SSY280 Model predictive control

Assignment 2

Steady state targets and disturbance modeling

Group 20

Dandan Ge

Fabian Melvås

January 2017

1 Introduction

This report is about obtaining zero off-set when applying MPC control to a MIMO plant, and using this knowledge to control a chemical reactor model. An offset-free controller is one that drives controlled outputs to their desired targets at steady state. In the linear model predictive control (MPC) framework, offset-free control is usually achieved by adding step disturbances to the process model.

2 Steady-state targets

In this section, we consider the following system with two inputs and two outputs:

$$A = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0.6 \end{bmatrix}, B = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.4 \\ 0.25 & 0 \\ 0 & 0.6 \end{bmatrix}, C = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

And the task is to find the setpoint targets $(x_s,\ u_s,\ z_s)$.

2.1 Task a) Both manipulated input are used for control

In this case the system inputs, m is equal to the number of outputs,p $p = m$ and the output setpoint is defined, $z_{sp} = [1\ \ -1]^T$. The given C matrix has dimension (2×4) and thus the output vector y has dimension (2×1). Since the qiven setpoint vector with controlled outputs z_{sp} has dimension (2×1) to $z_{sp} = y_{sp}$ and $C = C_z$ for this case. To find the steady state targets the system in 1 is solved.

$$\underbrace{\begin{bmatrix} I - A & -B \\ C_z & 0 \end{bmatrix}}_{A_{eq}} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ z_{sp} \end{bmatrix}}_{b_eq} \tag{1}$$

Solve the system of linear equations in MATLAB either using $A_{eq} \setminus b_{eq}$ or the function 'linsolve' to find the steady state targets.
The steady state target is:

$$x_s = \begin{bmatrix} 2.5 \\ -1.5 \\ 1.25 \\ -2.25 \end{bmatrix} \qquad u_s = \begin{bmatrix} 2.5 \\ -1.5 \end{bmatrix} \qquad z_s = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

In order to check if the output setpoint is possible to attain, $HCx_s = C_z x_s = z_{sp} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ must be fulfilled. Since $C = C_z$ the 'selection matrix' $H = I_4$ and then it is fulfilled, when assuming the controlled outputs are a selection of the measured outputs.

2.2 Task b) Only the first manipulated input is available for control

In this case only the first manipulated input is available for control, so the number of outputs p are more than the number of inputs m, $(p > m)$. Then there are more equations than unknown and is an overdetermined problem witch in general has no solutions, since there is not a full row rank. If there is any solution, it means that some equations are linear combinations of others. Which isn't the case for this task so the output setpoint in (a) is not attainable, and we can not get the target setpoint in this case.

It is possible though to get a solution using least square for this problem. Then the solution that has the smallest error is attained by minimizing the difference between Cx_s and setpoint y_{sp}. This is done below by solving the following optimization problem using $Q_s = I$ and $R_s = 0$:

$$\min_{x_s,u_s} \quad (|Cx_s - y_{sp}|^2_{Q_s}), \quad Q_s \succeq 0 \tag{2}$$

$$\text{s.t.} \quad \underbrace{\begin{bmatrix} I-A & -B \end{bmatrix}}_{A_{eq}} \underbrace{\begin{bmatrix} x_s \\ u_s \end{bmatrix}}_{z} = \underbrace{0}_{b_{eq}} \tag{3}$$

Rewrite the optimization problem on form:

$$\min_{z} \quad \frac{1}{2} z^T H_M z + f^T z \tag{4}$$

$$\text{s.t.} \quad A_{eq} z = b_{eq} \tag{5}$$

With $z = \begin{bmatrix} x_{s_1} & x_{s_2} & x_{s_3} & x_{s_4} & u_{s_1} \end{bmatrix}^T$

$$\Rightarrow \min_{x_s,u_s} \quad (\left|Cx_s - y_{s_p}\right|^2_{Q_s}) \tag{6}$$

$$= \min_{x_s,u_s} \quad ((Cx_s - y_{s_p})^T Q_s (Cx_s - y_{s_p})) \tag{7}$$

$$= \min_{x_s,u_s} \quad (x_s^T \underbrace{C^T Q C}_{2\bar{H}_x} x_s \underbrace{-2y_{s_p}^T Q C}_{f_x^T} x_s + \underbrace{y_{s_p}^T Q y_{s_p}}_{constant}) \tag{8}$$

Then the we get the Hessian matrix H:

$$H_M = \begin{bmatrix} 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, f = \begin{bmatrix} -2 \\ -2 \\ 2 \\ 2 \\ 0 \end{bmatrix}$$

$$A_{e_q} = \begin{bmatrix} 0.5 & 0 & 0 & 0 & -0.5 \\ 0 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & -0.25 \\ 0 & 0 & 0 & 0.4 & 0 \end{bmatrix}, b_{e_q} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Then using the MATLAB function '`quadprog`' the target setpoints that is optimal for this case is obtained:

$$x_s = \begin{bmatrix} 0.4 \\ 0 \\ 0.2 \\ 0 \end{bmatrix}, \qquad u_s = 0.4, \qquad z_s = \begin{bmatrix} 0.4 \\ 0.2 \end{bmatrix}$$

2.3 Task c) Only the first output has a set point

In this case both manipulated inputs are available for control, but only the first output has a setpoint, so the number of inputs, m are more than the number of outputs, p ($p < m$).
To find the best steady state targets, we need to solve the following optimization problem using ($Q_s = 0$, $R_s = I$), $u_{s_p} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$:

$$\min_{x_s,u_s} \quad (\left|u_s - u_{s_p}\right|^2_{R_s}) \tag{9}$$

$$\text{s.t.} \quad \begin{bmatrix} I-A & -B \\ C_z & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} 0 \\ z_{s_p} \end{bmatrix} \tag{10}$$

Rewrite the optimization problem on form :

$$\min_{z} \quad \frac{1}{2} z^T H_M z + f^T z \tag{11}$$

$$\text{s.t.} \quad A_{eq} z = b_{eq} \tag{12}$$

With $z = \begin{bmatrix} x_{s_1} & x_{s_2} & x_{s_3} & x_{s_4} & u_{s_1} & u_{s_2} \end{bmatrix}^T$

$$\min_{x_s,u_s} \quad (\left|u_s - u_{s_p}\right|^2_{R_s}) \tag{13}$$

$$= \min_{x_s,u_s} \quad ((u_s - u_{s_p})^T R_s (u_s - u_{s_p})) \tag{14}$$

$$= \min_{x_s,u_s} \quad (u_s^T \underbrace{R_s}_{2\bar{H}_u} u_s \underbrace{-2u_{s_p} R_s}_{f_u^T} u_s + \underbrace{u_{s_p}^T R_s u_{s_p}}_{constant} \tag{15}$$

