}_t$. The information from the model predictions can be combined with the information from the measurements to give better estimates of $\mathbf{x}_t$. Because no mathematical models are perfect, model predictions contain some degree of model prediction errors. We assume the model predictions can also be written in the additive noise form

$$\hat{\mathbf{y}}_t = \mathbf{x}_t + \boldsymbol{\delta}_t \quad (2)$$

where $\boldsymbol{\delta}_t$ is a M×1 vector of random variables representing model prediction errors assumed to be Gaussian white noise, $\boldsymbol{\delta}_t \sim \mathrm{N}(\mathbf{0}, \mathbf{R})$. Now that at time t, $\mathbf{y}_t$ and $\hat{\mathbf{y}}_t$ are known, the most likely values of $\mathbf{x}_t$ can be obtained by simultaneously minimizing the weighted sum of squared measurement and model prediction errors such that

$$\text{Min } J(\hat{\mathbf{x}}_t) = \frac{1}{2}\Big[(\mathbf{y}_t - \hat{\mathbf{x}}_t)^{\mathrm{T}} \mathbf{V}^{-1} (\mathbf{y}_t - \hat{\mathbf{x}}_t) + (\hat{\mathbf{y}}_t - \hat{\mathbf{x}}_t)^{\mathrm{T}} \mathbf{R}^{-1} (\hat{\mathbf{y}}_t - \hat{\mathbf{x}}_t)\Big] \quad (3)$$

Analytical solution to $\hat{\mathbf{x}}_t$ is available by taking partial derivatives for the objective function with respect to $\hat{\mathbf{x}}_t$ and setting it to zero yields

$$\hat{\mathbf{x}}_t = (\mathbf{V}^{-1} + \mathbf{R}^{-1})^{-1}(\mathbf{V}^{-1}\mathbf{y}_t + \mathbf{R}^{-1}\hat{\mathbf{y}}_t) \quad (4)$$

Rearranging Equation (4) gives

$$\hat{\mathbf{x}}_t = \hat{\mathbf{y}}_t + \mathbf{K}(\mathbf{y}_t - \hat{\mathbf{y}}_t) \quad (5)$$

where $\mathbf{K} = (\mathbf{V}^{-1} + \mathbf{R}^{-1})^{-1}\mathbf{V}^{-1} = (\mathbf{I} + \mathbf{V}\mathbf{R}^{-1})^{-1}$, and $\mathbf{I}$ is the identity matrix. Equation (5) is the expression of the DDR formula, which indicates the estimates of the current process variables (the reconciled values) are comprised of two terms that are respectively the model predicted values $\hat{\mathbf{y}}_t$ plus the measurement corrections $\mathbf{K}(\mathbf{y}_t - \hat{\mathbf{y}}_t)$. $\mathbf{K}$ is the gain of the DDR and it ranges from $\mathbf{0}$ to $\mathbf{I}$. When measurement noise is significantly larger than model prediction errors, the gain of the DDR, $\mathbf{K} \to \mathbf{0}$. Consequently, $\hat{\mathbf{x}}_t$ approach model predictions, $\hat{\mathbf{x}}_t \to \hat{\mathbf{y}}_t$. On the other hand, when the model predictions have significant errors, then $\mathbf{K} \to \mathbf{I}$, and $\hat{\mathbf{x}}_t$ approach raw measurements, $\hat{\mathbf{x}}_t \to \mathbf{y}_t$.

The estimation error by the DDR algorithm is given by

$$\boldsymbol{\xi}_t = \hat{\mathbf{x}}_t - \mathbf{x}_t \quad (6)$$

Putting Equation (5) into (6) and using Equations (1) and (2) give

$$\boldsymbol{\xi}_t = \boldsymbol{\delta}_t + \mathbf{K}(\boldsymbol{\varepsilon}_t - \boldsymbol{\delta}_t) \quad (7)$$

Taking expectations of both sides of Equation (7) yields

$$\mathrm{E}[\boldsymbol{\xi}_t] = \mathbf{K}\mathrm{E}[\boldsymbol{\varepsilon}_t] + (\mathbf{I} - \mathbf{K})\mathrm{E}[\boldsymbol{\delta}_t] \quad (8)$$

Because $\mathrm{E}[\boldsymbol{\varepsilon}_t] = \mathbf{0}$ and $\mathrm{E}[\boldsymbol{\delta}_t] = \mathbf{0}$, thus $\mathrm{E}[\boldsymbol{\xi}_t] = \mathbf{0}$, meaning that the DDR is an unbiased estimator. From Equation (7), the variance of the estimation error can be given by

$$\mathrm{V}(\boldsymbol{\xi}_t) = \mathbf{K}^{\mathrm{T}}\mathrm{V}(\boldsymbol{\varepsilon}_t)\mathbf{K} + (\mathbf{I} - \mathbf{K})^{\mathrm{T}}\mathrm{V}(\boldsymbol{\delta}_t)(\mathbf{I} - \mathbf{K}) \quad (9)$$

Putting $\mathrm{V}(\boldsymbol{\varepsilon}_t) = \mathbf{V}$, $\mathrm{V}(\boldsymbol{\delta}_t) = \mathbf{R}$ and $\mathbf{K} = (\mathbf{V}^{-1} + \mathbf{R}^{-1})^{-1}\mathbf{V}^{-1}$ into (9) and rearranging yield

$$\mathrm{V}(\boldsymbol{\xi}_t) = \left(\mathbf{V}^{-1} + \mathbf{R}^{-1}\right)^{-1} \quad (10)$$

Equation (10) indicates that the variance of the estimation error is a function of $\mathbf{V}$ and $\mathbf{R}$. Because both $\mathbf{V}$ and $\mathbf{R}$ are symmetrical and positive definite matrices, the elements of $\mathrm{V}(\boldsymbol{\xi}_t)$ are smaller than those in $\mathbf{V}$ and $\mathbf{R}$, indicating that the combination of the measured and model predicted values results in more accurate estimates than using only one of them individually. The predictor-corrector form for the DDR algorithm can be schematically illustrated in Figure 1.

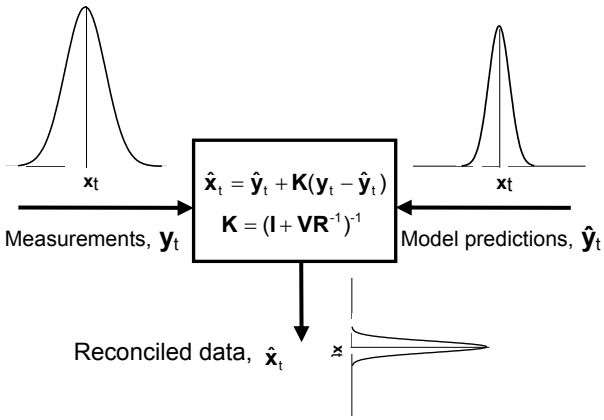

Figure 1 Predictor-corrector form of DDR algorithm.

It is important to mention that model prediction errors are rarely white Gaussian noise, and their statistical properties, such as its covariance matrix **R**, is difficult to evaluate. Indeed, models can be represented by complex functions and numerous linear and nonlinear model structures, can contain parameters with varying degree of uncertainty, and their predictions can be affected by erroneous input variables. Consequently, elements in **R**, in other words, elements in the DDR gain matrix **K** can be treated as tuning parameters.

