Computers & Chemical Engineering, Volume 32, Issue 7, 24 July 2008, Pages 1533-1545

Differential Recurrent Neural Network based Predictive Control

R K Al Seyab Yi Cao*

School of Engineering, Cranfield University, UK

Accepted for publication by *Computers and Chemical Engineering* on 19 July 2007

Keywords

Nonlinear Systems, System Identification, Predictive Control, Recurrent Neural Network, Automatic Differentiation.

Abstract

In this paper an efficient algorithm to train general differential recurrent neural network (DRNN) is developed. The trained network can be directly used in the nonlinear model predictive control (NMPC) context. The neural network is represented in a general nonlinear state-space form and used to predict the future dynamic behavior of the nonlinear process in real time. In the new training algorithms, the ODEs of the model and the dynamic sensitivity are solved simultaneously using Taylor series expansion and automatic differentiation (AD) techniques. The same approach is also used to solve the online optimization problem in the predictive controller. The

*To whom correspondence should be addressed. Email:y.cao@cranfield.ac.uk

efficiency and effectiveness of the DRNN training algorithm and the NMPC approach are demonstrated through a two-CSTR case study. A good model fitting for the nonlinear plant at different sampling rates is obtained using the new method. A comparison with other approaches shows that the new algorithm can considerably reduce network training time and improve solution accuracy. The DRNN based NMPC approach results in good control performance under different operating conditions.

1 Introduction

Model predictive control (MPC) strategies have been well received by industry because they are intuitive and can explicitly handle MIMO systems with input and output constraints. Until recently, industrial applications of MPC have relied on linear dynamic models even though most processes are nonlinear. MPC based on linear models is acceptable when the process operates at a single set-point and the primary use of the controller is the rejection of small disturbances. Operating points of modern chemical processes vary over large regions and cannot be modelled adequately using linear models. These conditions are observed in many situations such as, change over in continuous processes, tracking problems in start-up and batch processes and the control of nonlinear reactors. To properly control these processes a nonlinear dynamic process model should be used. Predictive control with a nonlinear model is referred as nonlinear model predictive control or NMPC.

In many cases, nonlinear system identification is an inevitable step in a NMPC project. Possibly, it is also the most costly and time consuming part of the project (Zhao *et al.*, 1998). Therefore, an efficient and effective approach of nonlinear system identification is critical to the success of NMPC. Unlike linear systems, there is no uniform way to parameterize a general nonlinear dynamic system. Among many existing techniques, the universal approximation properties of neural networks makes them a powerful tool for modelling nonlinear systems (Funahashi and Nakamura, 1993). The structure of neural networks may be classified as feed-forward and recurrent. Most of the publications in nonlinear system identification use feed-forward neural network (FFNN) with back-propagation or some other variations for training, for example (Temeng *et al.*, 1995; Tan and Cauwenberghe, 1996). The main drawback of FFNN is that it can only provide predictions for a predetermined finite number of steps, in most cases, only one step. This drawback makes such models not well suited for predictive control, where variable multi-step predictions are desired. A NMPC based on multiple FFNN has been proposed (Jazayeri, 2004). In this approach, a combination of multiple FFNN with one hidden layer are used to model an m-input n-output nonlinear dynamic system. This system consists of a two-dimensional array of FFNN blocks and each block consists of a one-step-ahead predictive neural model, which is identified to represent each output of the MIMO system. These models are employed to predict the future outputs over the prediction horizon of P time steps. This approach might solve the multi-steps ahead prediction problem of the FFNN but it needs the training of a new FFNN for every extension to the prediction horizon, or a large number of networks when a long prediction horizon is needed.

Recurrent neural network (RNN) on the other hand are capable of providing long range predictions even in the presence of measurement noise (Su and McAvoy, 1997). Therefore, RNN models are better suited for NMPC. RNN with internal dynamics has been adopted in several recent works. Models with such networks are shown (Funahashi and Nakamura, 1993; Jin *et al.*, 1995), to have the capability of capturing various plant nonlinearities. They have also been shown to be more efficient than FFNN in terms of the number of neurons required to model a dynamic system of a certain order (Delgado *et al.*, 1995; Hush and Horne, 1993). In addition, they are more suitable to be represented in state-space format, which is quite commonly used in many important control algorithms (Zamarreno and Vega, 1998).

The static FFNN together with tapped-delay lines provides a way to model nonlinear dynamic systems in discrete-time (Miller *et al.*, 1990; Narendra and Parthasarathy, 1990). In general, this type of model is also referred to the nonlinear autoregressive with exogenous inputs (NARX) model or neural network autoregressive with exogenous(NNARX) model (Norgaard *et al.*, 2000).

NNARX model provides a description of the systems in terms of a nonlinear function of delayed input, output, and prediction error. Many MPC based on NNARX models were proposed in the literature (Korenberg and Paarmann, 1991; Mathews, 1991). Recently, (Fabro *et al.*, 2005) developed a fuzzy predictive controller architecture, tuned by genetic algorithms (GA), to the startup control of a distillation column. Recurrent neural network type NNARX was used to identify the nonlinear process and predict its behavior based on control actions applied to the system. They developed also, a constructive algorithm to find the best parameters during the training process. The modified training algorithm alters the delayed connections, inserting new delays when the convergence of the training process could not achieve certain levels of correctness. Also, the same type of RNN was used by (Chu *et al.*, 2004) as an internal model of their proposed NMPC approach, combined feedforward/feedback MPC (CMPC). They named the neural network model (which is the same to NNARX structure) as an external RNN' or (ERN). Two ERNs were used to identify the nonlinear process, a distillation column for ethanol and water mixture (one for the top and the other for the bottom temperatures). Training and testing of the ERN were performed with the toolbox of Matlab (version 6.5, The MathWorks Inc).

The main difficulty with NNARX neural networks or other variants even for SISO systems, is the determination of an appropriate model structure (*i.e.* the number of delays units or (model order)), that best represents the process dynamics. RNN in general state-space model was used by many researchers as the best solution for this problem (Zhao *et al.*, 1998; Zamarreno and Vega, 1998; Kambhampati *et al.*, 2000a).

RNN can be discrete-time neural networks, (Zamarreno and Vega, 1998), or continuous-time (differential) neural network or DRNN, (Funahashi and Nakamura, 1993; Kambhampati *et al.*, 2000a). The continuous-time RNN brings further advantages and computational efficiency over the discrete formulation even if at the end both are represented on the computer using only discrete values (Pearlmutter, 1995). Discrete-time RNN can only work for a particular sampling frequency and no information is given about the model trajectories between the sampling instants. If the sampling frequency is to change, the model has to be re-built. Hence, it is not convenient to use discrete-time RNN for multi-rate control. In contrast, once a continuous-time RNN has been created, it can be used for any sampling frequency (Kambhampati *et al.*, 2000a; Kambhampati *et al.*, 2000b), even for continuous-time NMPC. Although a continuous-time RNN has clear advantages, it has rare been used in NMPC. The main reason probably is due to the difficulty to solve the differential parameter optimisation problem associated with the continuous-time nonlinear model identification problem.

In this work, a continuous time version of recurrent neural networks in state-space form is used as the internal model of NMPC. The general form of nonlinear state-space model has been widely used for control system analysis and design either in white-box or black-box situations. Such a model is capable of capturing full nonlinear dynamics (Zhao *et al.*, 1998).

Neural network training is actually a nonlinear optimization problem. Various training strategies have been suggested in the literature, such as the back-propagation method (Rumelhart *et al.*, 1986), the conjugate gradient method (Leonard and Kramer, 1999), Levenberg-Marquardt optimization (Marquardt, 1963), or methods based on genetic algorithms (Goldberge, 1989). To solve the nonlinear optimization problem associated with DRNN training, the calculation of a large number of dynamic sensitivity equations is required. Depending on the number of sensitivity equations involved, the sensitivity calculation could take more than 90 percent of the total computation time required for solving a training problem. Hence, sensitivity calculation is a bottleneck in training DRNN. Ways to find the sensitivity of a dynamic system (Storen and Hertzberg, 1999) are: perturbation, sensitivity equations, and adjoint equations. In a perturbation approach, finite difference (FD) is used to approximate derivatives. Hence at least N perturbations to the dynamic system are needed to get the solution of a N-parameter sensitivity problem (Storen and Hertzberg, 1999). Alternatively, sensitivity can also be obtained by simultaneously solving the original ordinary differential equations (ODEs) together with nN sensitivity equations, where n is the number of states (Schlegel *et al.*, 2004). Finally, sensitivity can be calculated by solving n adjoint equations (in reverse direction).

Recently, the AD techniques have been applied to tackle the dynamic optimization problem (Griesse and Walther, 2004). In a previous work, (Cao and Al-Seyab, 2003), a first-order approximation was derived using AD to simplify the dynamic sensitivity equations associated with a NMPC problem so that computation efficiency was improved. In most published works of using AD for dynamic optimization, AD has only been used to generate low (first and/or second) order derivatives. Nevertheless, in the new NMPC formulation proposed by Cao (2005), AD has been successfully adopted to produce high-order Taylor coefficients, which significantly improved computation efficiency of the NMPC. In this work, this formulation has been successfully extended to train the DRNN to speed up calculations and to increase efficiency. The developed DRNN can be directly used for NMPC within the above formulation. The training and control algorithms of DRNN are suitable for most process systems. A two-CSTR case study is presented to demonstrate the usage and benefit of the algorithms proposed. With the DRNN model developed, the sampling frequency can been freely altered to improve control performance. The network training time is significantly reduced by using the new algorithm comparing with other methods. Using the trained DRNN as its internal model, the NMPC controller gives satisfactory control performance at different operating conditions.