Then the Hessian matrix should be:

$$H_M = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}, f = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$A_{e_q} = \begin{bmatrix} 0.5 & 0 & 0 & 0 & -0.5 & 0 \\ 0 & 0.4 & 0 & 0 & 0 & -0.4 \\ 0 & 0 & 0.5 & 0 & -0.25 & 0 \\ 0 & 0 & 0 & 0.4 & 0 & -0.6 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, b_{e_q} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The target setpoints are:

$$x_s = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.25 \\ 0.75 \end{bmatrix}, \qquad u_s = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \qquad z_s = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Comments: Here the number of inputs are more than the number of outputs, so there are fewer equations than unknown, so this is an undetermined problem. An undetermined linear system has either no solution or infinitely many solutions. In our case there are infinitely many solutions and we solve it by using the MATLAB function '`quadprog`' to find the optimal one which is given above.

3 Chemical reactor control

3.1 Introduction

In this part we apply MPC to this chemical reactor model, including a disturbance model to achieve off-set control of the controlled outputs, namely the concentration, c and the tank level, h. The controlled system has the following state space matrices given:

$$A = \begin{bmatrix} 0.2681 & -0.00338 & -0.00728 \\ 9.7032 & 0.3279 & -25.44 \\ 0 & 0 & 1 \end{bmatrix} B = \begin{bmatrix} -0.00537 & 0.1655 \\ 1.297 & 97.91 \\ 0 & -6.637 \end{bmatrix} C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The following are some preparations before:

3.1.1 Augmented model including disturbances

$$\begin{bmatrix} x \\ d \end{bmatrix}^+ = \underbrace{\begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix}}_{A_e} \begin{bmatrix} x \\ d \end{bmatrix} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{B_e} u$$

$$y = \underbrace{\begin{bmatrix} C & C_d \end{bmatrix}}_{C_e} \begin{bmatrix} x \\ d \end{bmatrix}$$

3.1.2 A designed Stated estimator for process and disturbance states

To be able to estimate states and disturbances the equation 16 is calculates for every iteration in the loop. To find the gain Le the function `kalman` was used in MATLAB. Input to the function is tuning parameters `Qk` and `Rk` representing the variance of the process noise and measurement noise, `NN` as the covariance of the different noises, `delayed` so the function found estimation using the time step before and `sysd = ss(Ae,Be_new,Ce,[],Ts);` where `Be_new` $= [Be,\ G]$ where $G = I_{n+nd}$ representing the process noise as n and nd is the number of states and disturbances. `[kest,Le,P] = kalman(sysd,Qk,Rk,NN,'delayed');`

$$\begin{bmatrix} \hat{x} \\ \hat{d} \end{bmatrix}^+ = \underbrace{\begin{bmatrix} A & B_d \\ 0 & I \end{bmatrix}}_{A_e} \begin{bmatrix} \hat{x} \\ \hat{d} \end{bmatrix} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{B_e} u + \underbrace{\begin{bmatrix} L_x \\ L_d \end{bmatrix}}_{L_e} \left(y_k - \underbrace{\begin{bmatrix} C & C_d \end{bmatrix}}_{C_e} \begin{bmatrix} \hat{x} \\ \hat{d} \end{bmatrix} \right) \tag{16}$$

3.1.3 The extended MPC algorithm with a steady state target calculation

$$\begin{bmatrix} I - A & -B \\ HC & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} B_d \hat{d} \\ z_{sp} - HC\hat{d} \end{bmatrix}$$

Where x_s and u_s is the state and input in steady state, and z_{sp} is the desired output setpoint which is set it to a zero vector in this assignment since off-set control is applied.

3.1.4 The extended MPC algorithm to handle different values for prediction and control horizon

The optimization problem for the MPC algorithm is:

$$\min_{x_s,u_s} ((Cx_{s,N} + C_d\hat{d}_N - y_{sp})^T P_f (Cx_{s,N} + C_d\hat{d}_N - y_{sp}) +$$
$$\sum_{i=0}^{M-1}(u_{s,i} - u_{sp})R_s(u_{s,i} - u_{sp}) + (u_{s,i} - u_{sp})\underbrace{(N-M-1)R_s}_{R_N}(u_{s,i} - u_{sp}) +$$
$$\sum_{i=0}^{N-1}(Cx_{s,i} + C_d\hat{d}_i - y_{sp})Q_s(Cx_{s,i} + C_d\hat{d}_i - y_{sp})) \quad (17)$$

s.t.

$$\begin{bmatrix} I-A & -B \\ HC & 0 \end{bmatrix}\begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} B_d\hat{d} \\ z_{sp} - HC\hat{d} \end{bmatrix}$$

Rewrite the optimization problem on form:

$$\min_{z} \quad \frac{1}{2}z^T H_M z + f^T z \quad (18)$$
$$\text{s.t.} \quad A_{eq}z = b_{eq} \quad (19)$$

With $z = \begin{bmatrix} \delta x^T(k+1) & \delta x^T(k+2) & \cdots & \delta x^T(k+N) & \delta u(k) & \cdots & \delta u(k+M) \end{bmatrix}^T$.
For this algorithm, the control horizon is $M = 3$, the prediction horizon is $N = 10$ and the final cost penalty is $P_f = Q_s$. The hessian matrix and equality constraints for the optimization problem are:

$$H_M = \begin{bmatrix} Q_s & 0 & & \cdots & & & 0 \\ 0 & \ddots & & & & & \\ & & Q_s & \ddots & & & \vdots \\ \vdots & & & P_f & & & \\ & & & & R_s & & \\ & & & & & \ddots & 0 \\ 0 & & & \cdots & & 0 & R_N \end{bmatrix}, \quad f = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$A_{eq} = \begin{bmatrix} -I & 0 & \cdots & & 0 & \cdots & 0 & B & 0 & 0 \\ A & -I & 0 & \cdots & & 0 & \cdots & 0 & B & 0 \\ 0 & A & -I & 0 & \cdots & & 0 & \cdots & 0 & B \\ \vdots & \ddots & \ddots & & & & & & & \vdots \\ & \ddots & & & & & \vdots & & & \\ 0 & & \cdots & & 0 & A & -I & 0 & 0 & B \end{bmatrix}, b_{eq} = \begin{bmatrix} -A \\ 0 \\ \\ \vdots \\ \\ 0 \end{bmatrix} \delta x$$

