Assignment 6 - Project Kårstø Statpipe Butane Splitter

TTK4210 - Advanced Control of Industrial Processes

Sondre Myrberg

April 19, 2018

Contents

List of Figures

1 Introduction

In this report we will look into a model of the Kårstø Statpipe Butan Splitter, and tune nine different controllers to make this distillation column perform within the specification, namely that there are maximum 4% of n-butane in the top product and maximum 2.5% iso-butane in the bottom product. This will be achieved by LV-contol which is control based on reflux flow (L) from reflux drum into the top column and vapour flowrate (V) in the bottom of the column to control the composition in the respective column. A simplified model of the plant is shown in Figure 1.1. Here we see that there are two main loops, one for each of the controlled variables L and V to control the compositions. A more detailed model is showed in Figure 1.2, where all controllers we want to tune are marked with red text, as these controller names will be referred to often in this report. Firstly we will start by tuning the secondary controllers in the loops, before moving on to the higher level controllers, before we finish by simulating the whole system in different scenarios.

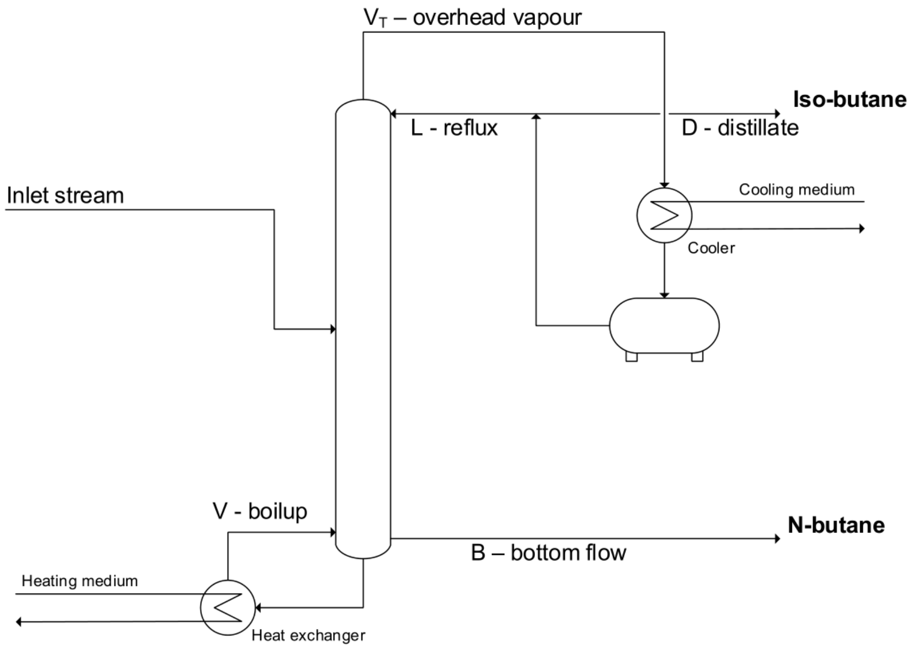

Figure 1.1: Simplified model of the plant shown in Figure 1.2

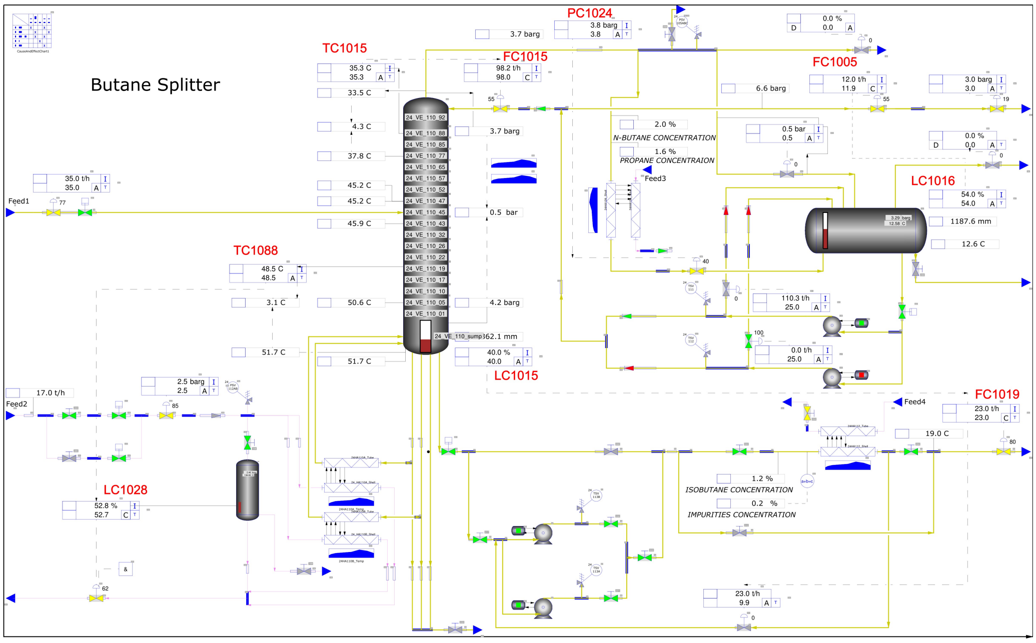

Figure 1.2: Full system with controllers numbered with red text

2 Identification and tuning of the controllers

In this chapter we will look at the tuning of the different controllers of the system. Firstly we will look at the secondary controllers, before we later move on to the higher level controlllers. A summary of the different controller parameters can be found in Table 2.1.