The paper is organized as follows. In section 2, a training algorithm is proposed using AD techniques. Section 3 presents formulations for predictive control to use a DRNN as internal model with AD techniques. These algorithms are applied to the case study in section 4 and in section 5 some conclusions of the work are provided.

2 DRNN and training

2.1 Model training

Assume a model-unknown continuous-time nonlinear dynamic system has n_u inputs and n_y outputs. N points of input, $u(k), k = 0, \dots, N-1$ and output, $\tilde{y}(k), k = 0, \dots, N-1$ data are collected at sampling rate h. These data are used to train a recurrent neural network (RNN) so that the RNN trained can predict the dynamic behavior of the system with reasonable accuracy. According to the universal approximation theory of artificial neural networks, there are many types of neural networks from multi-layer perceptrons (MLP) to radial basis functions (RBF), which can be constructed as recurrent networks to approximate the nonlinear system. The training algorithm to be discussed is suitable for any kind of networks. Hence, the DRNN to be considered is represented in the following general form.

$$\begin{aligned} \dot{x} &= f(x, u, \theta) \qquad (1) \\ y &= Cx \end{aligned}$$

where $x \in \mathbb{R}^{n_x}$ is the hidden state of the DRNN, $u \in \mathbb{R}^{n_u}$ the input, $y \in \mathbb{R}^{n_y}$ the output, $\theta \in \mathbb{R}^{n_\theta}$ the parameters of DRNN to be trained, and $C = [I\ 0]$, i.e. the outputs are equal to the first n_y states. The collected input data are directly applied to the DRNN by assuming constant input between two sampling instants. The initial state of the DRNN is $x(0) = [\tilde{y}^T(0)\ 0]^T$. Then, for a given set of parameters, θ, output can be predicted by (1). The training algorithm to be proposed aims to minimize the total prediction error:

$$\varphi = \frac{1}{2}\sum_{k=1}^{N} e_k^T e_k = \frac{1}{2}E^T E \tag{2}$$

where $e_k := y(kh) - \tilde{y}(k)$ and $E := [e_1^T\ \cdots\ e_N^T]^T$. To solve this optimization problem efficiently, the gradient, $\varphi_\theta := d\varphi/d\theta$ is to be calculated using the AD techniques described as follows.

Let the state of (1) be a Taylor series up to d terms, i.e. $x(t) = \sum_{k=0}^{d} x_{[k]} t^d$, where $x_{[k]} = \frac{1}{k!}\frac{d^k x}{dt^k}$. Then, the Taylor expansion of f can be directly obtained using AD techniques(Cao, 2005) as $f = \sum_{k=0}^{d} f_{[k]} t^k$ where Taylor coefficients, $f_{[k]}$ are functions of coefficients, $x_{[j]}$, with $j \leq k$, i.e.

$$f_{[k]} = f_{[k]}(x_{[0]}, \ldots, x_{[k]}, u, \theta) \tag{3}$$

Since $\dot{x} = f$, $x_{[k+1]} = \frac{1}{k+1} f_{[k]}(x_{[0]}, \ldots, x_{[k]}, u, \theta)$, i.e. all coefficients, $x_{[k]}$ can be iteratively obtained from $x_{[0]} = x(0)$. Then, at next sampling point, $x(h) = \sum_{i=0}^{d} x_{[i]} h^i$ and it will be used as the initial value for integration of next interval. For output, the Taylor coefficients are $y_{[k]} = Cx_{[k]}$ and $y(h) = Cx(h)$. More importantly, AD can be used to calculate sensitivities of (3)(Christianson, 1992).

$$A_{x[i]} := \frac{\partial f_{[i]}}{\partial x_{[0]}} = \frac{\partial f_{[i+j]}}{\partial x_{[j]}} \tag{4}$$

$$A_{\theta[i]} := \frac{\partial f_{[i]}}{\partial \theta} \tag{5}$$

The total derivatives are iteratively accumulated from (4) and (5) as follows.

$$B_{x|0|} := I \tag{6}$$

$$B_{x|i|} := \frac{dx_{|i|}}{dx_{|0|}} = \frac{\partial x_{|i|}}{\partial x_{|0|}} + \sum_{k=0}^{i-1} \frac{\partial x_{|i|}}{\partial x_{|k|}} \frac{dx_{|k|}}{dx_{|0|}} = \frac{1}{i}\left(A_{x|i-1|} + \sum_{k=0}^{i-1} A_{x|i-k-1|} B_{x|k|}\right) \tag{7}$$

$$B_{\theta|i|} := \frac{dx_{|i|}}{d\theta} = \frac{\partial x_{|i|}}{\partial \theta} + \sum_{k=0}^{i-1} \frac{\partial x_{|i|}}{\partial x_{|k|}} \frac{dx_{|k|}}{d\theta} = \frac{1}{i}\left(A_{\theta|i-1|} + \sum_{k=0}^{i-1} A_{x|i-k-1|} B_{\theta|k|}\right) \tag{8}$$

Equation (8) requires $B_{\theta|0|}$, which is iteratively calculated from previous sampling interval as follows.

$$\begin{aligned} B_{\theta|0|}(0) &= 0 \\ B_{\theta|0|}((k+1)h) &= \frac{dx((k+1)h}{d\theta} = \frac{\partial x((k+1)h}{\partial \theta} + \frac{\partial x((k+1)h}{\partial x(h)}\frac{dx(kh)}{d\theta} \\ &= \sum_{i=0}^{d} B_{\theta|i|}(kh)h^i + \left(\sum_{i=0}^{d} B_{x|i|}(kh)h^i\right) B_{\theta|0|}(kh), \qquad k = 0,\dots,N-1 \end{aligned}$$

Jacobian matrices of (2) are calculated from Taylor coefficients:

$$\begin{aligned} J_k &:= \frac{\partial e_k}{\partial \theta} = \frac{\partial y(kh)}{\partial \theta} = CB_{\theta|0|}(kh) \\ J &:= \frac{\partial E}{\partial \theta} = \begin{bmatrix} J_1^T & \cdots & J_N^T \end{bmatrix}^T \end{aligned}$$

Based on Levenberg-Marquardt algorithm(Marquardt, 1963), the parameters can be iteratively updated as follows.

$$\theta_{k+1} = \theta_k - (J^T J + \lambda I)^{-1} J^T E \tag{9}$$

where λ is determined by the algorithm to make sure the prediction error is reduced at each iterative step.

2.2 Model Validation

Many model validity tests for nonlinear models have been developed (Zhang and Morris, 1999), for example, the Akaike information criterion (AIC), the statistical χ^2 tests, the predicted squared error criterion, and the higher-order correlation tests.

One of common methods of validation is to investigate the residual (prediction errors) by cross validation on a test data set. Here, validation is done by carrying out a number of tests on correlation functions, including auto-correlation function of the residual and cross-correlation function between controls and residuals. If the identified model based on DRNN is adequate, the prediction errors should satisfy the following conditions of high-order correlation tests (Billings and Voon, 1986):

$$\begin{aligned} R_{ee}(\tau) &= E[e(t-\tau)e(t)] = \delta(\tau), \quad \forall\tau & (10) \\ R_{ue}(\tau) &= E[u(t-\tau)e(t)] = 0, \quad \forall\tau & (11) \end{aligned}$$

where $R_{xz}(\tau)$ indicates the cross-correlation function between $x(t)$ and $z(t)$, e is the model residual. These tests look into the cross-correlation amongst model residuals and inputs, and are normalized to be within a range of ± 1 so that the tests are independent of signal amplitude and easy to interpret (Billings and Voon, 1986). The significance of the correlation between variables is indicated by a confidence interval. For a sufficiently large data set with length N, the 95% confidence bounds are approximately $\pm 1.96/\sqrt{N}$. If these correlation tests are satisfied (within the confidence limits) then the model residuals are a random sequence and are not predictable from the model inputs. This provides additional evidence of the validity of the identified model.

3 Predictive control algorithm

The trained DRNN can be directly used as the internal model for a predictive controller. The control algorithm used in this work is based on the general formulation of nonlinear model predictive control using AD proposed in (Cao, 2005). At each sampling instance, a nonlinear least square optimization (Marquardt, 1963) is performed to minimize the performance index represented as the integral of square error and control effort. The Jacobian matrix required by the nonlinear least square optimization is calculated using the algorithm similar to the training algorithm except that the independent variable is control input rather than network parameters. More specifically, the optimization problem to be solved at each sampling instance is as follows:

$$\begin{aligned}
\min_{\substack{\underline{u}\le u_k\le \overline{u}\\ k=0,\dots,M-1}} \varphi &= \frac{1}{2}\sum_{k=1}^{P} e_{y,k}^T Q e_{y,k} + \sum_{k=1}^{M} \Delta u_k^T R \Delta u_k && (12)\\
\text{s.t. } \dot{x}(t) &= f(x(t),u(t)), \quad t\in[t_0,t_P] && (13)\\
y(t) &= Cx(t)+d(t)\\
x(t_0) &= x_0, \quad x_k := x(t_0+kh)\\
d(t) &:= y_m(t_0)-Cx(t_0), \quad t\in[t_0,t_P]\\
u_k &:= u(t_k)=u(t), \quad t\in[t_k,t_{k+1}]\\
e_{y,k} &:= y(t_k)-r_k, \quad k\in[1,P]\\
\Delta u_k &:= u_{k+1}-u_k, \quad k\in[1,M]\\
u_k &= u_{M-1}, \quad k\in[M,P-1]
\end{aligned}$$

where, M and P are the control and prediction horizons respectively, $Q\in\mathbb{R}^{n_y\times n_y}$ and $R\in\mathbb{R}^{n_u\times n_u}$ are the weighting matrices for the output error and the control signal changes respectively, $r_k\in\mathbb{R}^{n_y}$ is the output reference vector at t_k, d is a virtual disturbance estimated at the current time and used to reduce the model-plant mismatch, y_m is the measured output, $\overline{u}$ and $\underline{u}$ are constant vectors determining the input constraints as element-by-element inequalities.