2.2. Comparison of DDR with Kalman filter

It is interesting to note that the form of the DDR (Equation (5) is similar to that of a Kalman filter. The use of a Kalman filter demands a specific structure for process state/measurement models, namely,

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_{t-1} + \mathbf{w}_{t-1} \quad (11)$$

$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \boldsymbol{\varepsilon}_t \quad (12)$$

where **A**, **B**, and **C** are deterministic matrices with appropriate dimensions. If all the state variables of the process are directly measured, **C** is an identity matrix. $\mathbf{w}_t$ denotes process noise assumed to be white Gaussian noise and independent of $\boldsymbol{\varepsilon}_t$, $\mathbf{w}_t \sim \mathrm{N}(\mathbf{0}, \mathbf{S})$. An important assumption in the Kalman filter is that all uncertainties in the models of the process can be efficiently assimilated to $\mathbf{w}_t$. For the problem defined by Equations (11) and (12), the optimal estimates of the process variables by the Kalman filter are given by

$$\hat{\mathbf{x}}_t = \hat{\mathbf{y}}_t + \boldsymbol{\kappa}_t \left(\mathbf{y}_t - \mathbf{C}\hat{\mathbf{y}}_t\right) \quad (13)$$

where

$$\hat{\mathbf{y}}_t = \mathbf{A}\hat{\mathbf{x}}_{t-1} + \mathbf{B}\mathbf{u}_{t-1} \quad (14)$$

and the Kalman gain $\boldsymbol{\kappa}_t$ is recursively calculated by

$$\mathbf{P}_t^- = \mathbf{A}\mathbf{P}_{t-1}\mathbf{A}^T + \mathbf{S} \quad (15)$$

$$\boldsymbol{\kappa}_t = \mathbf{P}_t^-\mathbf{C}^T\left(\mathbf{C}\mathbf{P}_t^-\mathbf{C}^T + \mathbf{V}\right)^{-1} \quad (16)$$

$$\mathbf{P}_t = \mathbf{P}_t^- - \boldsymbol{\kappa}_t\mathbf{C}\mathbf{P}_t^- \quad (17)$$

Because the matrices $\mathbf{P}_t^-$, $\mathbf{P}_t$ and $\boldsymbol{\kappa}_t$ need to be updated at each sampling time, the Kalman filter is a time-variant system. However, for the process defined by Equations (11,12), matrices $\mathbf{P}_t^-$, $\mathbf{P}_t$ and $\boldsymbol{\kappa}_t$ can reach constant values. At this point, the Kalman filter is known as steady-state Kalman filter. It is important to comment that the recursive calculations of the matrices $\mathbf{P}_t^-$, $\mathbf{P}_t$ and $\boldsymbol{\kappa}_t$ are independent of the measurement realizations. Therefore, they can be computed offline without actually making any measurements. However, for nonlinear process models, the Kalman filter is modified by linearizing the nonlinear terms in the model at each sampling time, and it is known as an extended Kalman filter. In this case, the matrix **A** used to calculate the Kalman gain becomes time-dependent. Matrices $\mathbf{P}_t^-$, $\mathbf{P}_t$ and $\boldsymbol{\kappa}_t$ change correspondingly, and cannot be computed offline.

It is apparent that the Kalman filter involves complex mathematical manipulations. In addition, because it is difficult to determine the covariance matrix of the process noise, **S**, elements in **S** are commonly viewed as tuning parameters rather than measurable constants in implementations of the Kalman filters. It also has been observed that, if matrix **C** in the Kalman filter is an identity matrix, the form of the DDR algorithm is identical to that of the Kalman filter but their gain matrices are calculated in different ways. However, if both **R** for the DDR filter and **S** for the Kalman filter are appropriately tuned, the two filters have been shown equivalent to each other [5]. However, compared to the Kalman filter, the concept of DDR filter is relatively concise and straightforward so that it is easier to understand and implement. In addition, the DDR filter can use a variety of structures for process models (e.g., linear and nonlinear).

2.3. AANN-based DDR algorithm

The DDR algorithm of Equation (5) requires calculating model predictions at each sampling time. Then, the model predictions are reconciled with the measured ones to produce reconciled values for process monitoring and control purposes. For the case where the model predictions cannot be explicitly calculated, the DDR problem formulated by Equation (3) can be modified as

$$\text{Min } J(\hat{\mathbf{x}}_t) = \frac{1}{2}\Big[\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right)^T\mathbf{V}^{-1}\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right) + \mathbf{f}^T\left(\hat{\mathbf{x}}_t\right)\boldsymbol{\Omega}^{-1}\mathbf{f}\left(\hat{\mathbf{x}}_t\right)\Big] \quad (18)$$

where $f(\hat{\mathbf{x}}_t)$ is a functional vector of process models, and $\boldsymbol{\Omega}$ is the covariance matrix of model residuals, whose elements are treated as tuning parameters. Using Equation (18), at each sampling time, the reconciled values can be obtained by iteratively solving the optimization problem using nonlinear programming (NLP) techniques. However, for complex processes longer computation time may be required for the optimization. To remedy this problem, a method to train an autoassociative neural network (AANN) to perform dynamic data reconciliation was proposed by Bai et al. [15]. Once trained offline, the neural network can directly perform data reconciliation online without any iteration, and the neural DDR becomes more suitable for real-time applications.

The AANN architecture for a dynamic process is presented in Figure 2. It has similar architecture as a conventional feedforward neural network, composed of an input layer, three hidden layers and an output layer. The first and third hidden layers contain a relatively larger number of neurons, while the second hidden layer contains a lower number of neurons. The feature of an AANN is to perform data compression by the third layer (bottleneck layer). The first, second and third layers compress the input information to a lower dimension, then the fourth and fifth layers regenerate the main underlining features of the original information. The compression/regeneration process enables the network to represent the input information in a compressed form that can often reveal the essence of the data.

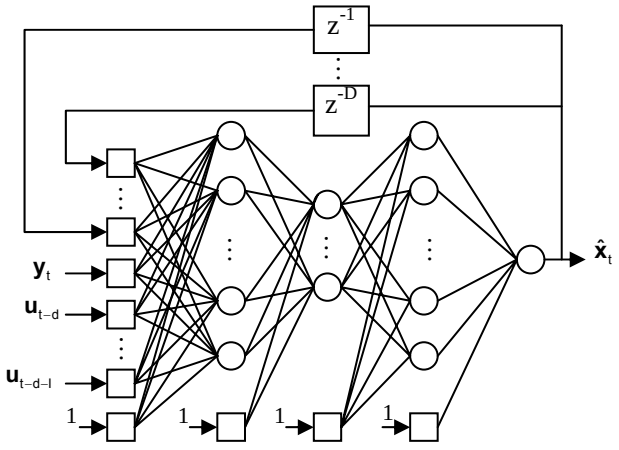

Figure 2 Architecture of an AANN for a dynamic system.

During neural network training, process models are encapsulated within the structure of the network simultaneously to perform data reconciliation. The number of neurons in the input and output layers are determined by the structures of process models employed, while the number of neurons in each hidden layer is determined by trial-and-error. To capture the dynamics of the process, the output of the network $\hat{x}_t$, delayed a number of times, is fed back to the input layer so that the AANN can incorporate both temporal and spatial patterns. In Figure 2, D represents the required number of time delays for the process output variables, and $\mathbf{u}_{t-d}$, …, $\mathbf{u}_{t-d-l}$ represent the inputs with a time delay of d, d+1, …, d+l.