Table 2.1: Summary of controller parameters in the plant

Controller	K_p	T_i	G
24_FC1005	1.18125	5	1.72
24_FC1015	0.2	5	1.28
24_FC1019	0.27225	5	0.41
24_LC1015	9.66	1200	−3.22
24_LC1016	3.3	750	−0.89
24_LC1028	47.9	128	−1.13
24_PC1024	17.5	2400	−1.62
24_TC1015	66	1440	−0.44
24_TC1088	5.02	1440	−0.49

2.1 Tuning of secondary controllers

Firstly we want to tune the secondary controllers, which do not depend much on which control structure is used for composition control. The controllers this applies to is the flow controllers 24_FC1005, 24_FC1015 and 24_FC1019, the level controller 24_LC1028, and also the pressure controller 24_PC1024. These controllers all have fast dynamics compared to the composition control and primary level controllers.

2.1.a Tuning of 24_FC1019

Firstly we tune the controller 24_FC1019, which is a flow controller controlling the bottom flow B of N-butane. When trying to apply Skogestad IMC-tuning to a first order model, the results were poor as there are close to none time delay, as expected in a valve, which lead to this being a unprecise model. Ziegler-Nichols method, as described in [1], was then a natural choice, and with a critical gain of $K_{p_k} = 0.605$ and integral time $T_i \to \infty$ we got the osciallations shown in Figure 2.1.1. By zooming into this we read the period of the oscillations as $T_k = 6\,\mathrm{s}$. This yields the gain and integral time

$K_p = 0.27225$ and $T_i = 5\,\mathrm{s}$. With this tuning we get the response shown in Figure 2.1.2. This is a good response to a relatively large step reference change. Before the reference step the controller has an external set point, that is why the set point is changing dynamically. When doing the step change, the controller settings was changed from external to internal set point, and then back to external for the second step.

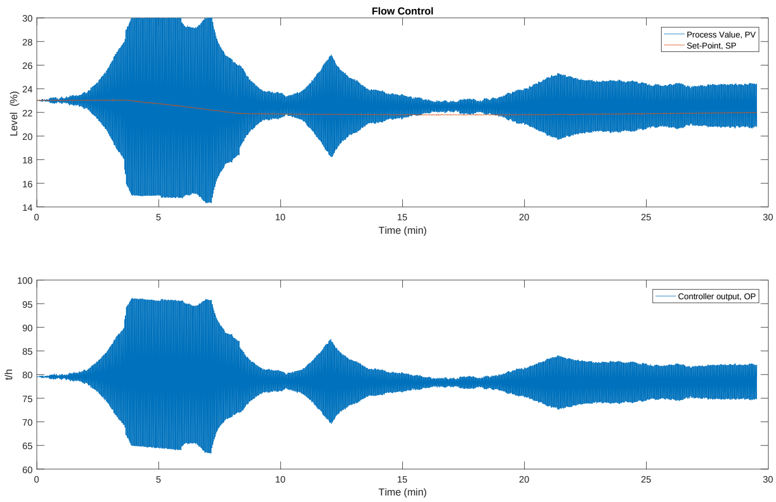

Figure 2.1.1: Oscillations of the flow controller 24_FC1019 with $K_{p_k} = 0.605$ from about 22 min

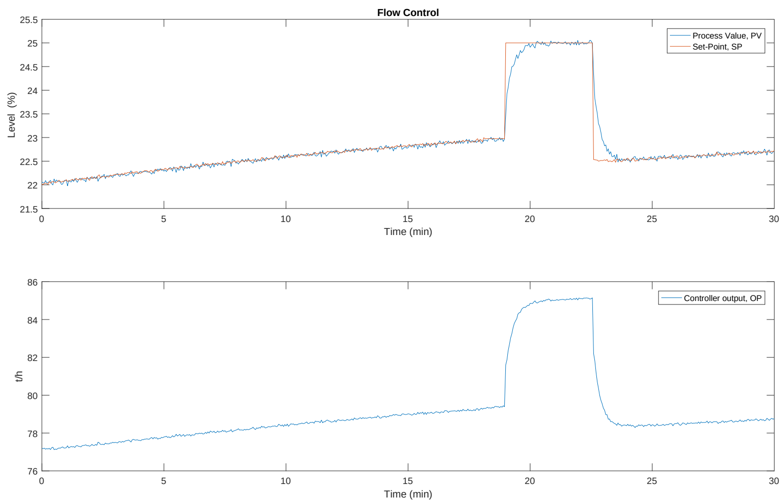

Figure 2.1.2: Response of 24_FC1019 tuned using Ziegler-Nichols method

2.1.b Tuning of 24_FC1005

We now tune the controller 24_FC1005, another flow controller, this one controlling D, the distillate flow of Iso-butane. Using the same method as in the tuning of 24_FC1019, we find the critical values $K_{p_k} = 2.625$ and $T_k = 6$s. This yields the parameters $K_p = 1.18125$ and $T_i = 5$s. As we see in Figure 2.1.4, the we get a slow oscillation when switching back to external reference, but this is due to oscillations in the set point, and not the process value. We see that the process nicely follows the reference with high accuracy.