3.2 Tasks

We are free to choose how the integrating disturbance affects the states and measured outputs through the choice of B_d and C_d. The only restriction is that the augmented system is detectable. The interesting part here is to find out if we are going to successfully get an offset free controller with these modification, which is also the goals.
The target problem is assumed feasible. Augment the system model with a number of integrating disturbance equal to the number of measurements ($n_d = p$) (not the number of controlled variables) to garantee offset free control; choose any $B_d \in \mathbb{R}^{n \times p}$ and $C_d \in \mathbb{R}^{p \times p}$ such that:

$$rank \begin{bmatrix} I - A & -B_d \\ C & C_d \end{bmatrix} = n + p,$$

If the plant output y(k) goes to steady state y_s, the closed-loop system is stable, and constraints are not active at steady state, then there is zero offset in the controlled variables, that is $Hy_s = z_{sp}$.

3.2.1 Task d) Block diagram of the model predictive controller

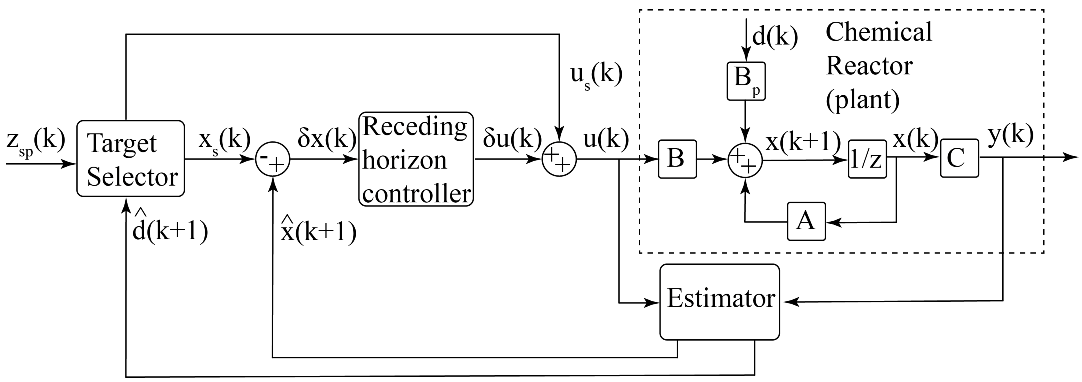

Figure 1: MPC controller consisting of: receding horizon controller, state estimator and target selector

3.2.2 Task e) Model 'a' in Matlab

The augmented system is detectable if and only if the nonaugmented system (A,C) is detectable, and the following condition holds:

$$rank \begin{bmatrix} I - A & -B_d \\ C & C_d \end{bmatrix} = n + n_d,$$

Where n is the dimension of the states and n_d is the number of integrating disturbance.

For model 'a' this rank condition holds, so the augmented system for 'a' is detectable. Now the integrating disturbances are added to the two controlled variables (c and h) by choosing

$$C_d = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, B_d = 0$$

The result is shown in the figure 2,4 and 4 below:

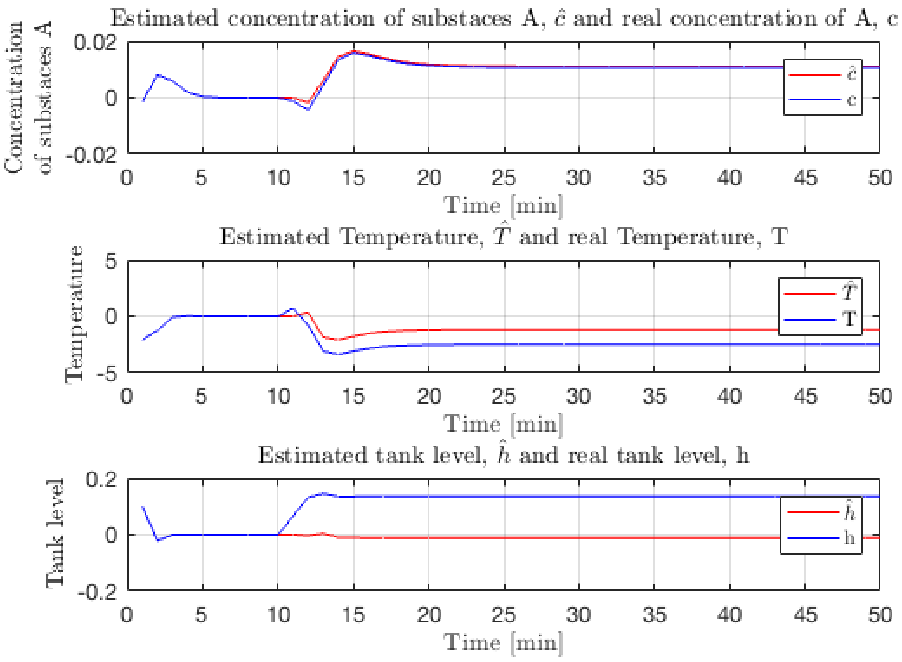

Figure 2: Estimated states and "real" states for model 'a'.