The objective function used to train a dynamic AANN to perform date reconciliation can be written as

$$\text{Min } J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{t=0}^{N}\Big[\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right)^T \mathbf{V}^{-1}\left(\mathbf{y}_t - \hat{\mathbf{x}}_t\right) + \mathbf{f}^T\left(\hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t-1}, ..., \mathbf{u}_{t-d}, \mathbf{u}_{t-d-1}, ...\right)\boldsymbol{\Omega}^{-1} \mathbf{f}\left(\hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t-1}, ..., \mathbf{u}_{t-d}, \mathbf{u}_{t-d-1}, ...\right)\Big] \quad (19)$$

where $\boldsymbol{\theta}$ is a vector of connection weights (parameters) in the AANN, N is the total number of data points used to train the network. The elements in $\boldsymbol{\Omega}$, in this case, are treating as tuning parameters in training the network. Among training algorithms, backpropagation is the simplest method. However, more advanced training algorithms such as the quasi-Newton or the Levenberg-Marquardt methods are most often used for better accuracy and faster convergence. In the first iteration in training the dynamic AANN, the output vector fed back to the input layer is not known, but can be assigned the raw measurements. Then, the network is trained until satisfactory convergence criterion is met. After the first iteration, the vectors of the network outputs are fed back as the inputs, and then the network is trained again. After several recurrent iterations, the feedback vectors and the objective function will not change, indicating the training of the dynamic network has been completed.

It is important to note that the AANN-based DDR can actually embrace any model structures. For cases where process models can be used to explicitly calculate the model predictions, either the predictor-corrector DDR or the AANN-based DDR algorithm can be developed. However, the use of AANN-based DDR provides the advantages that it does not need online tuning because it has been tuned offline during its own training. Furthermore, the use of AANN-based DDR is straightforward. It is not required to calculate the model predicted values at each time, since information about process dynamics has been encoded into the network during training. The scheme of using AANN to perform dynamic data reconciliation is illustrated by Figure 3.

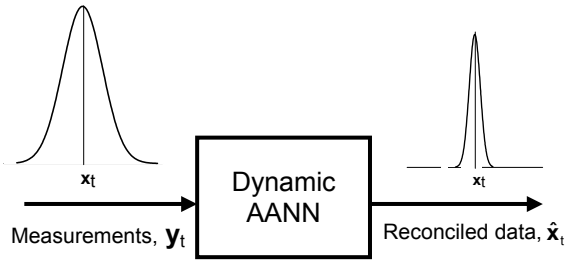

Figure 3 Scheme of AANN-based DDR algorithm.

3. Results Discussion

The methodologies of DDR developed were tested in control of a binary (benzene/toluene) distillation column. The distillation column, presented in Figure 4, has four PI control loops. 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 period of 30 s was used. The dynamic distillation simulator was based on rigorous distillation models, (i.e., mass and heat balances, vapor-liquid equilibrium, and tray hydraulics). The nominal steady-state values for all measurements and their noise levels are listed in Table 1.

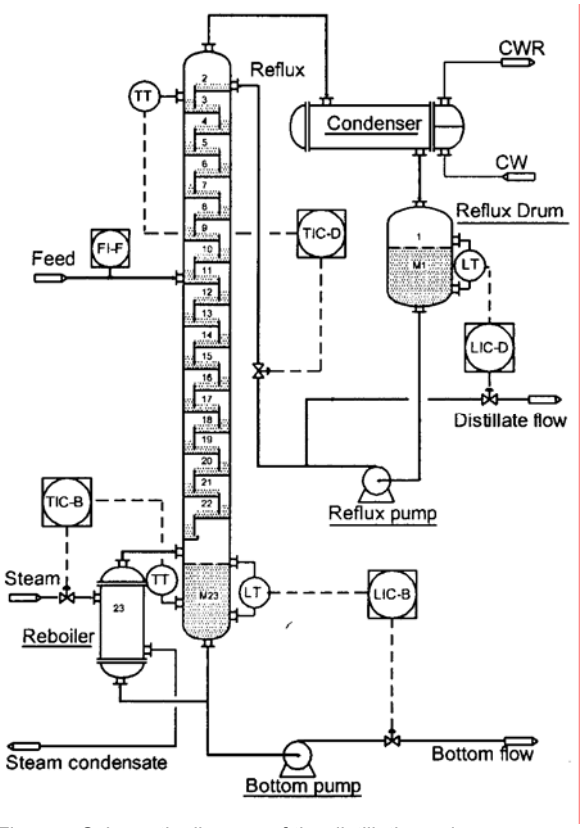

Figure 4 Schematic diagram of the distillation column.

Table 1 Nominal steady-state values and noise levels of measured variables for the distillation column.

Variable	Units	Steady State	Standard Deviation
Feed flow	kmol/h	100.0	0.50
Top temperature	°C	84.2	0.25
Bottom temperature	°C	117.4	0.25
Reflux drum level	m	0.50	0.02
Column base level	m	0.70	0.02

3.1 Effect of measurement noise on performance of controllers

For each pair of controlled and manipulated variables, open-loop responses of the controlled variables to a step change in the manipulated variables were first simulated to obtain the process reaction curve. The open-loop responses were approximated by either first-order-plus-dead-time (FOPDT), or pure-integrator-plus-dead-time (PIPDT) models. Then, Ziegler-Nichols empirical tuning rules were used to determine initial estimates of the controller parameters, K_C and τ_I (see first row of Table 2). Without measurement noise, the performance of the controllers was tested for disturbance regulation when the column was subjected to a series of step changes in feed flow rate having magnitudes of 20%, -40% and 20% of steady-sate values, respectively. Results of this simulation are presented in Figure 5. These results illustrate that the controllers have good performance in the absence of measurement noise.

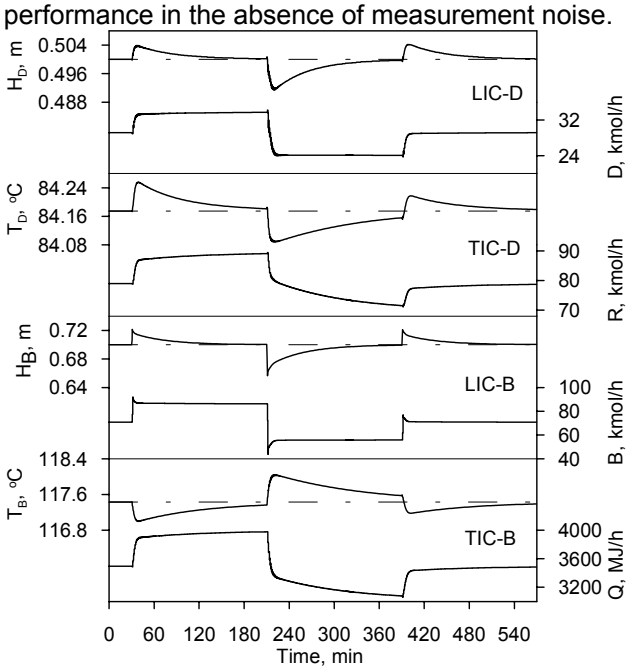

Figure 5 Closed-loop responses with Z-N tunings without measurement noise for series of step changes in feed flow rate. H_D: reflux drum level, D: distillate flow, T_D: top temperature, R:

reflux flow, H_B: base level, B: bottom flow, T_B: bottom temperature, Q: reboiler heat duty.

Table 2 Controller tuning parameters for the distillation column.

Noise	Controller	LIC-D		TIC-D		LIC-B		TIC-B	
		K_C kmol.h^{-1}.m^{-1}	τ_I min	K_C kmol.h^{-1}.^{o}C^{-1}	τ_I min	K_C kmol.h^{-1}.m^{-1}	τ_I min	K_C MJ.h^{-1}.^{o}C^{-1}	τ_I min
No	Initial Z-N	-960.0	46.5	-87.0	48.3	-960.0	46.5	850.7	71.6
Yes	Detuned	-43.2	15.0	-2.2	5.0	-76.4	5.0	45.2	5.0
Yes	With DDR	-92.2	15.0	-4.9	5.0	-173.8	5.0	150.6	5.0

Next, using the standard deviations given in Table 1, white Gaussian noise was added to the true values of the controlled variables to provide more realistic measured values. Subsequently, these noisy measurements were used by the controllers to generate closed-loop responses. Results for the same series of feed flow disturbances are presented in Figure 6. The presence of the measurement noise significantly reduced the performance of all control loops. The dynamic responses of the controlled variables, except for bottom temperature, displayed high-frequency variations around their setpoints, while the manipulated variables were characterized by high magnitude variations, often saturated oscillations that completely masked the expected responses.