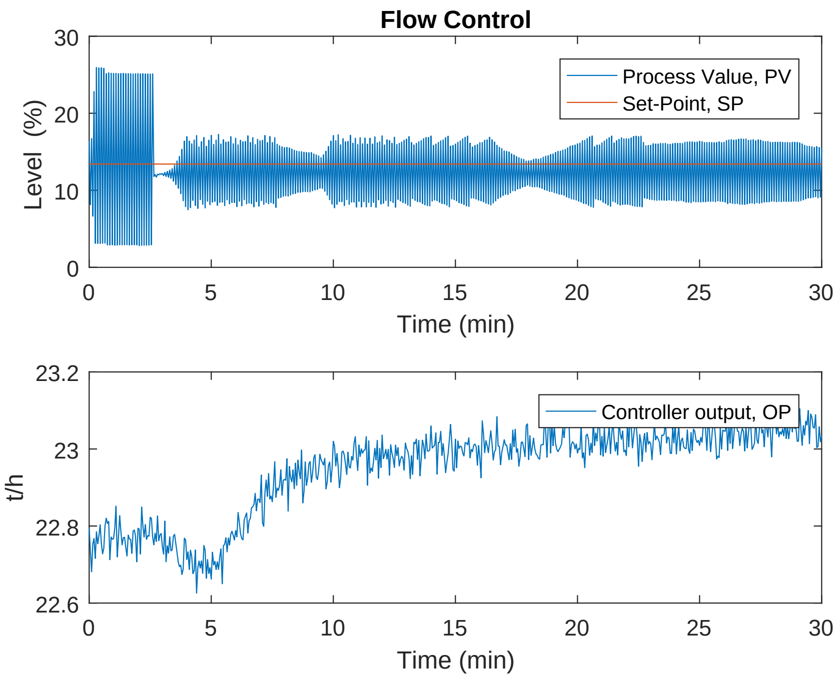

Figure 2.1.3: Oscillations of the flow controller 24_FC1005 with critical gain $K_{p_k} = 2.625$ from about 23 min

2.1.c Tuning of 24_LC1028

Further we tune the level controller 24_LC1028 wich controls the area for heat exhange in the bottom heat exchanger. This makes it indirectly control the bottom vapour flowrate V. Here we use the SIMC-method for tuning of PI(D)-controllers. Using this method we now need to take into account the internal scaling in the controller. By setting the integral time $T_i \to \infty$ we get a simple P-controller which we apply a step in the reference. By using the SIMC-tuning rules from [2] applied on a integrating step response we get

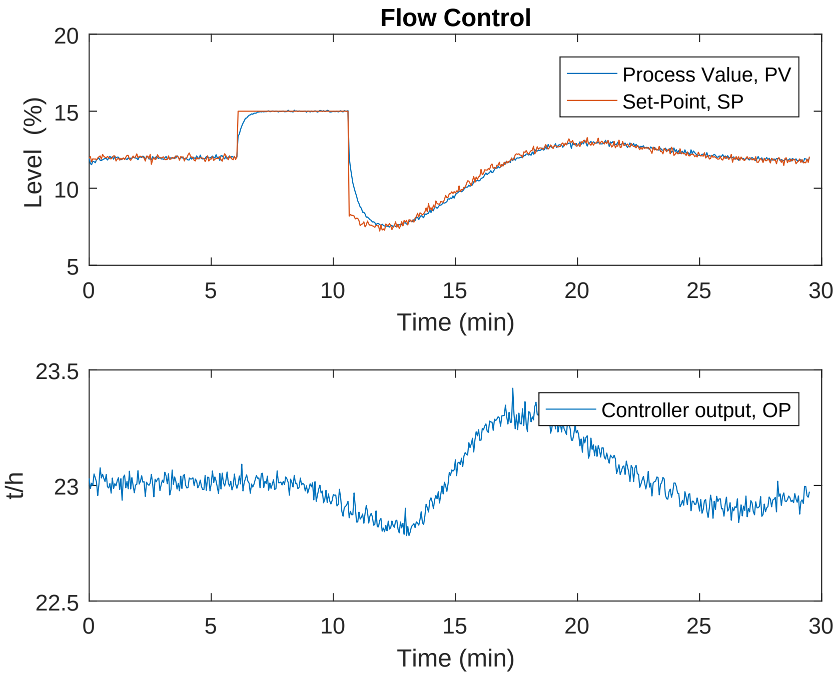

Figure 2.1.4: Response of `FC_1005` tuned using Ziegler-Nichols method

$$
\begin{aligned}
k' &= \frac{\Delta y}{\Delta t \Delta u} \\
K_c &= \frac{1}{k'} \frac{1}{\tau_c + \theta} \\
T_i &= 4(\tau_c + \theta)
\end{aligned}
\tag{2.1.1}
$$

where τ_c is chosen as 3θ to achieve robustness. Smaller τ_c improves preformance, but makes the system less robust, as described in [2]. By measuring on Figure 2.1.5 we get the observations

$$
\begin{aligned}
\Delta u &= 1 \\
\Delta y &= -0.3413 \\
\Delta t &= 462 \\
\theta &= 8
\end{aligned}
\tag{2.1.2}
$$

which again gives the results

$$
\begin{aligned}
K_c &= -42.3 \\
T_i &= 128
\end{aligned}
\tag{2.1.3}
$$

The scaling factor can be calculated as shown in (2.1.6), and when applying a step input with $\Delta y_{\text{ref}} = 2$, we get the immediate response $\Delta u_{\text{meas}} = -17.68$. $\Delta u_{\text{exp}} = K_p \Delta y_{\text{ref}} =$

20, which gives the scaling factor

$$G = \frac{u_{\text{exp}}}{u_{\text{meas}}} = -1.13 \tag{2.1.4}$$

which again gives the controller parameters

$$\begin{aligned} K_{p,\text{applied}} &= GK_c = 47.9 \\ T_i &= 128 \end{aligned} \tag{2.1.5}$$

This again leads to fairly good response, although the controller is maybe slightly aggressive. This is shown in Figure 2.1.6. The equations for the scaling parameter are given as

$$\begin{aligned} \Delta u_{\text{exp}} &= K_p \Delta y_{\text{ref}} \\ G &= \frac{\Delta u_{\text{exp}}}{\Delta u_{\text{meas}}} \\ \implies K_{p,\text{applied}} &= GK_p \end{aligned} \tag{2.1.6}$$