The prediction horizon $[t_0,t_P]$ is divided into P intervals, $t_0,t_1,\cdots,t_P$ with $t_{i+1}=t_i+h_i$ and $\sum_{i=0}^{P-1}h_i=t_P-t_0$. For piecewise constant control, assume the optimal solution to (12) is $u(t) \equiv u(t_k) = u_{|0|}(k)$ for $t_k \leq t \leq t_{k+1}$, $k = 0, \cdots, P-1$. Then, only the solution in the first interval is to be implemented and whole procedure will be repeated at next sampling instant.

Let $v \in \mathbb{R}^{M \times n_u}$ be defined as $v := [u^T_{|0|}(0) \cdots u^T_{|0|}(M-1)]^T$. Problem (12) is a standard nonlinear programming problem (NLP) which can be solved by any modern NLP solvers. To efficiently solve the online optimization problem of the predictive controller the same gradient calculation strategy of the NMPC approach proposed by (Cao, 2005) is used.

A simple method is used to estimate the initial value of the model states required to solve the optimization problem at each sample time. In this method, the new states are updated from the old values using the dynamic equation (13). Also, the state estimate error was reduced further by adding the virtual disturbance d to the output. No terminal penalty is used in this work and a good tuning of h, P, M, Q, and R was found enough to ensure the close-loop stability for the case study in different operation conditions.

4 Case Study

4.1 Two-CSTR Process

A chemical system common to many chemical processing plants, known as a Continuous Stirred Tank Reactor (CSTR), was utilized as a suitable test for many control methods. It suffices to know that the CSTR constituted by a jacketed, perfectly mixed reactor, where an exothermic, first order and irreversible chemical transformation from reactant **A** to product **B** takes place.

DRNN training and predictive control algorithms proposed are applied to a two-CSTR process. A process comprising of two CSTRs (CSTR1 and CSTR2) in series with an intermediate mixer introducing a second feed (Cao, 1995) is investigated. The process is schematically shown in Figure 1.

A full description of the system and a six-state model is available elsewhere (Cao and Yang, 2004). The control problem is to maintain both tank temperatures, $\tilde{y} = [T_{o1}, T_{o2}]^T$, at desired values in regulating and servo control problems. The manipulated variables (MV) are the cooling water flow-rates of two tanks, i.e. $u = [Q_{cw1}, Q_{cw2}]^T$, which corresponds to the second control scheme discussed by Cao and Yang (2004). Both MVs are subject to constraints, $0.05 \le Q_{cw1}, Q_{cw2} \le 0.8$ $[\mathrm{m}^3/\mathrm{s}]$.

4.2 Model identification

Three sets of 600-second input and output data are collected by applying random input signals to the 6-state nonlinear simulation model. The sampling rate of the training data is 0.1 [s], whist the other two sets for validation are sampled at every 0.05 [s] and 0.02 [s] respectively.

The nonlinear dynamic system is approximated by a MLP DRNN shown in Figure 2 and represented as follows:

$$\begin{aligned}\dot{x} &= W_2\sigma_s(W_x x + W_u u + b_1) + b_2\\ y &= Cx\end{aligned}$$

where, $W_x \in \mathbb{R}^{n_h \times n_x}$, $W_u \in \mathbb{R}^{n_h \times n_u}$, and $W_2 \in \mathbb{R}^{n_x \times n_h}$ are connection weights, $b_1 \in \mathbb{R}^{n_h}$ and $b_2 \in \mathbb{R}^{n_x}$ are bias vectors, whilst each element of the vector $\sigma_s(\cdot) \in \mathbb{R}^{n_h}$ represents the sigmoid-*tanh* function as the neural activation function, *i.e.*

$$\sigma_s(n) = \frac{2}{1+e^{-2n}} - 1 \tag{14}$$

The parameter vector is $\theta = \begin{bmatrix}\mathrm{vec}(W_x)^T & \mathrm{vec}(W_u)^T & b_1^T & \mathrm{vec}(W_2)^T & b_2^T\end{bmatrix}^T \in \mathbb{R}^{n_\theta}$, where $n_\theta = n_x \times (n_h + 1) + n_h \times (n_x + n_u + 1)$. For the two-CSTR process, $n_x = 6$ and $n_h = 6$ are selected for the DRNN model and $n_u = 2$. Therefore, $n_\theta = 96$. For one epoch of training, $n_x \times n_\theta \times N = 3456000$ sensitivity variables have to be calculated. This causes an enormous computation load to DRNN training. In a typical situation, sensitivity calculation will take more than 90% of the total training time. Hence, it is really a computation bottleneck to DRNN training. The new algorithm proposed can significantly improve training efficiency as shown in Table 1 by comparison with sensitivity calculation using ODE23 function in MATLAB. In Table 1, the computation time is for one training epoch, whilst the error is the maximum absolute error with respect to a reference solution obtained using ODE23 with a tolerance of 10^{-15}. AD results are obtained using ADOL-C (Griewank *et al.*, 1996) with a mex wrap in MATLAB. The results clearly show that the AD based algorithm can not only reduce computation time by one to two orders of magnitude, but also significantly improve accuracy.

The training and validating data sets (outputs) at sampling time 0.1 [s] is given in Figure 3. The initial values of the first two states of the network were chosen equal to the nominal values of the two tanks output temperature (= 362.995 [K]), while the other four states were set equal to zero. The network capability to approximate the two-CSTR dynamic response at different sampling rates (0.05 [s] and 0.02 [s]) were demonstrated by the validation results shown in Figures 4 and 5 respectively.

The correlation-based model validation results for the two-CSTR model in sampling time 0.1 [s], 0.05 [s], and 0.02 [s] were calculated according to equations (10) and shown in Figures 6 to 8. The dash-dot lines in each plot are the 95% confidence bounds ($\pm 1.96/\sqrt{600}$). It can be seen that only a small number of points are slightly outside the given bounds. This demonstrates that the model can be considered as being adequate for modelling this plant.

The trained network is able to predict dynamic behavior of the plant at different sampling rates with reasonable accuracy.

4.3 Predictive control

The control objective in the two-CSTR process is to maintain both tank temperatures, T_{o1} and T_{o2} at the desired values in the presence of

1. Cooling-water temperature ± 10 [K] fluctuations in T_{CW1} and T_{CW2} in the presence of actuator constraints.

2. Set-points change in the two output variables in the presence of actuator constraints.

A multi-loop PI controller was designed to control the process at the above regulating problem (Cao and Biss, 1996). The controller was successful in rejecting non-zero mean disturbances (± 10 [K] in T_{cw1} and T_{cw2}), but had somehow a long settling time and high peaks as shown in figures 9 and 10 respectively.

Cao and Yang (2004), designed a linear optimal controller using the H_2 and H_∞ norm to control the process at the disturbance rejection test above. The linear optimal controller was able to reject the disturbance effects in a short time with some small peaks compared with the PI controller (Cao and Yang, 2004). Set-point tracking tests were not included in the two works above.

In this work, the proposed nonlinear predictive controller is used to control the two-CSTR process at regulating and servo problems given above. The trained DRNN in subsection 4.2 is used as the internal model of the predictive controller.

The NMPC parameters are tuned as follows. The cost function is weighted by output weights, $[150, 100]$ and input weights, $[1, 0.5]$ respectively. To tune control horizon M, prediction horizon P, and sampling time h, initially set $h = 0.1$ [s], and $M = 1$ sampling interval, and $P = M = 1$ sampling interval. By varying P from 1 to 20 sampling intervals, a stable performance is obtained which satisfies all control specifications for $5 \leq P$. In fact, nominal stability is strongly affected by the prediction horizon length. However, the advantages of longer P are outweighed by the increase in computation time and result in more aggressive control moves (Henson, 1998). When $P \geq 13$ sampling intervals, the improvement on the system performance is negligible but computation time increases. Therefore $P = 10$ sampling intervals is selected to ensure that both the system stability and satisfactory control performance achieved within a reasonable computation time. The same steps are used to choose a suitable control horizon M, a reasonable range from the minimum value ($M = 1$) to 5 sampling intervals has been tested. A stable response without any constraints violation is detected within range $1 \leq M \leq 4$ sampling intervals. No performance improvement can be observed when $M \geq 3$ sampling intervals. Therefore $M = 1$ is chosen to provide a balance between performance and computation.

Simulation results shown in Figures 9 to 10 clearly indicate that the predictive controller with the DRNN model successfully achieves satisfactory performance without violating input constraints in the regulating control problem. The NMPC response also shows a short settling time with small peaks compared with the PI controller. The NMPC performance when $M = 1$ and $P = 5$ was approximately similar to that of the linear optimal controller (Cao and Yang, 2004) (see Figure 11). In the case of $M = 1$ and $P = 10$, the linear controller was faster than the NMPC. In order to getting more smooth MVs response in both regulating and set-point tracking problems, the second set of M and P has been chosen for all tests.