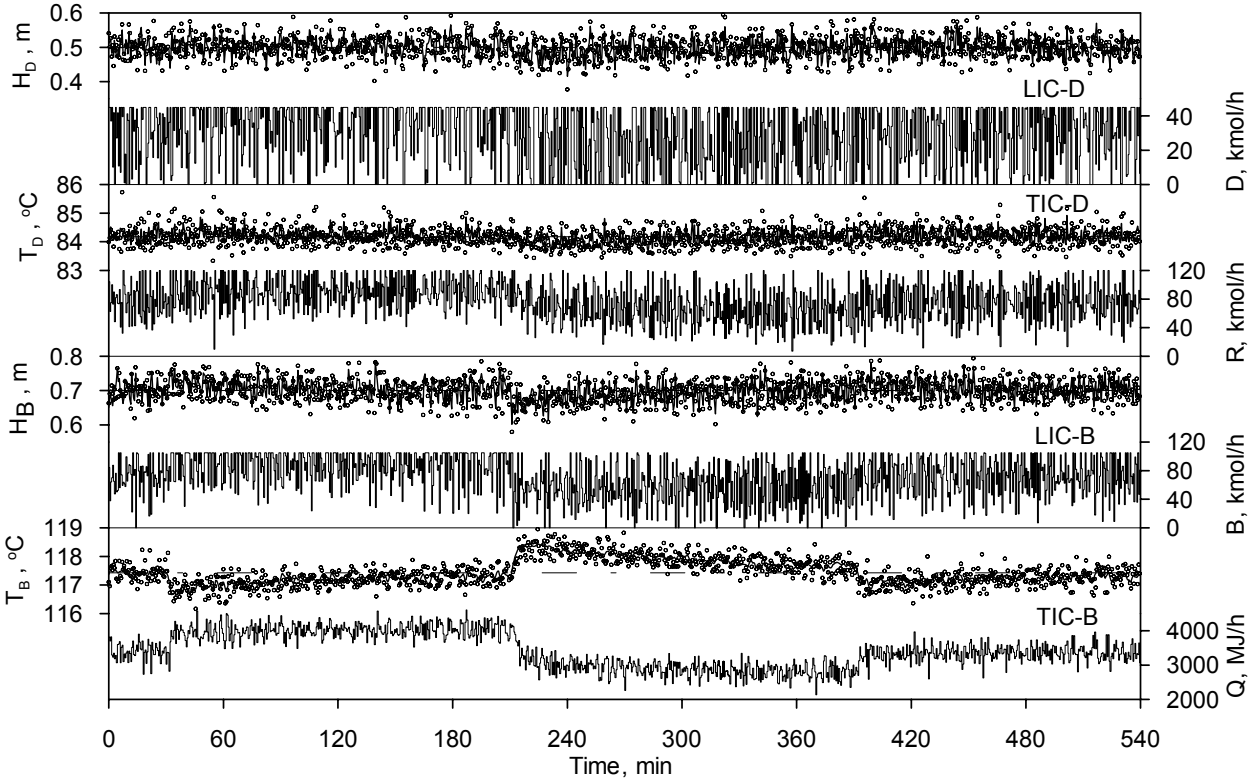

Figure 6 Closed-loop responses with Z-N tunings with measurement noise for series of step changes in feed flow rate. O: raw, —: true, — - —: setpoint.

To regain an acceptable process performance, the four controllers were retuned considering the impact of the measurement noise. For this multiple-input multiple-output (MIMO) discrete PI control system, the tuning objective function is to minimize

$$\Phi(\mathbf{K}_C, \tau_I) = \sum_{i=1}^{4}\left[\alpha_i \sum_{t=0}^{t_s}\left(x_{i,t}^{*} - x_{i,t}\right)^2 \Delta t + \beta_i \sum_{t=0}^{t_s}\left(u_{i,t} - u_{i,t-1}\right)^2 \Delta t\right] \quad (20)$$

where Φ denotes overall cost function of the control system, $x_{i,t}$ represents the true value of controlled variable i, $x_{i,t}^{*}$ represents the setpoint for controlled

variable i, and $u_{i,t}$ represents the manipulated variable for control loop i. Δt is the sampling time interval. α is a weighting factor for the integral of squared errors of the controlled variable (ISE), and β is a weighting factor for the integral of squared differences of the manipulated variable between sampling times t and t-1 (ISDU). t_s represents the total number of process sampling periods over which the integration is performed. The minimization of the ISE term in the objective function attempts to maintain the controlled variable as close as possible to its setpoint, whereas the minimization of the ISDU term attempts to prevent excessive adjustments of the manipulated variables. The selection of the weighting factors for ISE and ISDU terms is a trade-off between dynamic responses of the controlled and manipulated variables. The optimization was performed using quasi-Newton method with lower and upper bounds. Values of optimal controller parameters are presented in the second row of Table 2. The integral times, obtained when the measurement noise was considered, hit their lower bounds. The controller gains were reduced significantly, on average, 23 times smaller. Meanwhile, the integral times were, on average, ten times smaller. Closed-loop responses with the detuned controllers are presented in Figure 7. The high magnitude oscillations of the manipulated variables were significantly reduced compared to those in Figure 6. The dynamics of the controlled variables could be observed despite the noisy measurements.