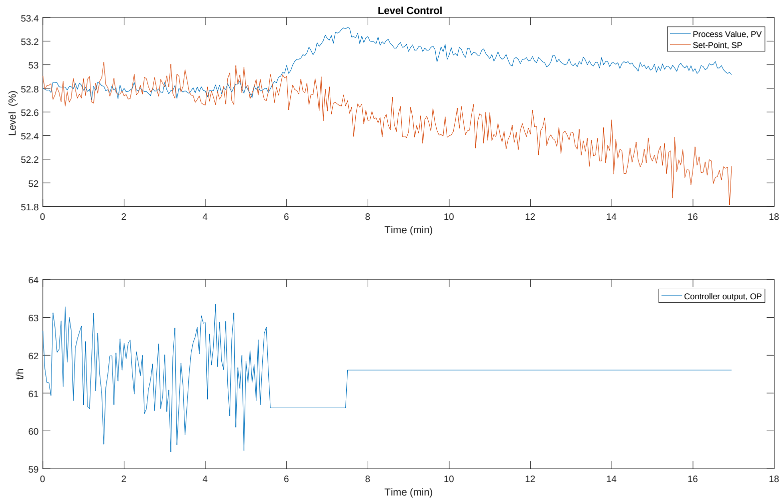

Figure 2.1.5: Output of `24_LC1028` with simple P-controller and applied step at $t \approx 7\,\text{min}$

2.1.d Tuning of `24_PC1024`

When applying a step oon the input of the pressure controller `PC_1024` we get the output response shown in Figure 2.1.7. When then applying the SIMC method to this output response, we see that we do not have a proper first order model with delay or a clean integrating response with delay, we instead treat the time from the step until the

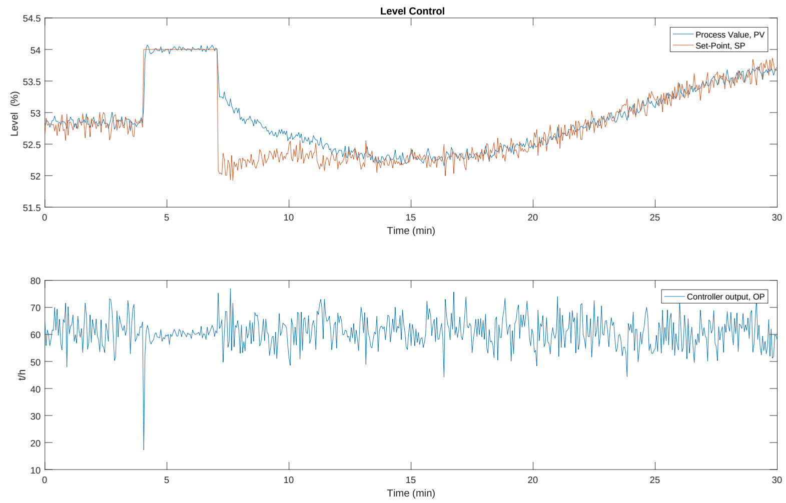

Figure 2.1.6: Output of 24_LC1028 tuned using SIMC and PID-scaling

integration starts properly as the time delay θ. We also calculate the internal gain as in (2.1.6) and obtain the parameters $G = -1.62$, $\delta u = 0.3$, $\delta t = 900$, $\delta y = -0.021$ and $\theta = 300$. Using the rules from (2.1.1) and chosing $\tau_c = \theta$ for faster control we get the controller parameters

$$\begin{aligned} K_p &= GK_c = 17.5 \\ T_i &= 4(\tau_c + \theta) = 2400 \end{aligned} \tag{2.1.7}$$

This integral time seems quite large, but when looking at the response of the PC_1024 after tuning the parameters, shown in Figure 2.1.8, we see that the controller response is fairly good, and keeps the pressure well within acceptable deviations.

2.1.e Tuning of 24_FC1015

When it came to the controller 24_FC1015, I came to the conclusion that all responses were unstable when trying to use the classic tuning methods. I therefore went back to the even more classic method of trial and error. By tweaking K_p and T_i within reasonable limits we hit a set of parameters, $K_p = 0.2$ and $T_i = 5$, which gave fairly good response. This is shown in Figure 2.1.9. We see that the process value follows the reference pretty good.