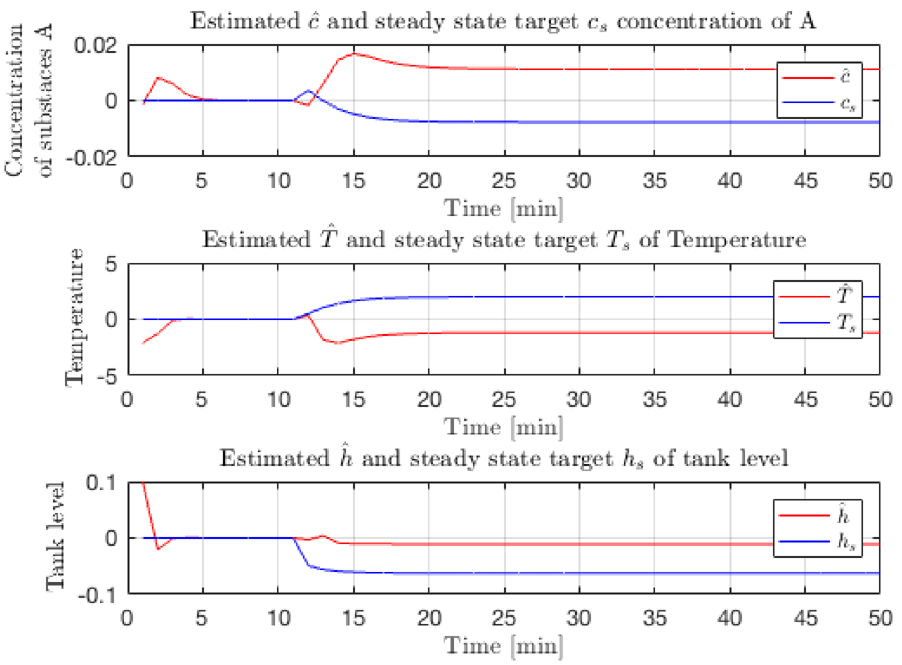

Figure 3: Estimated states and steady state targets for model 'a'.

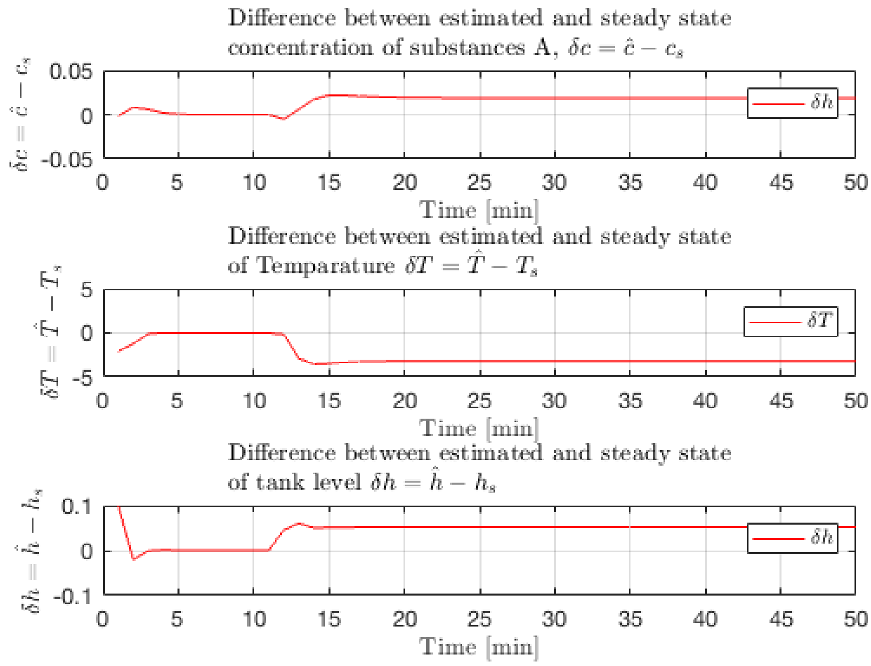

Figure 4: Offset for outputs both controlled and uncontrolled $\delta y = \hat{y} - y_s$, model 'a'.

Here notice that by adding integrating disturbances to the two controlled variables, c and h, both of these controlled variables as well as the third T, all still display nonzero offset form steady state. For the two controlled ones the offset is approximately 10% compared to the uncontrolled temperature.

3.2.3 Task f) Model 'b' in Matlab

A third disturbance is added to the second output giving:

$$C_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, B_d = 0$$

The augmented system is not detectable with this disturbance model. The rank of $\begin{bmatrix} I - A & -B_d \\ C & C_d \end{bmatrix}$ is only 5 instead of 6. The problem here is that the system level is itself an integrator, and we can not distinguish h from the integrating disturbance added to h. Still the algorithm is generating results which is visualized in figure 5,7 and 7.

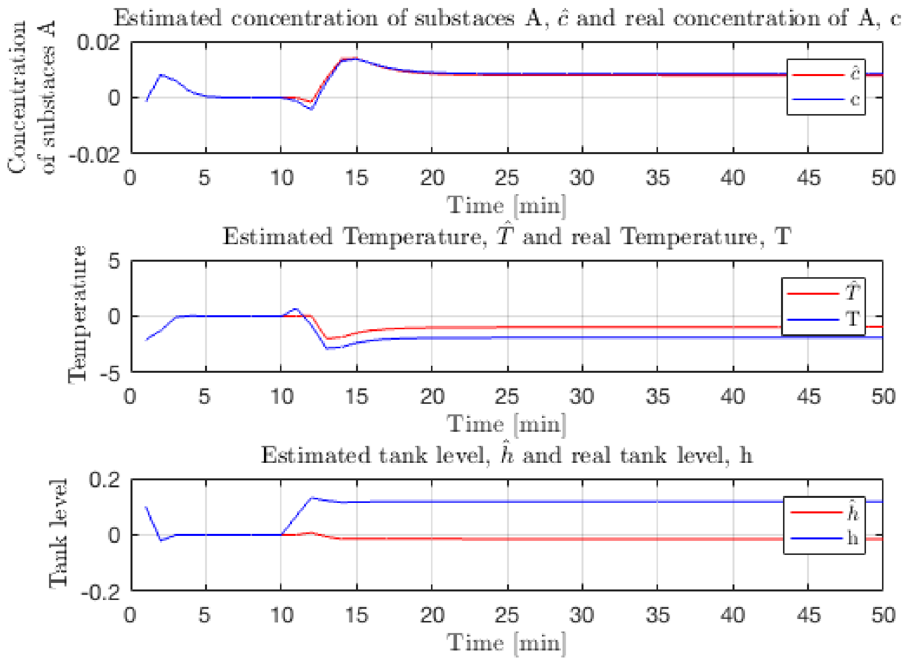

Figure 5: Estimated states and "real" states for model 'b'.