In the set-point tracking test, the performance of both outputs and MVs using the proposed DRNN based NMPC algorithm was the best comparing with the other two linear controllers as shown in Figures 12 and 13 respectively. To get a clear picture about the MVs behavior, Figure 14 shows a magnified plot for MVs response (part of Figure 13) during T_{o1} and T_{o2} set-point change from 367.5 $[K]$ to 365.995 $[K]$.

To test the controller sensitivity to the sampling time, simulations have also been done by varying h from 0.01 [s] to 0.3 [s]. A stable performance is detected in the range $0.02 \leq h \leq 0.25$ [s]. Figure 15 shows the two-CSTR performance at unmeasured disturbance rejection test using different values of the sampling time. Note that, the DRNN (trained at sampling time 0.1 [s]) is used as the internal model of the predictive controller during these tests. In fact, small sampling period generally improve performance but require a longer prediction horizon to adequately capture the process dynamics which means an increase in the online computation time. On the other hand, a large sampling period reduces online computation, but it can result in poor performance such as *ringing* between sample points (Henson, 1998). This fact has been proved by the results given in this work as shown in Figure 15. Sampling time is chosen to be 0.1 [s] for the best performance and less computation time.

5 Conclusions

This paper demonstrates the reliability of artificial neural networks in process control. An efficient algorithm has been proposed to train differential (continuous-time) recurrent neural networks to approximate nonlinear dynamic systems so that the trained network can be used as the internal model for a nonlinear predictive controller. The new training algorithm is based on the efficient Levenberg-Marquardt method combined with an efficient and accurate tool: automatic differentiation. The dynamic sensitivity equations and the ODEs of the recurrent neural network are solved accurately and simultaneously via AD. Big time saving to solve sensitivity equations with a higher accuracy are observed using the new algorithm compared with a traditional method. Also, the trained network shows the capability to approximated the multivariable nonlinear plant at different sampling time without the need to re-train the networks. Based on the identified neural network model, a NMPC controller has been developed. The similar strategy that used in the network training has been used to solve the online optimization problem of the predictive controller. The capability of the new nonlinear identification algorithm and NMPC algorithm are demonstrated through the two-CSTR case study with satisfactory results.

References

Billings, S. A. and Voon, W. S. F. (1986). Correlation based model validity tests for nonlinear models. *Int. J. Control* **44**, 235 244.

Cao, Y. (1995). Control structure selection for chemical process using input-output controllability analysis. PhD Thesis, University of Exeter.

Cao, Yi (2005). A formulation of nonlinear model predictive control using automatic differentiation. *Journal of Process Control* **15**(8), 851 858.

Cao, Y. and Al-Seyab, R. (2003). Nonlinear Model Predictive Control using Automatic Differentiation, *European Control Conference (ECC 2003)*, Cambridge, UK, p. in CDROM.

Cao, Y. and Biss, D. (1996), An extension of singular value analysis for assessing manipulated variable constraints. *J. Process Control*, **6**(1), 37–48.

Cao, Y. and Z. J. Yang (2004). Muiltobjective process controllability analysis. *Computers and Chemical Engineering* **28**(1 2), 83 90.

Christianson, B. (1992). Reverse accumulation and accurate rounding error estimates for Taylor series.. *Optimization Methods and Software* **1**, 81–94.

Chu, J. Z., Jang, S. S., and Chen, Y. N. (2004). A comparative study of combined Feedforward/Feedback model predictive control for nonlinear systems. *Canadian Journal of Chemical Engineering* **82**(6), 1263 1272.

Delgado, A., Kambhampati, C., Warwick, K. (1995). Dynamic recurrent neural network for system identification and control. *IEE Proceedings – Control and Applications*, **142**(4), 307-314.

Fabro, J. A., Arruda, L. V.R., and Jr., F. Neves (2005). Startup of a distillation column using intelligent control techniques. *Computers and Chemical Engineering* **30**, 309–320.

Funahashi, K. L. and Nakamura, Y. (1993). Approximation of dynamical systems by continues time recurrent neural networks. *Neural Networks* **6**, 183–192.

Goldberge, D. E. (1989). Genetic algorithms in search, optimization and machine learning, *Reading, MA:Addision-Wesley.*

Griesse, R. and Walther, A. (2004). Evaluating Gradients in Optimal Control: Continuous Adjoint Versus Automatic Differentiation, *Journal of Optimization Theory and Applications*, **122**(1), 63 86.

Griewank, Andreas, David Juedes and Jean Utke (1996). ADOL-C: A package for the automatic differentiation of algorithms written in C/C++. *ACM Transactions on Mathematical Software* **22**, 131 167.

Henson, M. A. (1998). Nonlinear model predictive control : current states and future directions. *Computer and Chem. Eng.*, **23**, 187 202.

Hush, D. R. and Horne, B. G. (1993). Progress in supervised neural networks. *IEEE Sig. Process, Mag.*, **1**, 8–39.

Jazayeri-Rad H. (2004). The nonlinear model predictive control of a chemical plant using multiple neural networks. *Neural Computer and Application* **13**, 2–15.

Jin L., Nikiforuk, P. , and Gupta, M. (1995). Approximation of discrete-time state-space trajectories using dynamic recurrent neural networks. *IEEE Transactions on Automatic Control*, **40**(7), 1266-1270.

Kambhampati, C., Garces, F., and Warwick, K. (2000a). Approximation of non-autonomous dynamic systems by continuous time recurrent neural networks. *Proceeding of the IEEE-INNS-ENNS International Joint Conference on Neural Networks IJCNN 2000*, **1**, 64–69.

Kambhampati, C., Craddock, R. J., Tham, M., Warwick, K. (2000b). Inverse model control using recurrent networks. *Mathematics and Computers in Simulation*, **51**, 181-199.

Korenberg, M. J. and Paarmann, L. D. (1991). Orthogonal approaches to time-series analysis and system identification. *IEEE Signal Proc. Mag.*, **7**, 29–43.

Leonard, J. A. and Kramer, M. A. (1999). Improvement of the back-propagation algorithm for trainig neural networks, *Chemical Eng.*, **14**, 337–341.

Marquardt, D. (1963). An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.* **11**, 431–441.

Mathews V. J. (1991). Adaptive polynomial filters. *IEEE Signal Proc. Mag.*, **7**, 10–26.

Miller, W. T., Sutton, R. S., and Werbos, P. J. (1990). Neural networks for control, MIT Press, Cambridge, MA.

Narendra, K. S. and Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans. Neural Networks*,**1**(1), 4–26.

Norgaard, M., Ravn, O., and Poulsen N. K. (2000). Neural networks for modeling and control of dynamic systems–A practitioners Handbook. *Springer-Verlag.*

Pearlmutter, B. A. (1995). Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Trans. on Neural Networks.*

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representiations by error propagation, *in Prallel Distributed Processing, D. E. Rumelhart and J. L. MeClelland, Eds.*, Cambridge MA: MIT Press.

Schlegel, M., Marquardt, W., Ehrig, R., Nowak, U. (2004). Sensitivity Analysis of Linearly Implicit Differential-Algebraic Systems by One-step Extrapolation, *Applied Numerical Mathematics*, **48**, 83 – 102.

Storen, S. and Hertzberg, T. (1999). Obtaining sensitivity information in dynamic optimization problems solved by the sequential approach, *Computer and Chemical Engineering*, **23**, 807 819.

Su, H. T. and McAvoy (1997). Artificial neural networks for nonlinear process identification and control, In: Henson, M. A. (1998) & Seborg, D.E. (Eds). Nonlinear Process Control, Chap. 7, 371–428, Englewood Cliffs, NJ: Prentice Hall.

Tan, Y., and Cauwenberghe, A. (1996). Nonlinear one step ahead control using neural networks: control strategy and stability design. *Automatica* **32**(12), 1701 1706.

Temeng, H., Schenelle, P., and, McAvoy T. (1995). Model predictive control of an industrial packed reactors using neural networks. *Journal of Process Control* **5**(1), 19 28.

Zamarreno, J. M. and Vega, P. (1998). State-space neural network, properties and application. *Neural Networks*, **11**, 1099–1112.

Zhang, J. and Morris, J. (1999). Recurrent neuro-fuzzy networks for nonlinear process modeling. *IEEE Trans. on Neural Networks* **10**(2), 313 326.