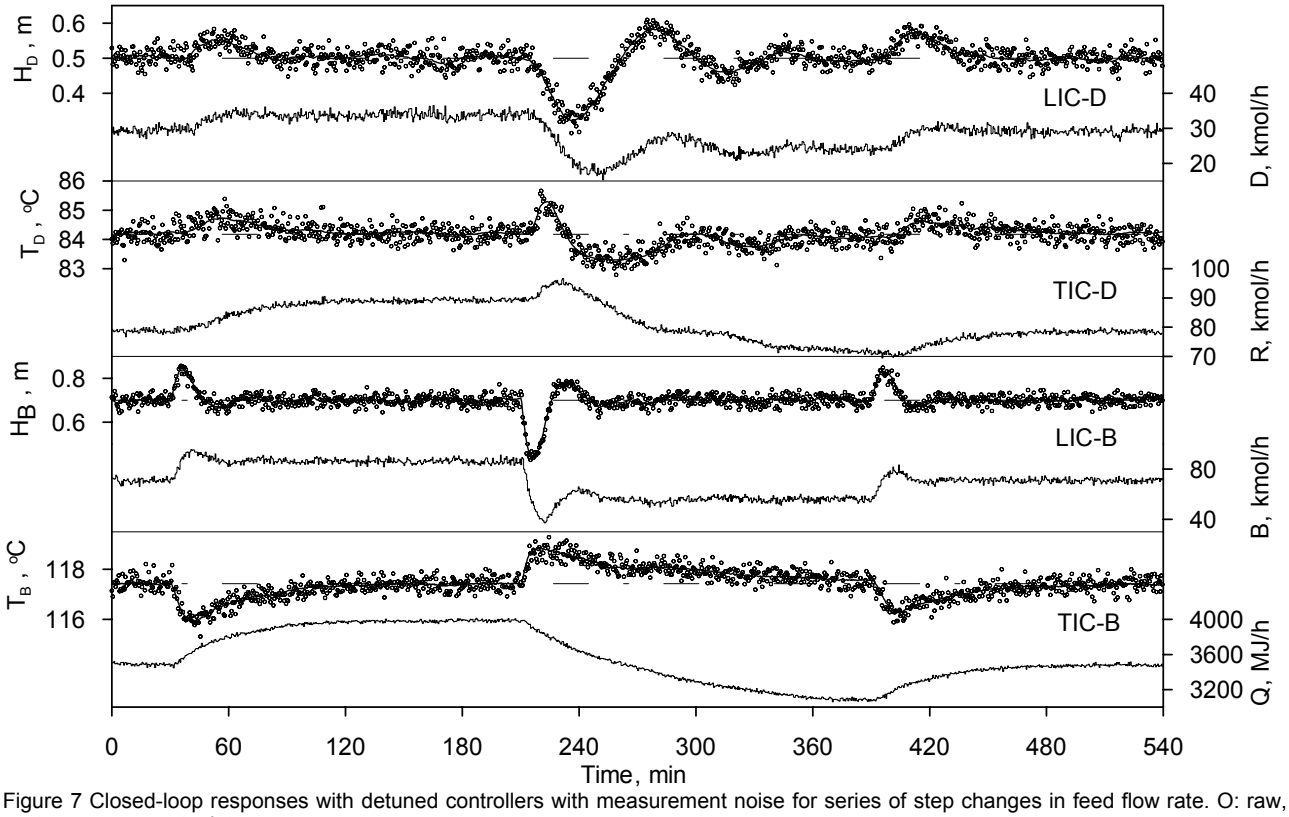

Figure 7 Closed-loop responses with detuned controllers with measurement noise for series of step changes in feed flow rate. O: raw, —: true, — - —: setpoint.

Using Equation (20), the overall cost function of the control system was evaluated for the initial Z-N tuning controllers without and with measurement noise, and for the detuned controllers. These results are presented in Table 3. Without measurement noise the initial Z-N tunings yielded very smaller cost function. With measurement noise the initial Z-N tunings resulted in very large cost function. However, when the controllers were detuned, the controller performance became better, but still about eight times larger than without measurement noise.

Table 3 Overall cost function of the control system for the distillation column

Controller	Noise	DDR	Cost function, Φ
Initial Z-N tuning	No	No	0.563
Initial Z-N tuning	Yes	No	2692
Detuning	Yes	No	4.663
With DDR	Yes	Yes	0.925

3.2. Improving controller performance by DDR

Because the presence of measurement noise in the controlled variables deteriorated the controller performance, a DDR algorithm for the distillation column was developed and embedded inside each control loop to reduce the noise propagation as depicted in Figure 8.

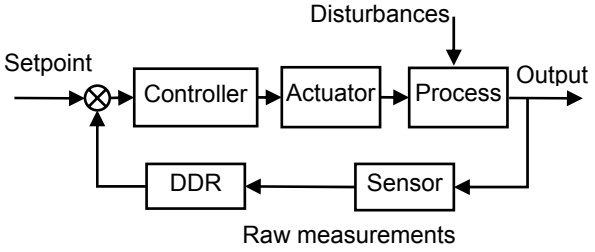

Figure 8 Scheme of DDR algorithm embedded inside feedback loops.

The complexity of the distillation process makes it impractical to develop phenomenological models to be used in the DDR. Consequently, the identified linear models around the process nominal steady state were used.

$$H'_{D,t} - H'_{D,t-1} = -\ 9.058\times10^{-4}D'_{t-1} - \ 9.058\times10^{-4}R'_{t-1} + 2.112\times10^{-5}Q'_{t-1} + 1.06\times10^{-5}Q'_{t-2} \quad (21)$$

$$T'_{D,t} - 0.9414T'_{D,t-1} = -5.64\times10^{-3}R'_{t-1} - 4.332\times10^{-3}R'_{t-2} + 2.5\times10^{-4}Q'_{t-5} - 1.184\times10^{-4}F'_{t-1} - 2.368\times10^{-4}F'_{t-2} \quad (22)$$

$$H'_{B,t} - H'_{B,t-1} = 1.023\times10^{-3}R'_{t-2} - 1.152\times10^{-3}B'_{t-1} - 2.056\times10^{-5}Q'_{t-1} - 1.028\times10^{-5}Q'_{t-2} + 7.942\times10^{-4}F'_{t-1} + 3.971\times10^{-4}F'_{t-2} \quad (23)$$

$$T'_{B,t} - 0.9228T'_{B,t-1} = -0.011R'_{t-8} - 0.00385R'_{t-9} + 4.867\times10^{-5}Q'_{t-1} + 6.084\times10^{-4}Q'_{t-2} - 7.583\times10^{-3}F'_{t-3} \quad (24)$$

where the prime indicates variables in their deviation forms. The DDR algorithm for the distillation column can be written in the compact form

$$\begin{bmatrix} H^r_{D,t} \\ T^r_{D,t} \\ H^r_{B,t} \\ T^r_{B,t} \end{bmatrix} = \begin{bmatrix} \hat{H}_{D,t} \\ \hat{T}_{D,t} \\ \hat{H}_{B,t} \\ \hat{T}_{B,t} \end{bmatrix} + \mathbf{K} \left\{ \begin{bmatrix} H^m_{D,t} \\ T^m_{D,t} \\ H^m_{B,t} \\ T^m_{B,t} \end{bmatrix} - \begin{bmatrix} \hat{H}_{D,t} \\ \hat{T}_{D,t} \\ \hat{H}_{B,t} \\ \hat{T}_{B,t} \end{bmatrix} \right\} \quad (25)$$

where the superscript "r" represents the reconciled values, "m" the measured values, and "$\wedge$" the model predicted values. **K** is the gain matrix of the DDR and is assumed to be diagonal. The elements in **K** were treated as tuning parameters and they were selected such that the reconciled values were as close as possible to their true values. The mean squared differences (MSD) between the reconciled and the true values as a function of the DDR gain for each controlled variable were evaluated, and the results are presented in Figure 9. The minimum MSD values were obtained with the following DDR gain matrix

$$\mathbf{K} = \begin{array}{c} \begin{matrix} H_D & T_D & H_B & T_B \end{matrix} \\ \begin{bmatrix} 0.21 & 0 & 0 & 0 \\ 0 & 0.24 & 0 & 0 \\ 0 & 0 & 0.21 & 0 \\ 0 & 0 & 0 & 0.23 \end{bmatrix} \end{array} \quad (26)$$

The DDR gain matrix moving away from the optimal gain matrix resulted in lager values of MSDs.

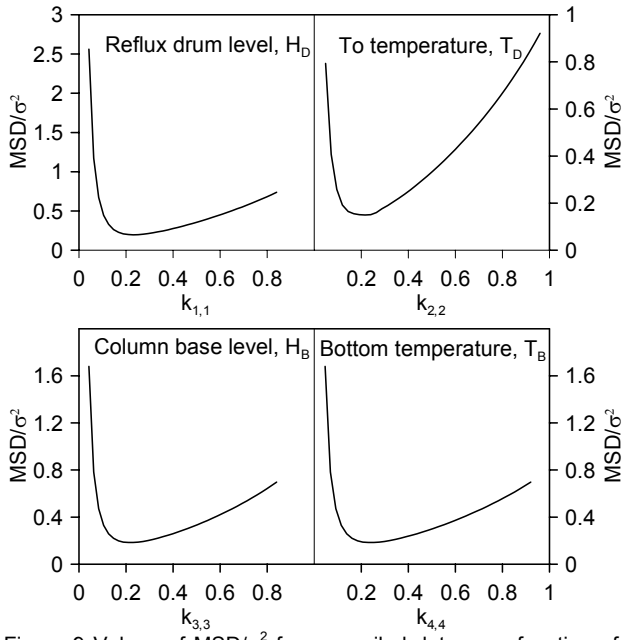

Figure 9 Values of MSD/σ^2 for reconciled data as a function of DDR gain. σ^2 is the variance of the raw measurements for each variable used to scale the MSD.