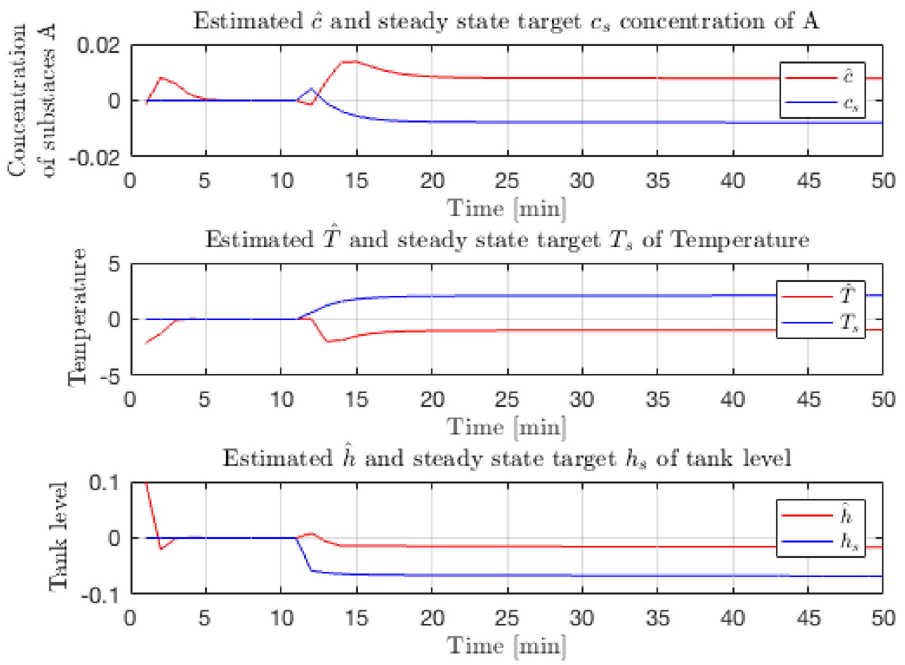

Figure 6: Estimated states and steady state targets for model 'b'.

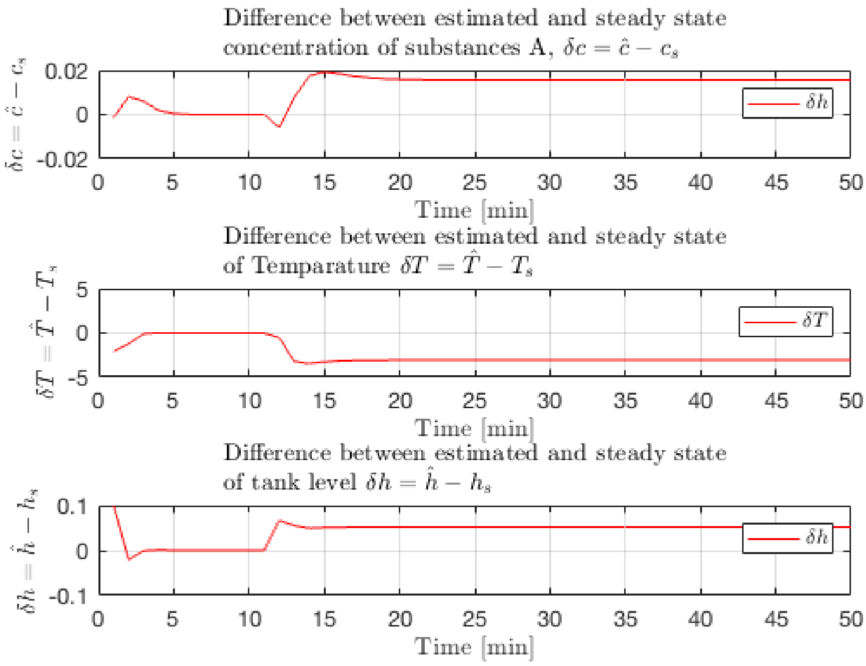

Figure 7: Offset for outputs both controlled and uncontrolled $\delta y = \hat{y} - y^s$, model 'b'.

Here it can be seen that the result is still not offset free and almost identical to the result for task e). To get a stable gain form the kalman function we were forced to increase the variance for the process noise a factor 10^3.

3.2.4 Task g) Model 'c' in Matlab

For this case we choose to model two integrating disturbances for the two controlled variables, c and h. One integrating third disturbance is added to the second of the manipulated inputs, the outlet flow rate.

$$C_d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, B_d = \begin{bmatrix} 0 & 0 & 0.1655 \\ 0 & 0 & 97.91 \\ 0 & 0 & -6.637 \end{bmatrix}$$

The augmented system is detectable for this disturbance model. The results for this are shown in figure 8,10 and 10. We see that we have zero offset in both the two controlled variables, c and h, and the second output T.

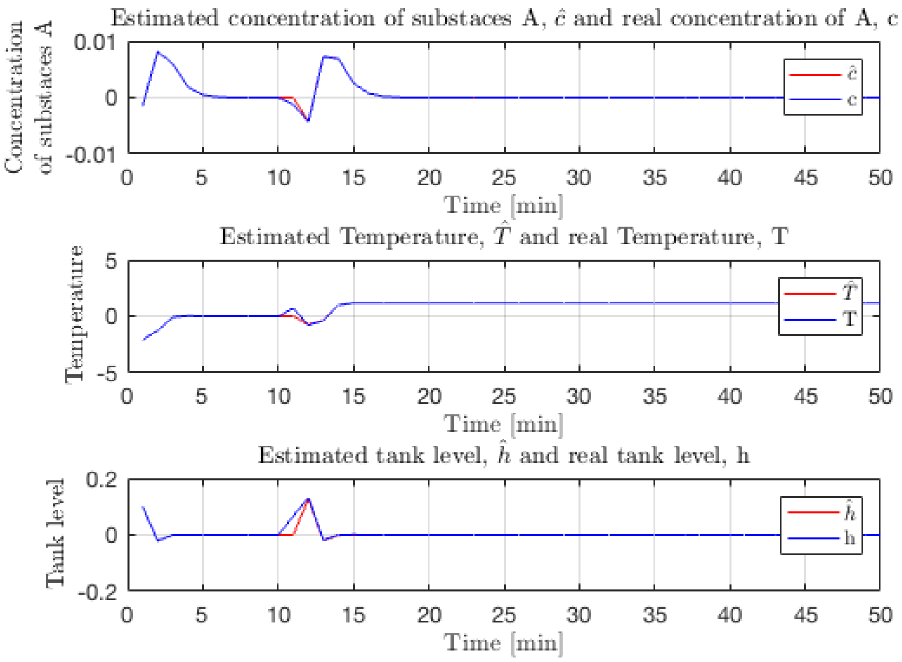

Figure 8: Estimated states and "real" states for model 'c'.