Zhao, H., Guiver, J., and Sentoni, G. (1998). An identification approach to nonlinear state space model for industrial multivariable model predictive control. *Proceeding of the American Control Conference, Philadelphia, Pennsylvania.*

Table 1: DRNN Training, Computing Time and Accuracy Comparison

Traditional Sensitivity Approach			ADOL-C		
Tolerance	Time, [ms]	Error	Actual Order	Time, [ms]	Error
10^{-3}	40.61	1.5899	3	2.437	0.001
10^{-6}	162.281	0.0738	6	4.078	1.562e-7
10^{-8}	272.657	4.551e-4	8	5.391	2.606e-10
10^{-10}	316.375	1.864e-6	10	6.937	1.0729e-12

List of Figures

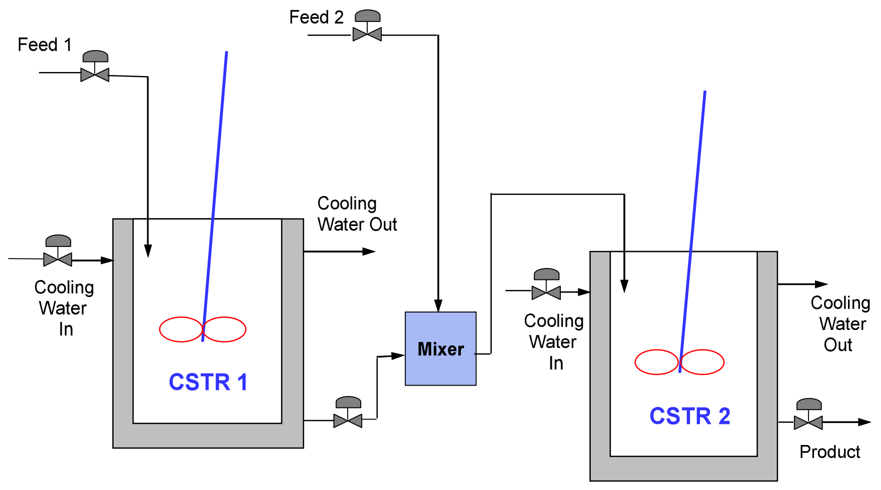

Figure 1: Two-CSTR plant

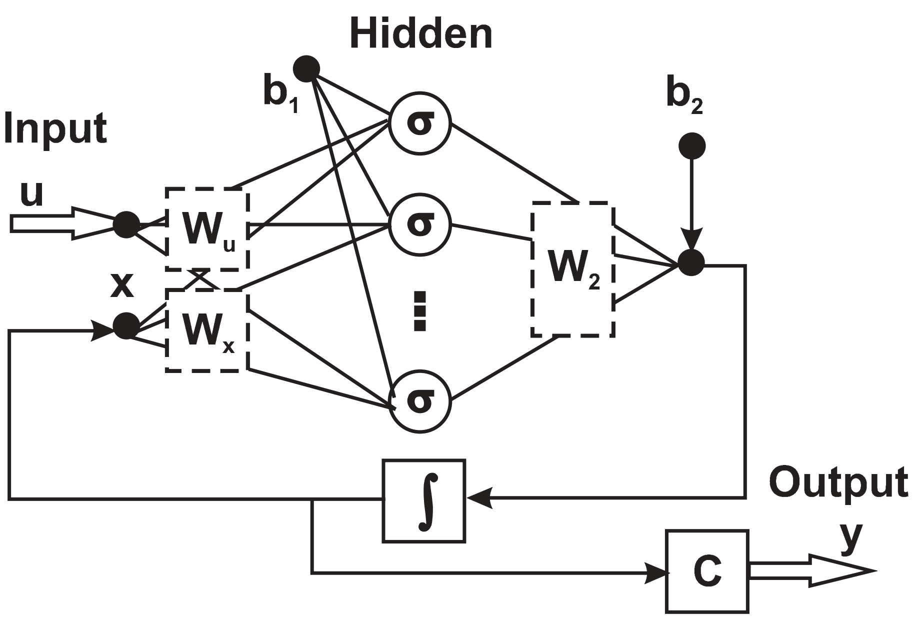

Figure 2: Structure of differential recurrent neural network

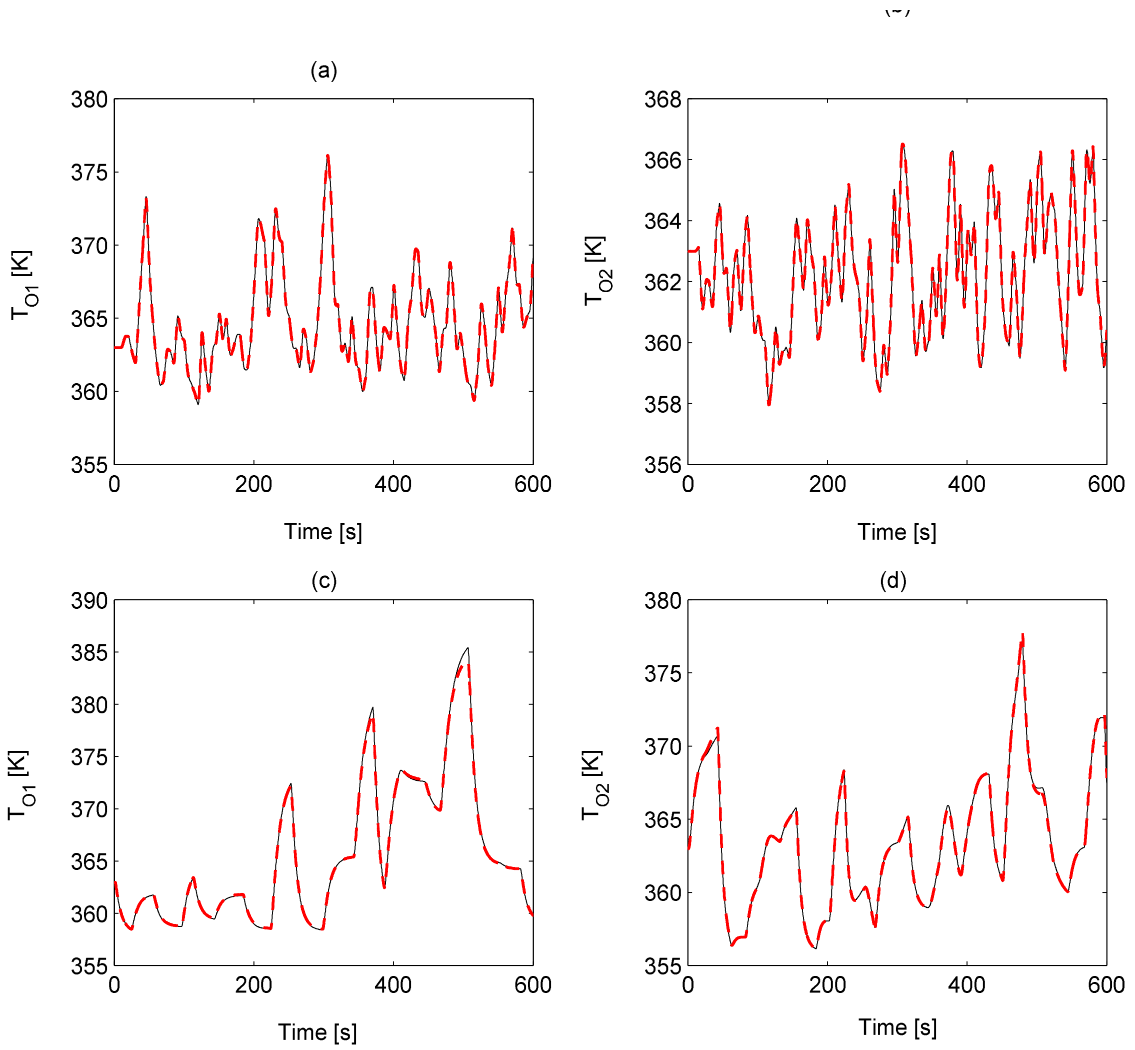

Figure 3: Plant data (solid) and prediction (dashed) comparison. (a) and (b) are training data. (c) and (d) are validation data, $\triangle t = 0.1$ [s].

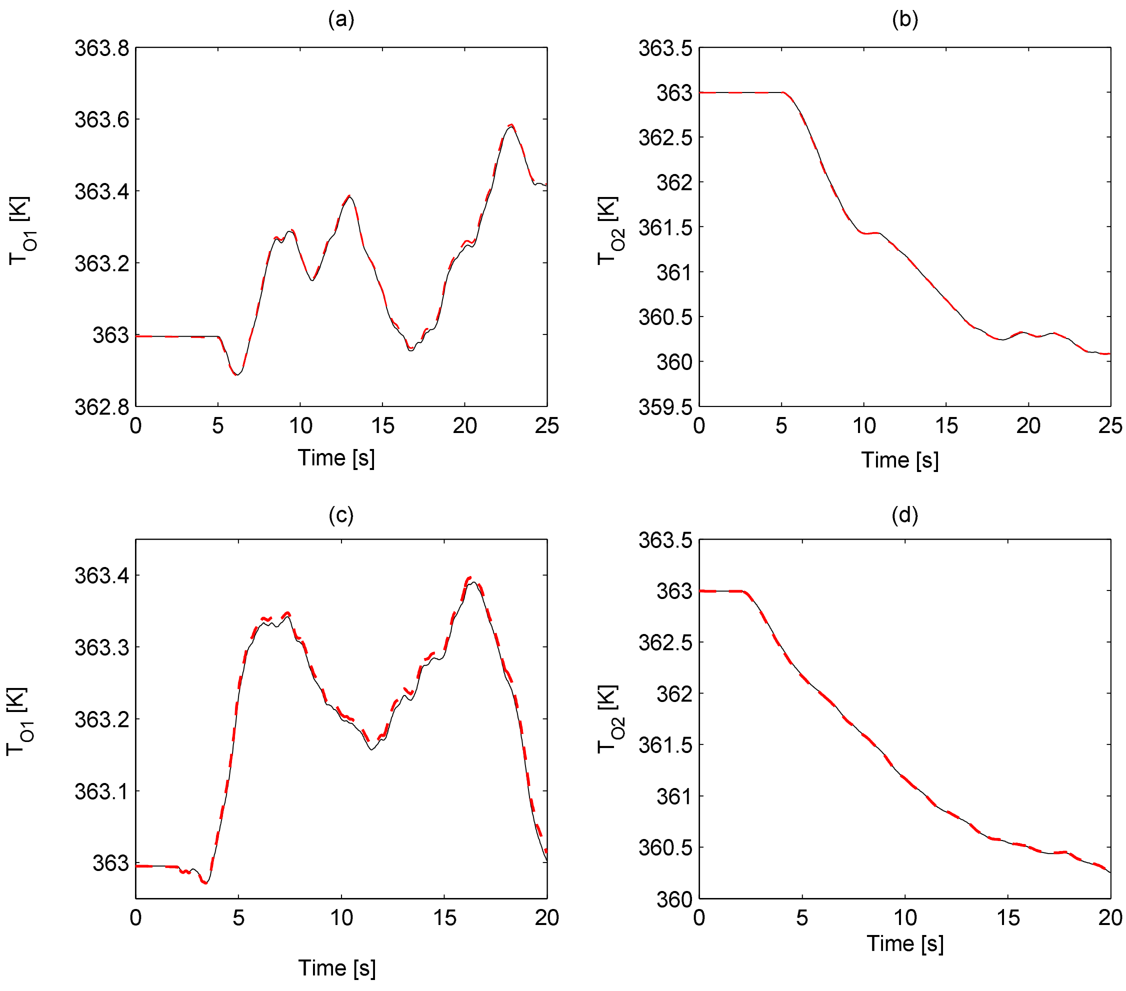

Figure 4: Plant data (solid) and prediction (dashed) comparison. (a) and (b) are validation data, $\triangle t = 0.05$ [s]. (c) and (d) are validation data, $\triangle t = 0.02$ [s].