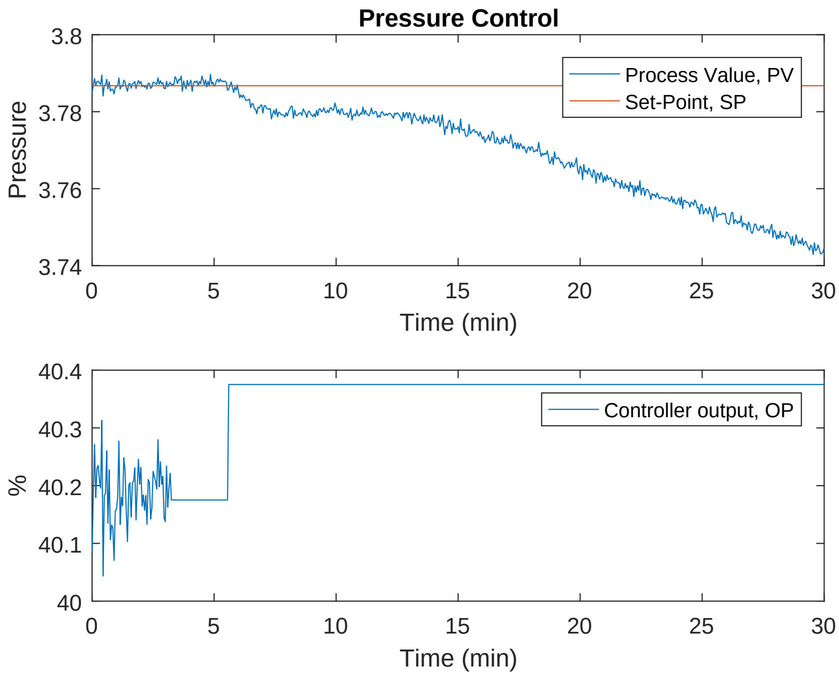

Figure 2.1.7: Response of the controller `PC_1024` when applying a step on input

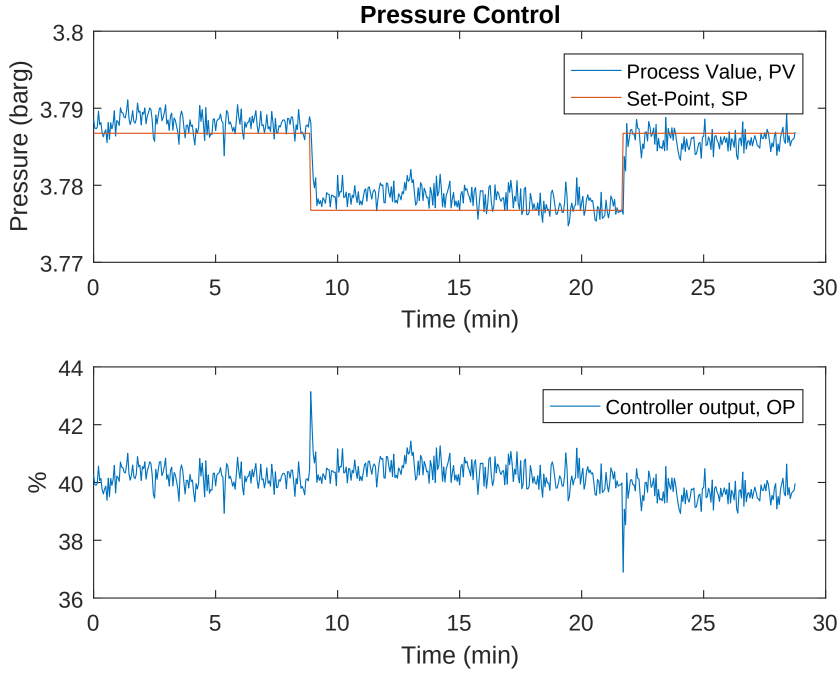

Figure 2.1.8: Response of the pressure controller PC_1024 after tuning and applying a step in the reference value

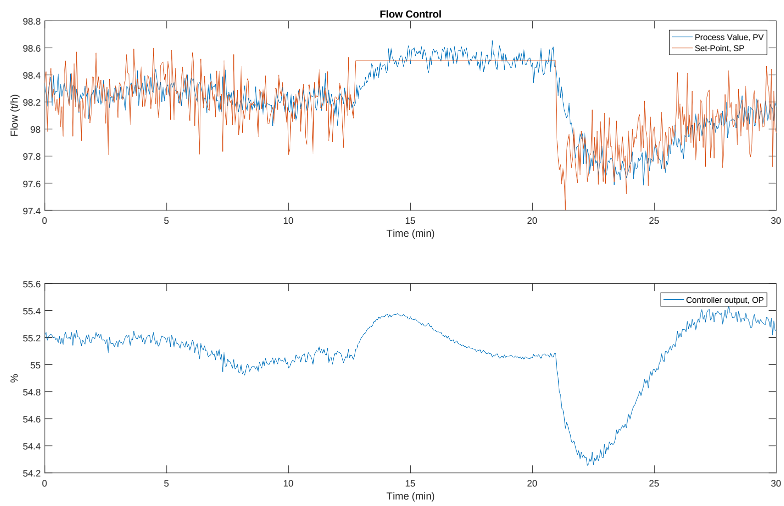

Figure 2.1.9: Response of the controller 24_FC1015 after tuning using trial and error

2.2 Tuning and identification of level controllers

We now move on to the identification and tuning of the level controllers `24_LC1015` and `24_LC1016`. This will be done by changing the setpoints of the controllers such that the manipulated variables are excited. The base code for this is shown in Listing A.1, where minor adjustments is done to the .txt-file we import based on what data we have exported from K-Spice and which contoller we have excited.

2.2.a Level controller 24_LC1016

We start by doing an set point exciting experiment on the controller `24_LC1016`, where we do several step changes in the set point, and then observe the output. We then import this to MATLAB and use the built in function `n4sid` to estimate a state space realization of the system we are looking at. The general code to do this is shown in Listing A.1 in Appendix A. In Figure 2.2.1 we see the actual response of the controller normalized and compared to the identified model. We see that the identified model is fairly good.

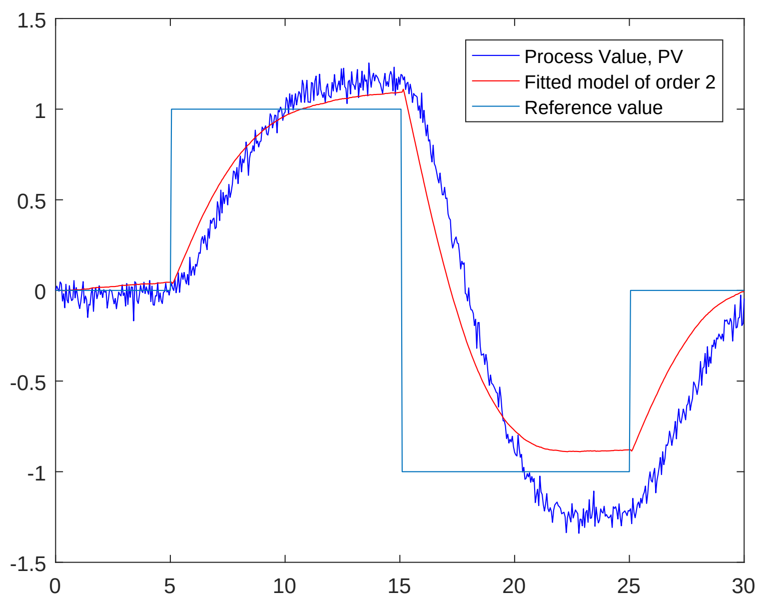

Figure 2.2.1: Actual response and response of identified model of order 2 for the controller `24_LC1016`

Using this model we get the second order discrete time system system

$$
\begin{aligned}
x_{n+1} &= \underbrace{\begin{bmatrix} 1.001 & 0.02635 \\ -0.009939 & 0.1251 \end{bmatrix}}_{A} x_n + \underbrace{\begin{bmatrix} -7.587 \cdot 10^{-5} \\ -0.006211 \end{bmatrix}}_{B} u_n \\
y_n &= \underbrace{\begin{bmatrix} 19.36 & -0.6743 \end{bmatrix}}_{C} x_n + \underbrace{0}_{D} u
\end{aligned}
\tag{2.2.1}
$$

which yields the step response shown in Figure 2.2.2, where we see that we have a somewhat underdamped system.