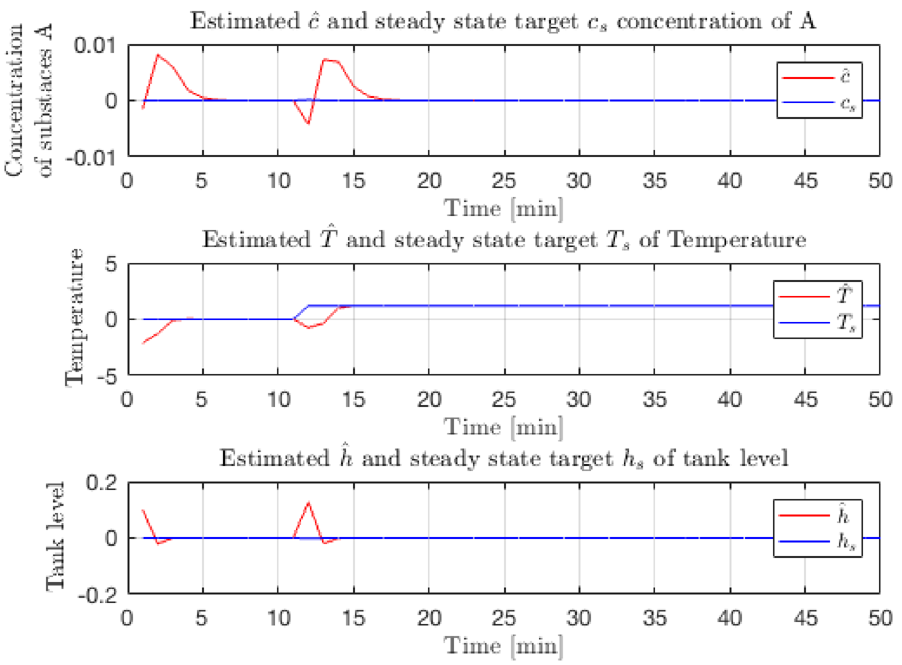

Figure 9: Estimated states and steady state targets for model 'c'.

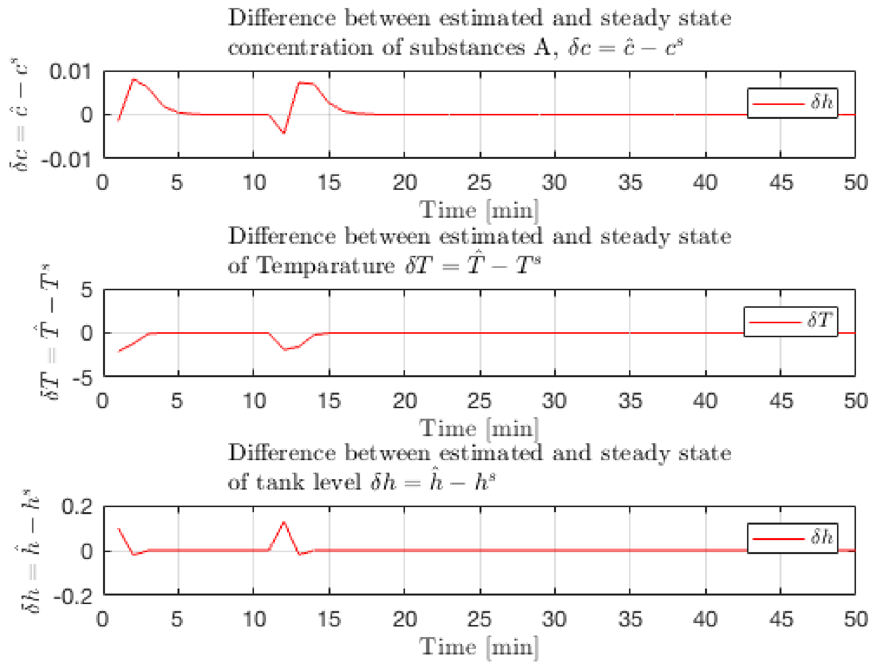

Figure 10: Offset for outputs both controlled and uncontrolled $\delta y = \hat{y} - y^s$, model 'c'.

Notice also that the dynamic behavior of all three outputs is superior to that achieved with the model using two integrating disturbances. The true disturbance, which is a step at the inlet flowrate, is better represented by including the integrator in the outlet flowrate. With a more accurate disturbance model, better overall control is achieved. The controller uses smaller manipulated variable action and also achieves better output variable behavior. An added bonus is that steady offset is removed in the maximum possible number of outputs.