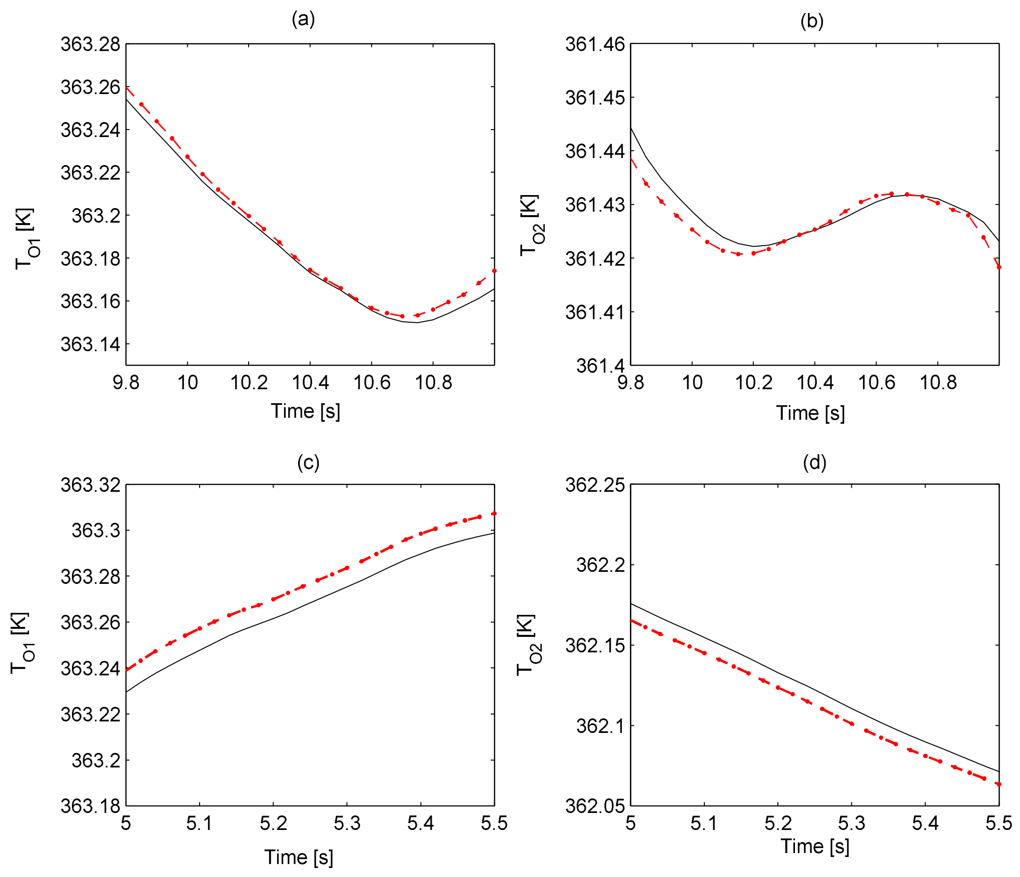

Figure 5: Plant data (solid) and prediction (dashed) comparison (magnified part of Figure 4). (a) and (b) are validation data, $\triangle t = 0.05$ s . (c) and (d) are validation data, $\triangle t = 0.02$ |s|.

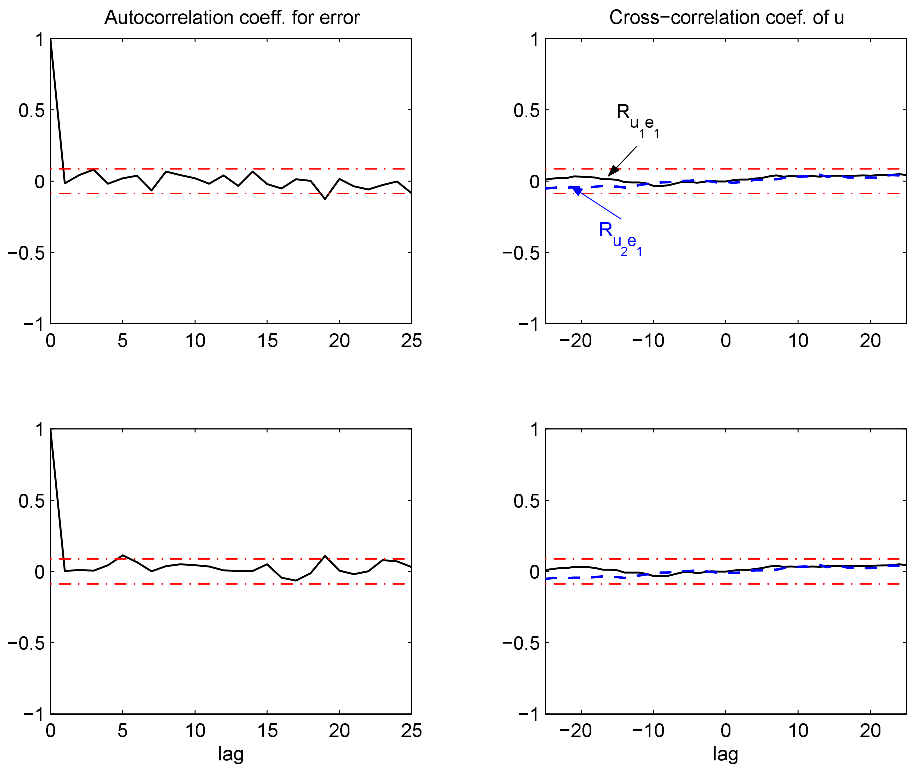

Figure 6: Validating test, $\triangle t = 0.1$ [s].

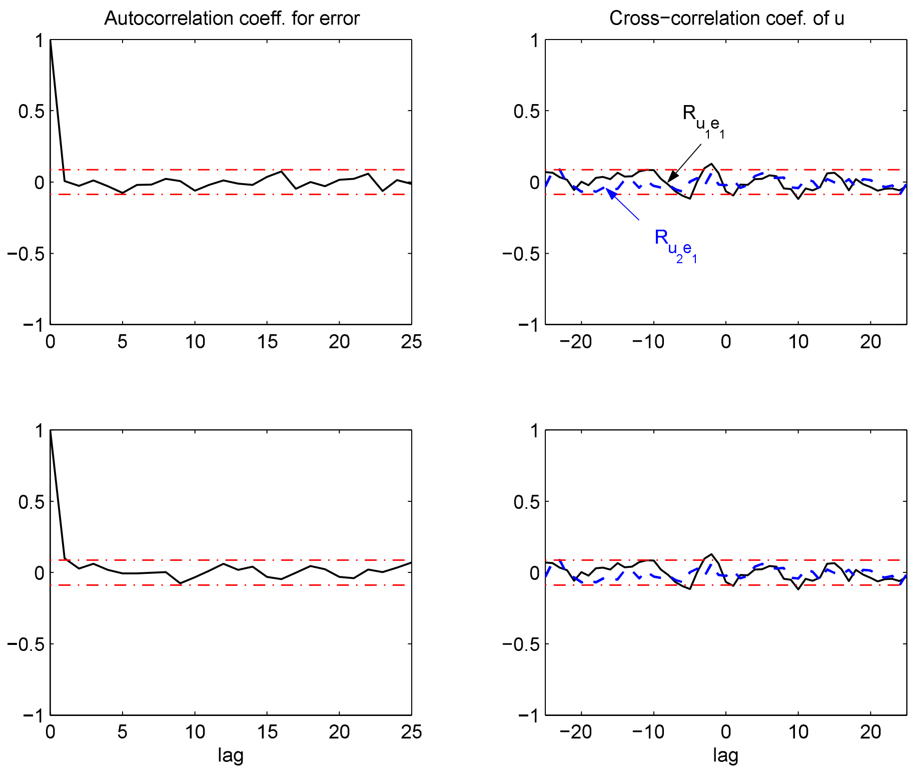

Figure 7: Validating test, $\triangle t = 0.05$ [s].