Using Equation (26) for the DDR gain matrix, the controllers were re-tuned by minimizing the same objective function of Equation (20) when the distillation column was submitted to the same sequence of feed flow rate disturbances. Optimal values of the controller parameters are presented in the third row of Table 2. The controller gains increased by factors of 2~3, compared to the detuned controllers, while the integral times remained unchanged, meaning that the controllers became more aggressive. The values of the overall cost function of the control system with the embedded DDR is presented in Table 3. The overall cost function decreased significantly from Φ=4.663 to Φ=0.925, indicating the controller performance was significantly enhanced by the DDR. The raw, reconciled and true values for the

controlled variables and the control actions are presented in Figure 10. The deviations of the controlled variables from their setpoints decreased considerably, while the excessive variations of the manipulated variables were reduced even with these more aggressive controllers.

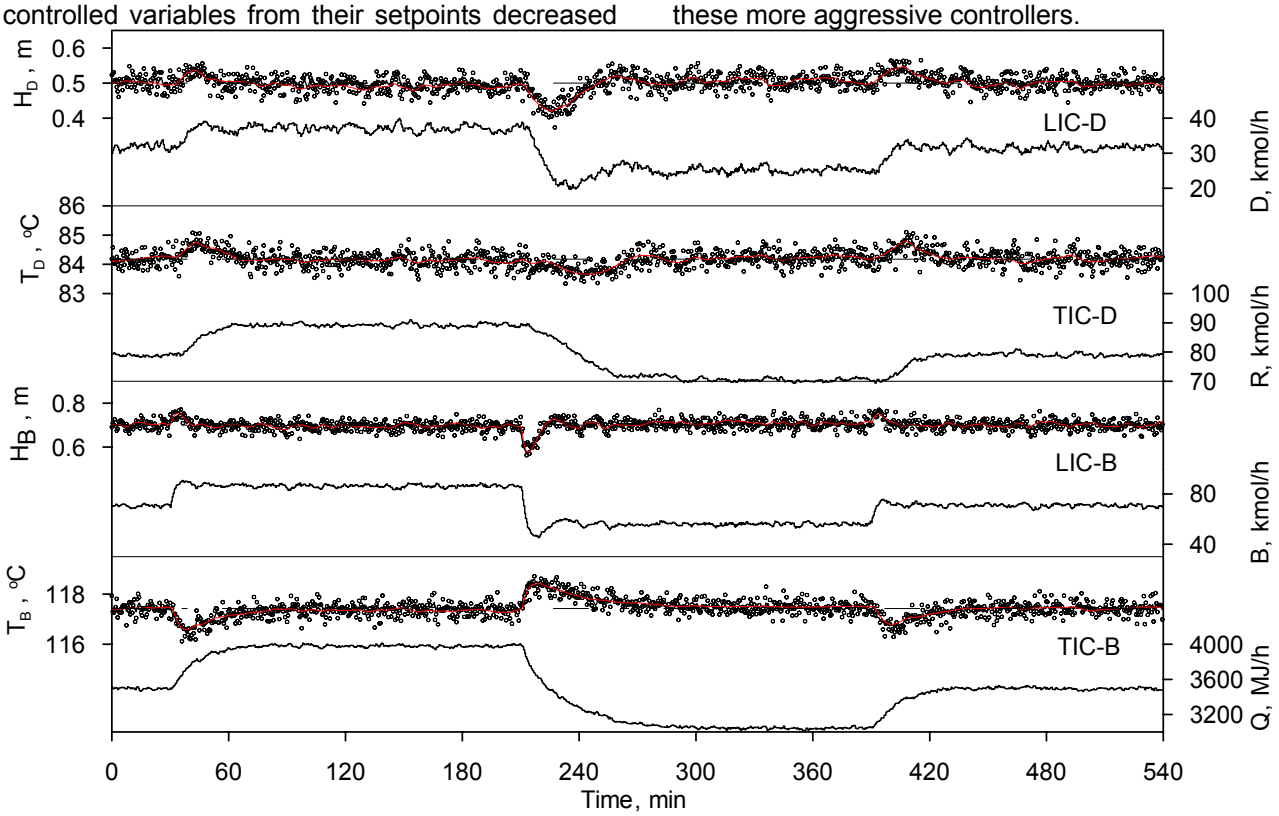

Figure 10 Closed-loop responses of the distillation column with DDR inside feedback loops for series of step changes in feed flow rate. O: raw, ▬: reconciled, —: true, — - —: setpoint.

3.3. Comparison of DDR to EWMA filter

The performance of the DDR in improving the controller performance was compared to that of classical EWMA filter. The EWMA filter for the process can be given by

$$\begin{bmatrix} H^{f}_{D,t} \\ T^{f}_{D,t} \\ H^{f}_{B,t} \\ T^{f}_{B,t} \end{bmatrix} = \Psi \begin{bmatrix} H^{f}_{D,t-1} \\ T^{f}_{D,t-1} \\ H^{f}_{B,t-1} \\ T^{f}_{B,t-1} \end{bmatrix} + (\mathbf{I} - \Psi) \begin{bmatrix} H^{m}_{D,t} \\ T^{m}_{D,t} \\ H^{m}_{B,t} \\ T^{m}_{B,t} \end{bmatrix} \qquad (27)$$

where the superscript "f" denoting the filtered values, Ψ is a diagonal matrix whose elements are tuning parameters for the filter. The optimal tuning parameters for the EWMA filter having the smallest MSD values for each controlled variable were

$$\Psi = \begin{bmatrix} 0.86 & 0 & 0 & 0 \\ 0 & 0.77 & 0 & 0 \\ 0 & 0 & 0.73 & 0 \\ 0 & 0 & 0 & 0.79 \end{bmatrix} \qquad (28)$$

Using this EWMA filter, the minimum overall cost function of the control system was Φ=1.36. Compared to Φ=0.925 yielded by the DDR, the performance of the DDR is better than that of the EWMA. The better performance of the DDR was attributed to the fact that the DDR is able to anticipate process dynamics because process dynamic models are an integral part of the DDR.

The controller performance was considerably improved by the DDR, even though simple linear models were used. The performance of the DDR depends on the accuracy of process models employed. More comprehensive models are expected to give better performance of the DDR. If "perfect" models were assumed and used, the measurement noise could be completely eliminated. In other words, the performance of the control system was studied with "perfect" DDR. The optimal overall cost function of the control system with the "perfect" DDR was Φ=0.22. The overall cost function of the control system without filter, with the EWMA filter, with the linear DDR and with the assumed "perfect" DDR are presented in Figure 11.

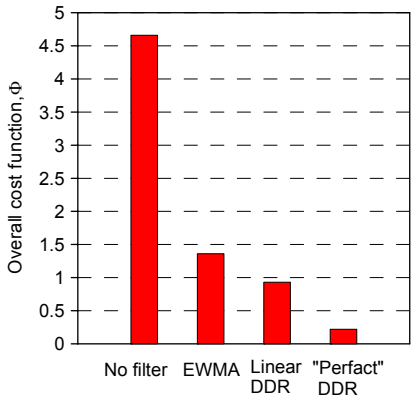

Figure 11 Comparison of overall cost function of the control system without and with different filters.

3.4. AANN-based DDR

As mentioned, the use of predictor-corrector form for the DDR needs to calculate model predictions at each sampling time and online tuning. Next, the use of AANN to perform online data reconciliation for the distillation column was investigated. Previous studies [17] showed that dynamic behavior of the two liquid levels is linear, whereas it is highly nonlinear for the two temperatures. The nonlinearity of the two temperatures lies in the fact that the top temperature could never decrease below the boiling point of pure benzene, and the bottom temperature could never exceed the boiling point of pure toluene under the column pressures. The DDR employing linear models was only effective for a narrow range of operating points around the steady state. When the controllers of the two temperatures had larger setpoint changes, the effectiveness of the DDR was reduced. Consequently, nonlinear models need to be developed for the two variables.