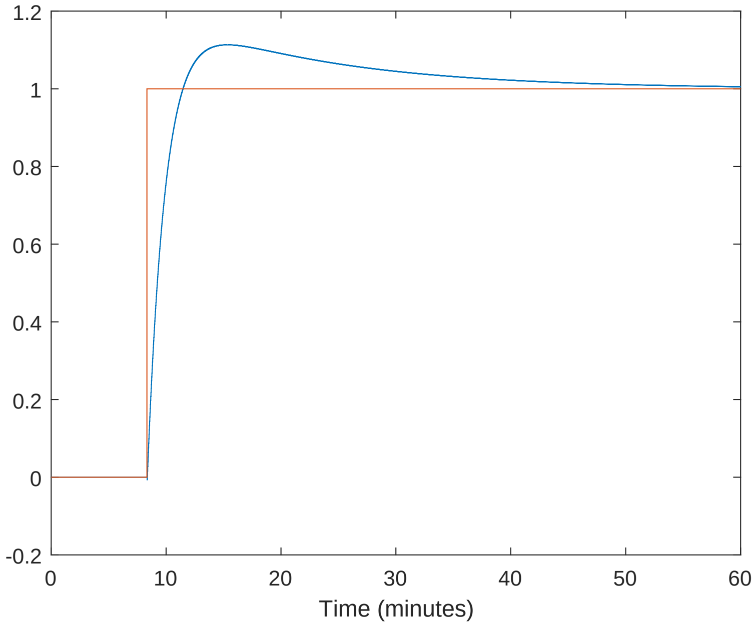

Figure 2.2.2: Step response of the controller `24_LC1016` for identified model before tuning

However, tuning the parameters of the controller does not give severly increased response. Anyway, with $T_i = 750\,\mathrm{s}$ and $K_p = -4$ in MATLAB we get the slightly faster response and faster approach to the stationary value. Using gain scaling as shown in Equation 2.1.6 we get the scaling and gain parameters

$$
\begin{aligned}
G &= \frac{u_{\text{exp}}}{u_{\text{meas}}} = \frac{4}{-4.83} = -0.83 \\
K_p &= G K_{p,\text{MATLAB}} = -0.83 \cdot (-4) = 3.32 \\
T_i &= 750
\end{aligned}
\tag{2.2.2}
$$

Applying these parameters and running the MCL-script given in Listing C.1 in Appendix C we get the response shown in Figure 2.2.3. Here we see that the response is pretty fast and accurate, but we also see anomalies in the start and end of the plot. This is due to the fact that the main composition controllers are put in manual mode after five minutes of the simulation, and back again to automatic mode after 115 minutes, so that the tuning of this level controller is to be more precise. When returning to automatic mode, the temperature controller `24_TC1015` looks to decrease the reflux flow into the chamber which again leads to less flow out of the drum where we try to control the level. Because of this what we care about here is the response between five and 115 minutes, which we see is pretty good.

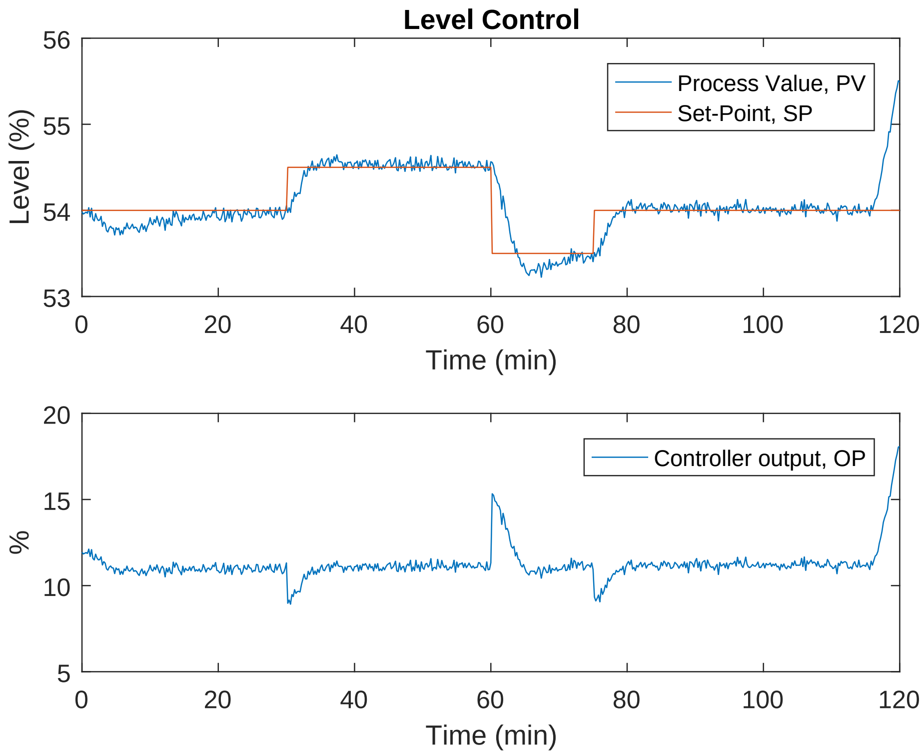

Figure 2.2.3: Tuned response of the controller `24_LC1016` when applying several step reference changes