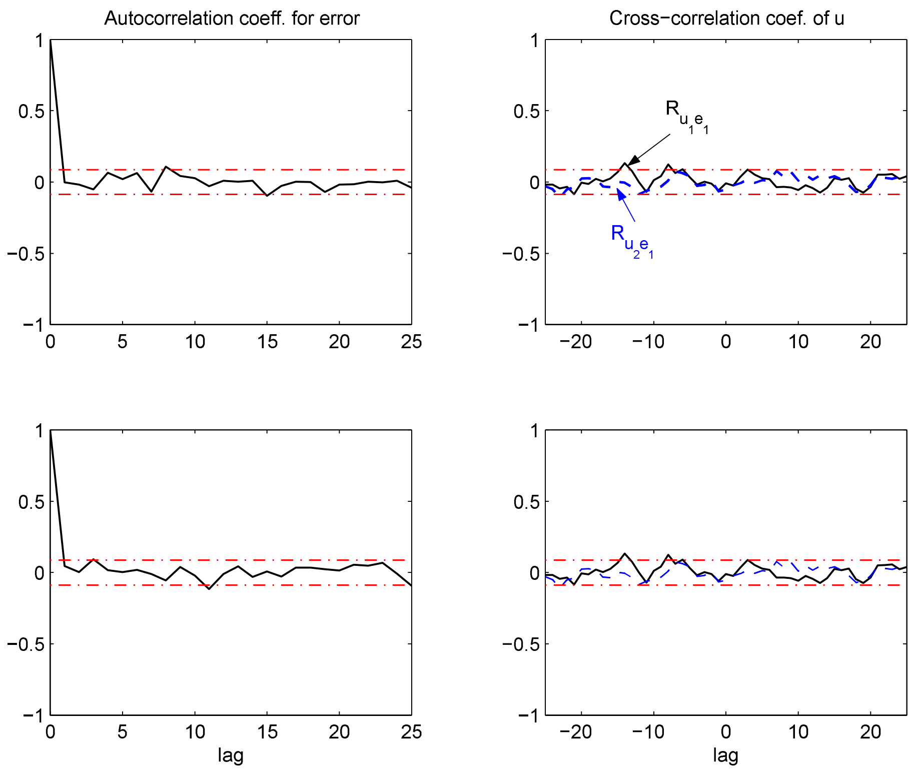

Figure 8: Validating test, $\triangle t = 0.02$ [s].

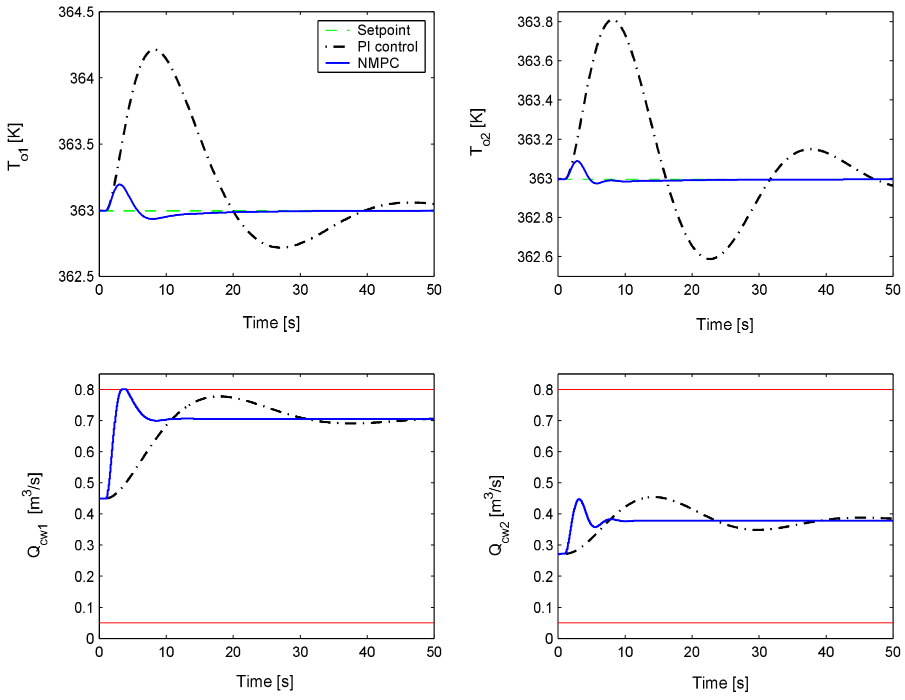

Figure 9: Two-CSTR performance during unmeasured disturbance rejection test (+10 [K] step change in T_{cw1} and T_{cw2} at $t = 2$ [s]).

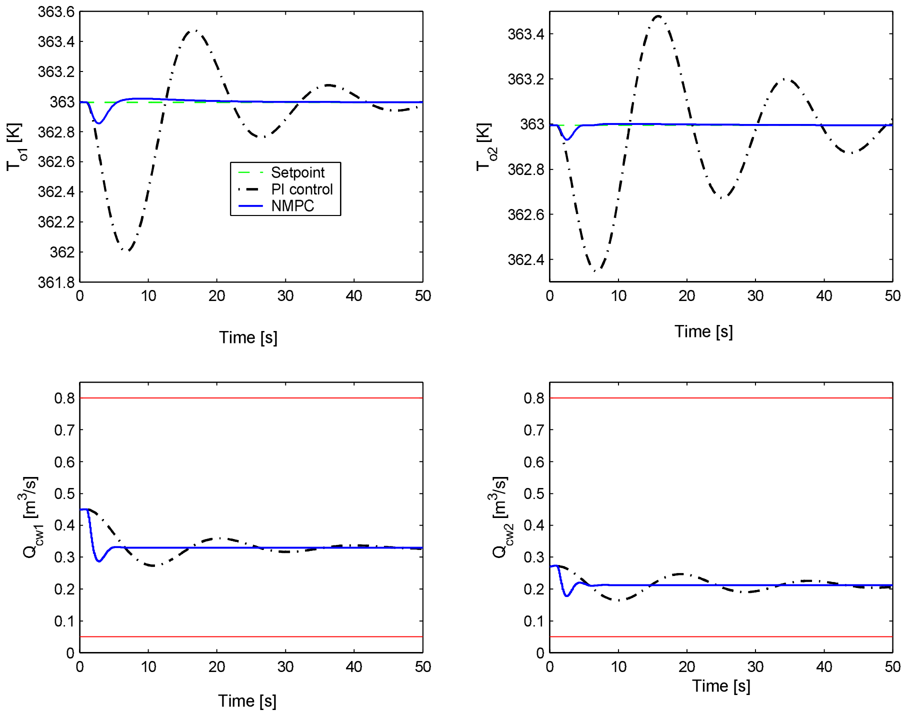

Figure 10: Two-CSTR performance during unmeasured disturbance rejection test (-10 [K] step change in T_{cw1} and T_{cw2} at $t = 2$ [s]).

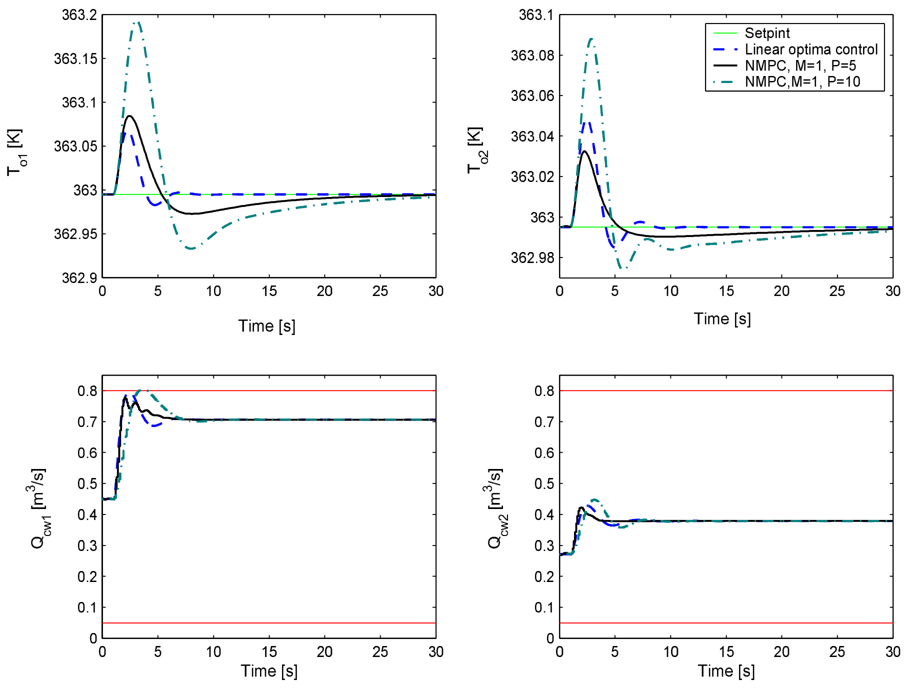

Figure 11: Two-CSTR performance during unmeasured disturbance rejection test (-10 [K] step change in T_{cw1} and T_{cw2} at $t = 2$ [s]).