4 Apendix Matlab code

4.1 Code for exercise a-c

```
clear all
close all
clc

A = diag([0.5 0.6 0.5 0.6]);
B = [diag([0.5 0.4]);diag([0.25 0.6])];
C = [1 1 0 0; 0 0 1 1];
zsp = [1  1]';

n = size(A,1); % n is the dimension of the state
m = size(B,2); % m is the dimension of the control signal
p = size(C,1); % p is the dimension of the measured output

H = eye(p);
% H = [1 0 0; 0 0 1]

Ta = [eye(n) A ,  B;
     H*C zeros(size(H,1),m)];

Tb = [zeros(n,1); zsp];

z_target = Ta\Tb;
z_target_linsolve = linsolve(Ta,Tb);
zs=C*z_target(1:n)

%% part b) p>m
clear all
close all
clc

A = diag([0.5 0.6 0.5 0.6]);
% B = [diag([0.5 0.4]);zeros(2,2)];
% B = [0.5 0; 0 0; 0.25 0; 0 0];
B = [0.5 0 0.25 0]';
C = [1 1 0 0; 0 0 1 1];
zsp = [1  1]';

n = size(A,1); % n is the dimension of the state
m = size(B,2); % m is the dimension of the control signal
p = size(C,1); % p is the dimension of the measured output

Qs = eye(p);
Q_bar = 2*C'*Qs*C;
R_bar = zeros(m,m);
Hm = blkdiag(Q_bar,R_bar);
f = [  2*zsp'*Qs*C, zeros(1,m)]';

H = eye(p);

Aeq = [eye(n) A ,  B];
rank(Aeq)

beq = [zeros(n,1)];

Ain = [];
Bin = [];
options = optimset('Algorithm','interior point convex','Display','off');
z = quadprog(Hm,f,Ain,Bin,Aeq,beq,[],[],[],options)
zs=C*z(1:n)

%% Part c p<m

clear all
close all
clc

A = diag([0.5 0.6 0.5 0.6]);
B = [diag([0.5 0.4]);diag([0.25 0.6])];
C = [1 1 0 0; 0 0 1 1];

zsp = [1]';
usp = [0  0]';

n = size(A,1); % n is the dimension of the state
m = size(B,2); % m is the dimension of the control signal
p = size(C,1); % p is the dimension of the measured output

H = [1 0];

Qs = zeros(n,n);
Q_bar = Qs;
R_bar = eye(m)*2;
Hm = blkdiag(Q_bar,R_bar);
f = [zeros(1,n), 2*usp'*R_bar]';

Aeq = [eye(n) A ,  B;
     H*C zeros(size(H,1),m)];

beq = [zeros(n,1);
              zsp];
```

```

Ain = [];
Bin = [];

options = optimset('Algorithm','interior point convex','Display','off');
z = quadprog(Hm,f,Ain,Bin,Aeq,beq,[],[],[],options);

zs=C*z(1:n)
```

4.2 Code for exercise e-g

```
%===========================================================================
% SSY280 Model Predictive Control 2012
%
% Homework Assignment 2:
% MPC Control of a Linearized MIMO Well Stirred Chemical Reactor
% Revised 2013 02 10
%===========================================================================

%****************** Initialization block ********************************

clear;
close all
tf=50;                    % number of simulation steps

%===========================================================================
% Process model
%===========================================================================

% h = 1; % sampling time in minutes
Ts = 60; % sampling time in seconds

% Definition of chemical reactor plant
A = [ 0.2681    0.00338    0.00728;
      9.7032    0.3279     25.44;
         0         0         1   ];
B = [  0.00537  0.1655;
       1.297    97.91 ;
       0         6.637];
C = [ 1 0 0;
      0 1 0;
      0 0 1];
Bp = [ 0.1175;
       69.74;
       6.637 ]; % Disturbances matrix

n = size(A,1); % n is the dimension of the state
m = size(B,2); % m is the dimension of the control signal
p = size(C,1); % p is the dimension of the measured output

d=0.01*[zeros(1*tf/5,1);ones(4*tf/5,1)]; % unmeasured disturbance trajectory

x0 = [0.01;1;0.1]; % initial condition of system's state


%===========================================================================
% Set up estimated model
%===========================================================================

% Three cases to be investigated

example = 'b';
switch example
    case 'a'
        nd = 2;
        Bd = zeros(n,nd);
        Cd = [1 0;0 0; 0 1];
        q=1; r=1;
    case 'b'
        nd=3;
        Bd = zeros(n,nd);
        Cd = [1 0 0;0 0 1;0 1 0];
        q=1000; r=1;
    case 'c'
        nd=3;
        Bd = [zeros(3,2) Bp];
        Cd = [1 0 0;0 0 0;0 1 0];
        q=1; r=1;
end



% Augment the model with constant disturbances

Ae = [A Bd; zeros(nd,n) eye(nd)]; %
Be = [B ; zeros(nd,m)]; %
% Be_2 = [zeros(n,nd);eye(nd)];
G = eye(n+nd);
Be_new = [Be, G];
Ce = [C Cd];

% Check augumented system stability
```

```
if rank([eye(n) A  Bd;C Cd]) == n+nd
    disp('augumented system is stable')
else
    disp('augumented system is not stable')
end
% Calculate estimated gain

sysd = ss(Ae,Be_new,Ce,[],Ts);
Qk = eye(n+nd)*q;   % Varience of process noise
Rk = eye(n)*r;   % Variance of measurement noise
NN = 0;          % Covariance of measurement noise and process noise
[kest,Le,P] = kalman(sysd,Qk,Rk,NN,'delayed');

% Check estimator stability poles <= 1
estimator_poles = eig(Ae Le*Ce)
abs(estimator_poles)

%==========================================================================
% Prepare for computing steady state targets
%==========================================================================

% Select 1st and 3rd outputs as controlled outputs
H = [1 0 0;0 0 1];

% Matrices for steady state target calculation to be used later

Ta = [eye(n) A ,  B;
    H*C zeros(size(H,1),m)];


% YOUR CODE GOES HERE

%==========================================================================
% Set up MPC controller
%==========================================================================

N=10;                   % prediction horizon
M=3;                    % control horizon

Q = diag([1 0.001 1]);  % state penalty
Pf = Q;                 % terminal state penalty
R = 0.01*eye(m);        % control penalty

    %======================================
    % Build Hessian Matrix
    %======================================

    %YOUR MODIFIED CODE FROM ASSIGNMENT 1 GOES HERE
Q_bar=kron(eye(N 1),Q);
R_bar=kron(eye(M 1),R);
% Hessian matrix
HM=(blkdiag(Q_bar,Pf,R_bar,R*(N M 1)));
    %==================================================
    % Equality Constraints
    %==================================================

Nn = kron(eye(N 1), A);
nn = zeros(n,n*N n);
nN = [nn;Nn];
NN = [nN,zeros(n*N,n)];
Mn = kron(eye(M),B);
MM = zeros(n*(N M),m*(M 1));
BM = repmat(B,N M,1);
mM = [Mn;MM,BM];

Aeq = [kron(eye(N),eye(n)) + NN,mM];
AA = [ A;zeros(n*N n,n)];

    % YOUR MODIFIED CODE FROM ASSIGNMENT 1 GOES HERE

    %==================================================
    % Inequality Constraints
    %==================================================

    Ain = [];
    Bin = [];

    %========================================================
    % Choose QP solver
    %========================================================

    solver = 'int';
    switch solver
        case 'int'
            options = optimset('Algorithm','interior point convex',...
                'Display','off');
        case 'set'
            options = optimset('Algorithm','active set','Display','off');
    end

%****************** End of initialization ******************************

%==========================================================================
% Simulation
%==========================================================================

% Initialization