2.2.b Level controller 24_LC1015

Now we move on to the level controller `24_LC1015` which controls the liquid level of the distillation column. By performing a step change in the reference and simulating for five hours we get the response shown in Figure 2.2.4. Here we apply a step change of 1.5% after 90 minutes and then simulate for another 210 minutes.
Further we fit a model using the same procedure as in subsection 2.2.a and obtain the model

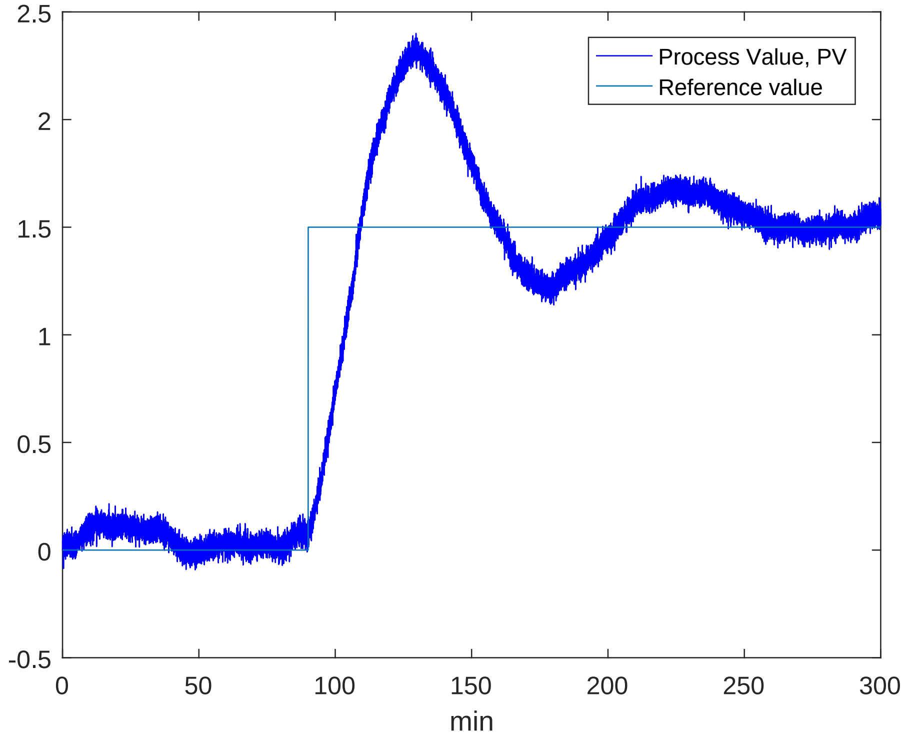

Figure 2.2.4: Response of the controller `24_LC1015` with step change after 90 minutes

$$
\begin{aligned}
x_{n+1} &= x_n - 1.155 \cdot 10^{-5} u_n \\
y_n &= 179.03 x_n
\end{aligned} \tag{2.2.3}
$$

This model fits very good with the obtained data, as shown in Figure 2.2.5.
This fitted model is now used for tuning the controller. With the initial parameters $K_p = 1$ and $T_i = 500$ as we used when identifying the system, gives the response shown in Figure 2.2.6. Here we see an oscillating system which is not desirable.
Finding the internal scaling gain and tuning by som quick trial and error we find a good set of parameters given by

$$
\begin{aligned}
G &= \frac{u_{\text{exp}}}{u_{\text{meas}}} = \frac{1}{-0.31} = -3.22 \\
K_p &= G K_{p,\text{MATLAB}} = -3.22 \cdot (-3) = 9.66 \\
T_i &= 1200
\end{aligned} \tag{2.2.4}
$$

These parameters applied to the K-Spice model gives the performance shown in Figure 2.2.7. Here we have just a small overshoot and fast approach to steady state value, which is really good compared to before tuning as shown in Figure 2.2.4.