To more adequately capture the nonlinear dynamics of the two temperatures, under open loop conditions the distillation column was excited by a series of random step changes in the manipulated variables. Using the data sets, a feedforward neural network having three layers with eight neurons (including the bias neuron) in the input layer, eight neurons in the hidden layer and one neuron in the output layer, was developed for the top temperature. Concisely this neural network having the structure {8, 8, 1} can be expressed as

$$\hat{T}_{D,t} = NN(\hat{T}_{D,t-1}, R_{t-1}, R_{t-2}, R_{t-3}, Q_{t-1}, Q_{t-2}, Q_{t-3}) \quad (29)$$

where NN represents the nonlinear transformations. Another feedforward neural network having the structure {6, 8, 1} was developed for the bottom temperature as

$$\hat{T}_{B,t} = NN(\hat{T}_{B,t-1}, R_{t-6}, R_{t-7}, Q_{t-1}, Q_{t-2}) \quad (30)$$

Then, the process models of Equations (21), (23), (29) and (30) were respectively encapsulated in the training objective function (see Equation (19)) to train AANNs for each controlled variable. For each AANN, the number of neurons in the input layer was determined by the structure of the dynamic models, while the number of neurons in each hidden layer was selected by trail-and-error. The structures of the AANNs are presented in Table 4.

Table 4 Number of neurons in each layer in the AANNs.

Layer	I	II	III	IV	V
AANN for H_D	7	13	2	13	1
AANN for T_D	9	12	3	12	1
AANN for H_B	7	14	3	14	1
AANN for T_B	7	14	3	14	1

The AANNs for the reflux drum level, the top temperature, the column base level and the bottom temperature can be respectively expressed as

$$H^r_{D,t} = AANN(H^m_{D,t}, D_{t-1}, R_{t-1}, Q_{t-1}, Q_{t-2}) \quad (31)$$

$$T^r_{D,t} = AANN(T^m_{D,t}, R_{t-1}, R_{t-2}, R_{t-3}, Q_{t-1}, Q_{t-2}, Q_{t-3}) \quad (32)$$

$$H^r_{B,t} = AANN(H^m_{B,t}, B_{t-1}, R_{t-2}, Q_{t-1}, Q_{t-2}) \quad (33)$$

$$T^r_{B,t} = AANN(T^m_{B,t}, R_{t-6}, R_{t-7}, Q_{t-1}, Q_{t-2}) \quad (34)$$

Notice that, the raw measurements were fed directly into the network, while the outputs of the network were the reconciled values. During offline training the AANN, the variance of the model residual for each controlled variable was the tuning parameter in the training objective function. When it was set to a large value, meaning that the raw measurements were not constrained severely, and consequently the reconciled values (i.e., the output of the network) were close to raw measurements. On the other hand, when it was set to a small value, meaning that a higher level of confidence was put on the models and as a result the model mismatch distorted the reconciled values. By trail-and-error, the variances of the model residuals were selected such that the performance of the AANNs was acceptable. The raw, reconciled and true values of the four variables in network training and validation are presented in Figure 12. It shows the reconciled values were close to the true values and had a lower variance than the raw measurements.

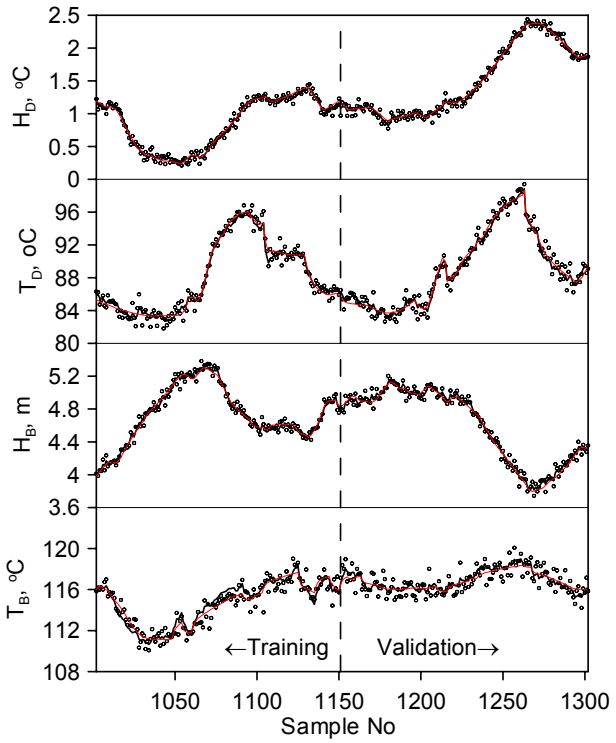

Figure 12 Raw, reconciled and true values for the controlled variables in AANN training and validation. o: raw, ▬: reconciled, —: true.

After the offline training, the AANNs were implanted into the feedback loops. The performance of the AANN-based DDR, along with the controllers, when the top and bottom temperature controllers had a series of setpoint changes is presented Figure 13. It shows that AANN-based DDR performed very well in online tracking the true values of the controlled variables.

With the AANN-based DDR, the optimal overall cost function of the controllers for the process was Φ=1.89. Using the process models (Equations (21), (23), (29), (30)), a DDR with predictor-corrector form was tested. For the same setpoint changes, it resulted in an optimal overall cost function at Φ=1.58 with optimal DDR gain matrix via properly online tuning. The better performance of the DDR with predictor-corrector form over the AANN-based DDR is attributed to the fact that process models directly participated in the data reconciliation, whereas, process models indirectly participated in data reconciliation in the AANN-based DDR. Information about the process dynamics contained in the models was lost partly during offline training the AANN.