```

```
% YOUR CODE GOES HERE   INITIALIZE ALL VARIABLES NEEDED
dhat = zeros(nd,1); % initialize estimated disturbaces
xdhat = [x0;dhat];  % initialize estimated state vector and disturbaces
xk = x0;            % initial state value for process
yk = C*xk;          % initial output value for process
zsp = zeros(m,1);   % Setpoint
uk = zsp;           % initial input value for process

% Construct matrixes to save values in
xdhat_save = zeros(n+nd,tf);
z_target_save = zeros(n+m,tf);
uk_save = zeros(m,tf);
yk_save = zeros(n,tf);
xk_save = zeros(n,tf);
delta_x_save = zeros(n,tf);
delta_u_save = zeros(m,tf);

% Simulate closed loop system

    for k = 1:tf

        %=====================================
        % Update the estimated state xhat(k|k 1)
        %=====================================

        xdhat = Ae*xdhat + Be*uk + Le*(yk Ce*xdhat);

        %===============================================
        % Update the process state x(k) and output y(k)
        %===============================================

        xk = A*xk + B*uk + Bp*d(k);
        yk = C*xk;

        %=========================================
        % Calculate steady state targets xs and us
        %=========================================

        dhat = xdhat(n+1:end);

        Tb = [Bd*dhat;
            zsp H*Cd*dhat];

        z_target = Ta\Tb;

        %============================================
        % Solve the QP (for the deviation variables!)
        %============================================
        xhat = xdhat(1:n);
        delta_x = xhat z_target(1:n);
        beq =AA*delta_x;
        % UPDATE RHS OF EQUALITY CONSTRAINT HERE

        % NOTE THAT HM IS USED FOR THE HESSIAN, NOT TO BE CONFUSED
        %   WITH H THAT IS USED FOR SELECTING CONTROLLED VARIABLES
        z = quadprog(HM,[],Ain,Bin,Aeq,beq,[],[],[],options);

        % CALCULATE THE NEW CONTROL SIGNAL HERE
         delta_u = z(n*N+1:n*N+m);
        uk = delta_u+z_target(n+1:n+m);
        % NOTE THAT YOU NEED TO GO FROM DEVIATION VARIABLES TO 'REAL' ONES!

        %==============================
        % Store current variables in log
        %==============================
        xdhat_save(:,k) = xdhat;
        z_target_save(:,k) = z_target;
        uk_save(:,k) = uk;
        yk_save(:,k) = yk;
        xk_save(:,k) = xk;
        delta_x_save(:,k) = delta_x;
        delta_u_save(:,k) = delta_u;

   end % simulation loop
%=======================================================================
% Plot results
%=======================================================================
textsize = 13;
t = (1:1:tf);

figure;
subplot(3,1,1);
plot(t,xdhat_save(1,:),'r',t,xk_save(1,:),'b'); % concentration subst A
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{c}$','c'},'Interpreter','latex')
title({'Estimated concentration of substaces A, $\hat{c}$ and real concentration of A, c'}...
    ,'Interpreter','latex')
ylabel({'Concentration','of substaces A'},'Interpreter','latex')
xlabel('Time [min]','Interpreter','latex')

subplot(3,1,2);
plot(t,xdhat_save(2,:),'r',t,xk_save(2,:),'b'); % Temperature
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{T}$','T'},'Interpreter','latex')
title({'Estimated Temperature, $\hat{T}$ and real Temperature, T'},...
    'Interpreter','latex')
```

```
ylabel({'Temperature'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')

subplot(3,1,3);
plot(t,xdhat_save(3,:),'r',t,xk_save(3,:),'b'); % Tank level
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{h}$','h'},'Interpreter','latex')
title({'Estimated tank level, $\hat{h}$ and real tank level, h'},...
    'Interpreter','latex')
ylabel({'Tank level'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')


figure;
subplot(3,1,1);
plot(t,delta_x_save(1,:),'r'); % concentration subst A
grid on;
set(gca,'fontsize',textsize)
legend({'$\delta h$'},'Interpreter','latex');
title({'Difference between estimated and steady state',...
    'concentration of substances A, $\delta c = \hat{c} c_s$'},...
    'Interpreter','latex')
ylabel({'$\delta c = \hat{c} c_s$'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')


subplot(3,1,2);
plot(t,delta_x_save(2,:),'r'); % Temperature
grid on;
set(gca,'fontsize',textsize)
legend({'$\delta T$'},'Interpreter','latex');
title({'Difference between estimated and steady state',...
    'of Temparature $\delta T = \hat{T} T_s$'},'Interpreter','latex')
ylabel({'$\delta T = \hat{T} T_s$'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')

subplot(3,1,3);
plot(t,delta_x_save(3,:),'r'); % Tank level
grid on;
set(gca,'fontsize',textsize)
legend({'$\delta h$'},'Interpreter','latex');
title({'Difference between estimated and steady state',...
    'of tank level $\delta h = \hat{h} h_s$'},'Interpreter','latex')
ylabel({'$\delta h = \hat{h} h_s$'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')

figure;
subplot(3,1,1);
plot(t,xdhat_save(1,:),'r',t,z_target_save(1,:),'b'); % concentration subst A
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{c}$','$c_s$'},'Interpreter','latex')
title({'Estimated $\hat{c}$ and steady state target $c_s$ concentration of A'},...
    'Interpreter','latex')
ylabel({'Concentration','of substaces A'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')

subplot(3,1,2);
plot(t,xdhat_save(2,:),'r',t,z_target_save(2,:),'b'); % Temperature
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{T}$','$T_s$'},'Interpreter','latex')
title({'Estimated $\hat{T}$ and steady state target $T_s$ of Temperature'},...
    'Interpreter','latex')
ylabel({'Temperature'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')

subplot(3,1,3);
plot(t,xdhat_save(3,:),'r',t,z_target_save(3,:),'b'); % Tank level
grid on;
set(gca,'fontsize',textsize)
legend({'$\hat{h}$','$h_s$'},'Interpreter','latex')
title({'Estimated $\hat{h}$ and steady state target $h_s$ of tank level'},...
    'Interpreter','latex')
ylabel({'Tank level'},'Interpreter','latex')% not given anny units o this
xlabel('Time [min]','Interpreter','latex')
```