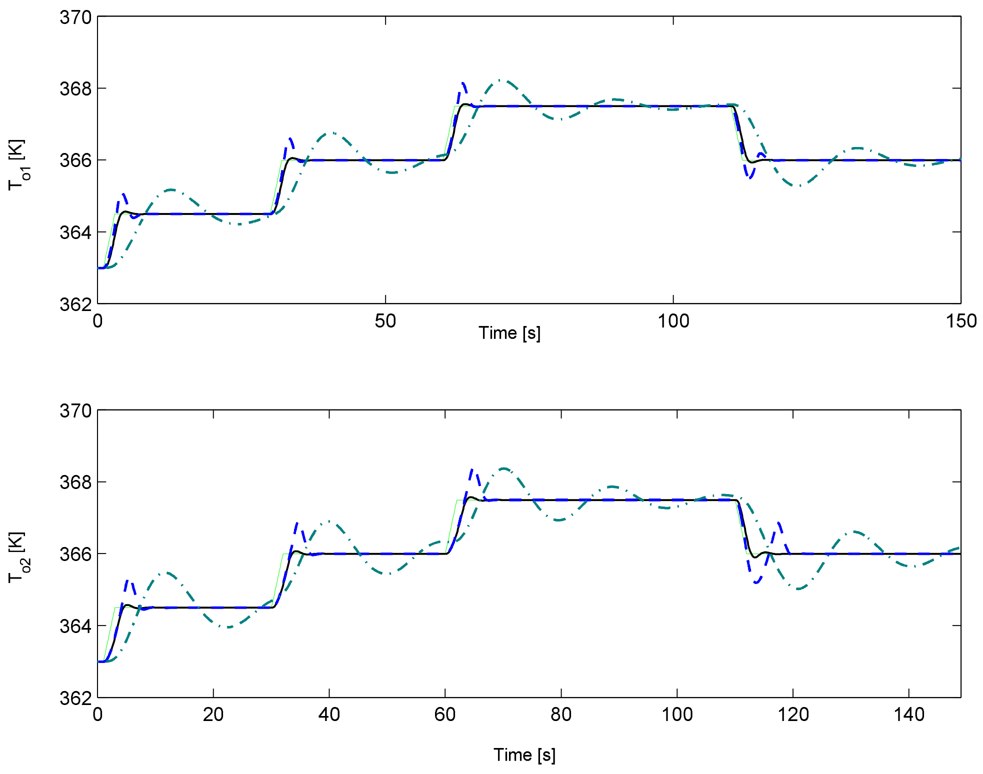

Figure 12: Two-CSTR performance during set-point tracking test, Outputs; NMPC (solid), Linear optimal controller (dash), PI (dash-dot).

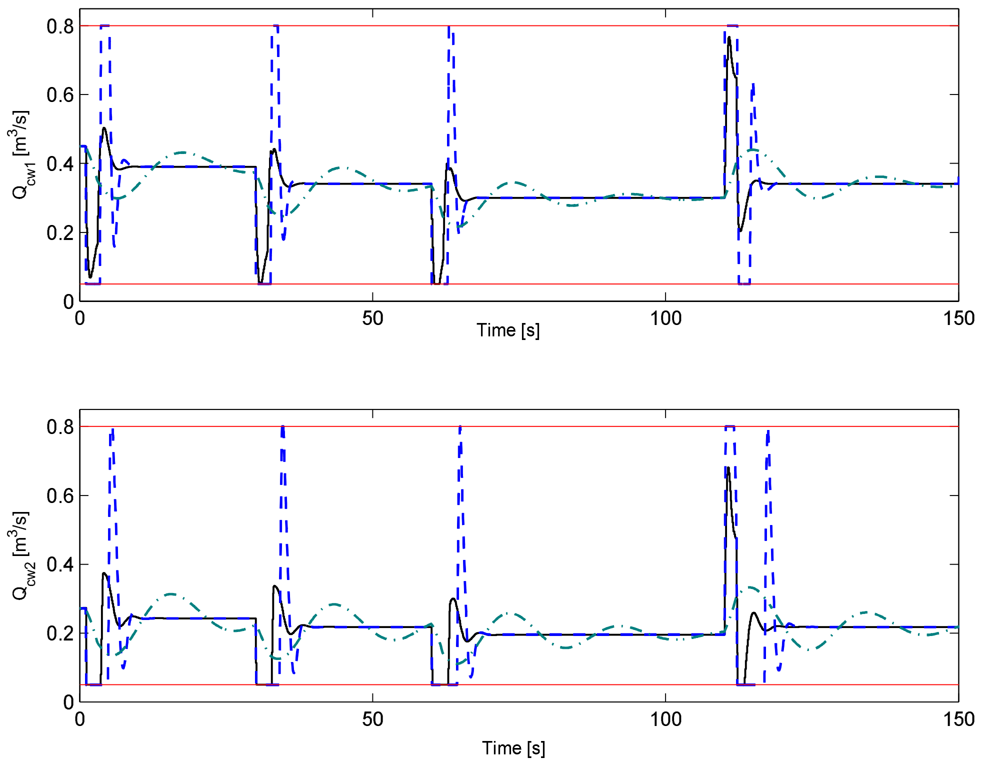

Figure 13: Two-CSTR performance during set-point tracking test, Inputs. NMPC (solid), Linear optimal controller (dash), PI (dash-dot).

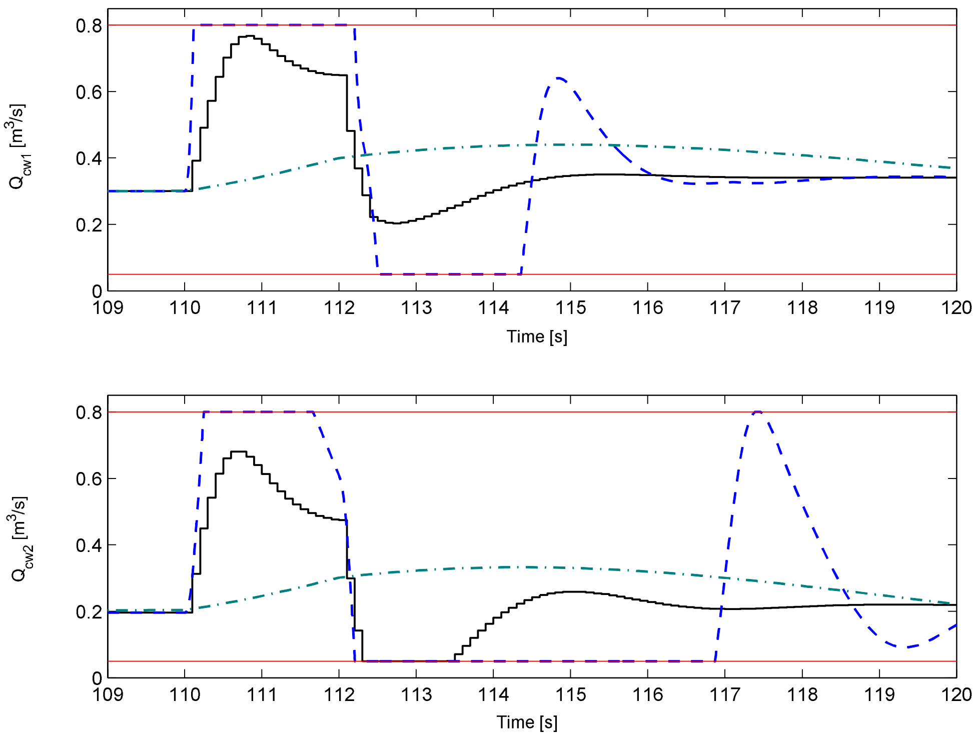

Figure 14: (Magnified part of Figure 13)Two-CSTR performance during set-point tracking test, Inputs. NMPC (solid), Linear optimal controller (dash), PI (dash-dot).

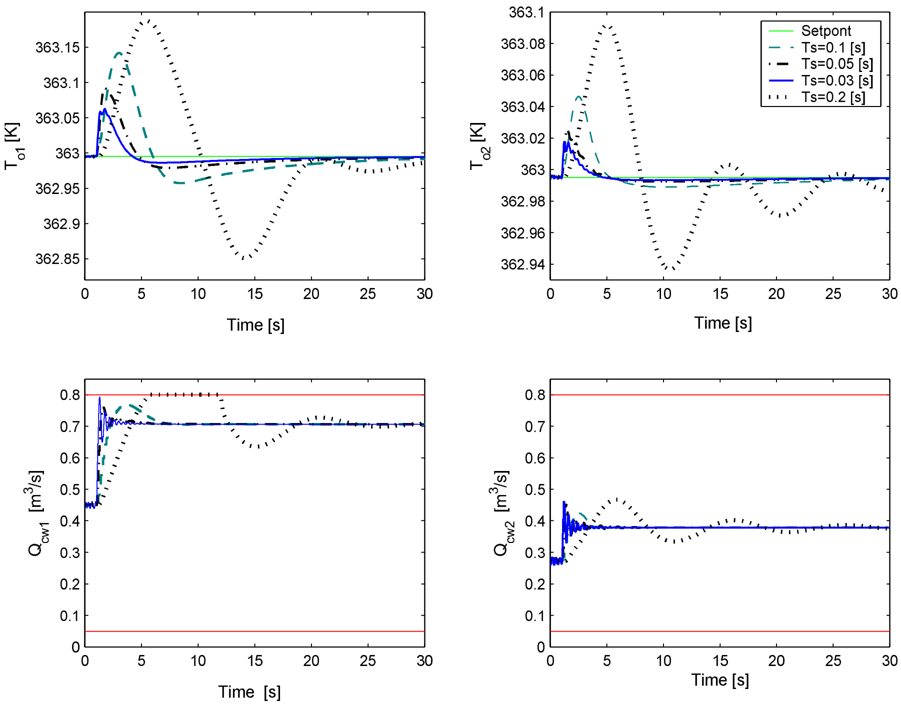

Figure 15: Two-CSTR performance using DRNN based NMPC algorithm during unmeasured disturbance rejection test (+10 [K] step change in T_{cw1} and T_{cw2} at $t = 2$ [s]).