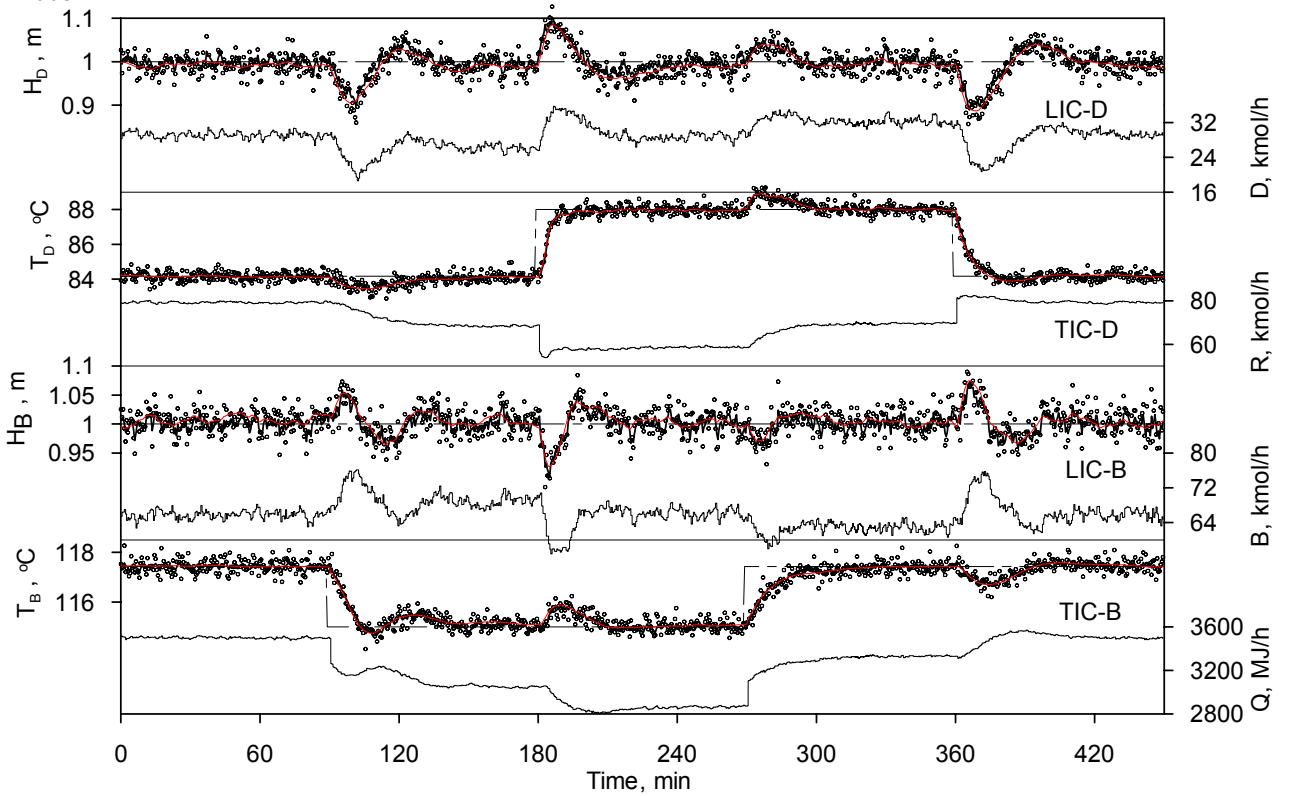

Figure 13 Performance of AANN-based DDR along with controllers when the top and bottom temperature controllers had a series of step setpoint changes. o: raw, ▬: reconciled, —: true, — - —: setpoint.

The effect of online tuning of the DDR gain for the predictor-corrector form on the performance of the control system was investigated. The optimal DDR gain matrix for this dynamic process when the two temperature controllers had step setpoint changes was

$$\mathbf{K} = \begin{array}{c} \begin{matrix} H_D & T_D & H_B & T_B \end{matrix} \\ \begin{bmatrix} 0.19 & 0 & 0 & 0 \\ 0 & 0.26 & 0 & 0 \\ 0 & 0 & 0.20 & 0 \\ 0 & 0 & 0 & 0.24 \end{bmatrix} \end{array} \qquad (35)$$

Then the matrix **K** was multiplied by a factor of c. The DDR with gain matrix c**K** was implemented in the process. The performance of the control system was evaluated as a function of the values of c, and the results are plotted in Figure 14. For comparison, the overall cost function achieved by the AANN-based DDR is also presented in Figure 14. The DDR with predictor-corrector form performed better than the AANN-based DDR over the range of -0.44<log(c)<0.33, which is equivalent to 0.36<c<2.14. When the gain matrix was set away from this region, the AANN-based DDR performed better. These results demonstrate that the AANN-based DDR is more robust in its implementation.

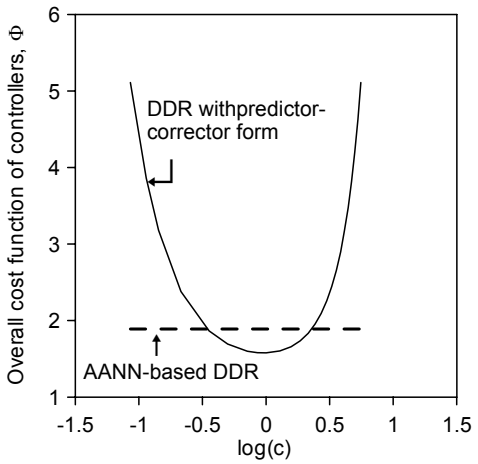

Figure 14 Overall cost function of control system as a function of the DDR gain.

4 Conclusions

The magnitude of measurement noise must be considered when designing controllers using models of the process. Otherwise, the controllers are too aggressive, resulting in excessive variations in the controlled and manipulated variables. On the other hand, the detuning of controllers due to the presence of noise leads to more sluggish responses and lower controller performance. The DDR methods developed in this work was shown to be an effective tool that can be implemented in real-time to reduce the impact of measurement noise before calculating the control action. The application of the DDR filter can result in significantly better feedback controller performance. It allows more aggressive controllers to be used, and at the same time, prevents the manipulated variables from excessive manipulations. The performance of DDR is better than classical EWMA filter because they can anticipate the process dynamics using process models. DDR can use a variety of process models (phenomenological, or empirical, linear, or nonlinear). Accurate process models play an important role for good performance of the DDR. A DDR using simple linear models can successfully attenuate the measurement noise. Further improvement of the DDR performance can be achieved using more comprehensive models that can more efficiently capture the underlying dynamics of the process. DDR can be in a predictor-corrector form, or an AANN-based form. Both forms have their advantages and disadvantages, and can compensate for each other.

Bibliography

[1] Kamen, E.W. and J.K. Su (1999). *Introduction to Optimal Estimation*. Springer, London.

[2] Roffel, B. and P. Chin (1987). *Computer Control in the Process Industries*. Lewis Publishers, Michigan.

[3] Brosilow, C. and B. Joseph (2002). *Techniques of Model-Based Control*. Prentice, New Jersey.

[4] Wilson, D.I.; M. Agarwal and D.W.T. Rippin (1998). Experiences implementing the extended Kalman filter on an industrial batch reactor. *Computers Chem. Engng*. 22, 1653-1672.

[5] Bai, S.; J. Thibault and D.D. McLean (2005). Dynamic data reconciliation: an alternative to Kalman filter. *J. Process Control*, submitted.

[6] Kuehn, D.R. and H. Davidson (1961). Computer control, II Mathematics of control. *Chem. Eng. Prog.*, 57, 44-47.

[7] Bai, S.; D.D. McLean and J. Thibault (2004). Enhancing controller performance via dynamic data reconciliation. *Can. J. Chem. Eng.*, in press.

[8] Binder, T.; L. Blank; W. Dahmen and W. Marquardt (2002). On the regularization of dynamic data reconciliation problems. *J. Process Control*, 12, 557-567.

[9] Bagajewicz, M. and Q. Jiang (1997). Integral approach to plant linear dynamic reconciliation. *AIChE J.*, 43, 2546-2558.

[10] Albuquerque, J. and L. Biegler (1996). Data reconciliation and gross error detection for dynamic systems. *AIChE J.*, 42, 2841-2856.

[11] Hodouin, D. and S. Makni (1996). Real-time reconciliation of mineral processing plant using bilinear material balance equations coupled to empirical dynamic models. *Int. J. Miner. Process.*, 48, 245-264.

[12] Ramamurthi, Y.; P.B. Sistu and B.W. Bequette (1993). Control relevant dynamic data

reconciliation and parameter estimation. *Computers Chem. Engng*, 17, 41-59.
[13] Liebman, M.J.; T.F. Edgar and L.S. Lasdon (1992). Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. *Computers Chem. Engng*, 16, 963-986.
[14] Darouach, M. and M. Zasadzinski (1991). Data reconciliation in generalized linear dynamic systems. *AIChE*, 37, 193-201.
[15] Bai, S. J. Thibault and D.D. McLean (2005). Use of autoassociative neural network for dynamic data reconciliation. 16th IFAC World Congress, Prague, Czech Republic.
[16] Bai, S., J. Thibault and D.D. McLean (2004). Closed-loop data reconciliation for the control of a binary distillation column. *Chem. Eng. Comm*., in press.
[17] Bai, S. D.D. McLean and J. Thibault (2005). Impact of model structure on the performance of dynamic data reconciliation. *Computers Chem. Engng*, submitted.