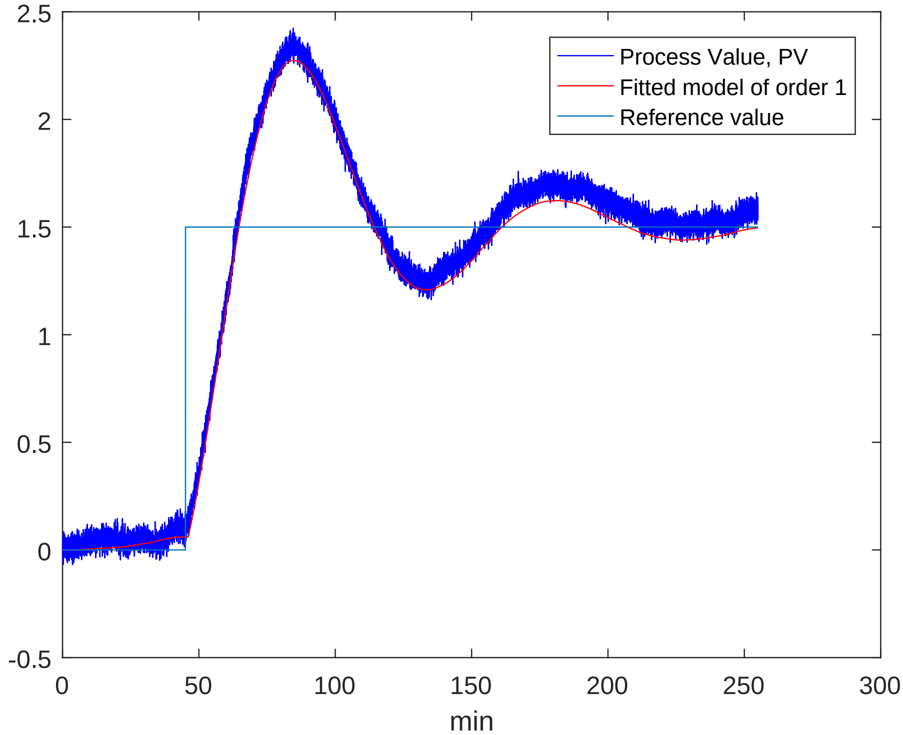

Figure 2.2.5: Fitted model and actual data for `24_LC1015`

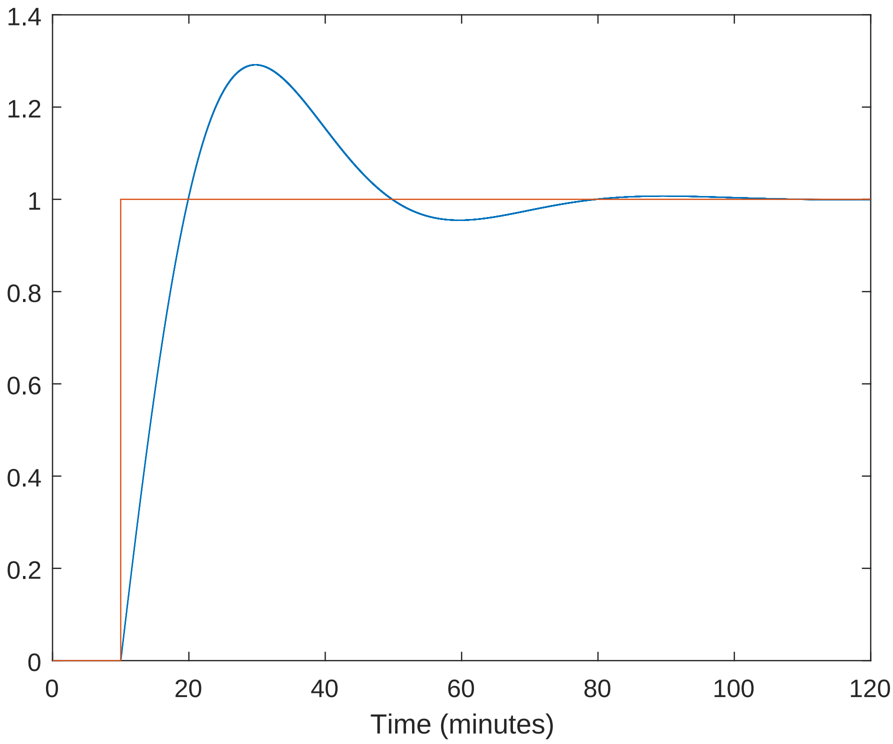

Figure 2.2.6: Response of the model for the controller `24_LC1015` with $K_p = 1$ and $T_i = 500$ before tuning

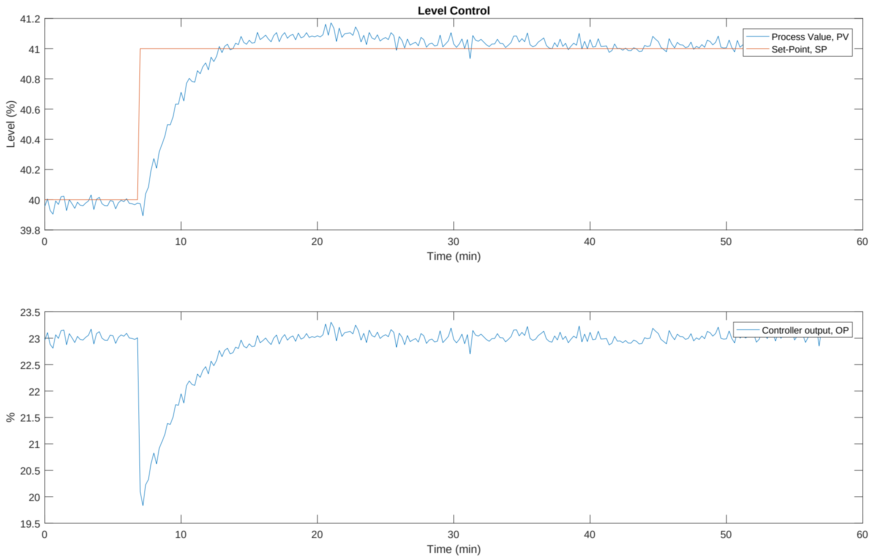

Figure 2.2.7: Response of the controller 24_LC1015 after tuning

2.3 Identification and tuning of the composition controllers

We now move on to two final controllers, namely the top level composition controllers. These controllers are temperature controllers that while operating makes sure the temperature in two trays in the distillation column is within acceptable regions such that the composition of what's inside the trays stays within their impurty-limits. Here we have `24_TC1015` controlling tray 88 to a set temperature of 35.30 °C while `24_TC1088` controls the temperature in tray 19 to a set temperature of 48.51 °C. The manipulated variables for the composition controllers can be excited directly, and that is what we aim to do.

2.3.a Identification and tuning of temperature controller `24_TC1088`

As mentioned this controller's task is to control the temperature in tray 19 such that there are maximum 2.5% iso-butane in the column. This is achieved by keeping the temperature in the tray at 48.51 °C which corresponds to 1.25% iso-butane. To identify this system we use the script Listing C.2 in Appendix C doing several step changes in the controller output. As this is an integrating process we then get the process values shown in Figure 2.3.1. Here we have also fitted a model of second order using the MATLAB script shown in Listing A.1, with slight tweaks so that we use the right controller of course. This results in a good fit, which we see in Figure 2.3.1. The modeled second order system can be described by the state space realization

$$\begin{aligned} x_{n+1} &= \underbrace{\begin{bmatrix} 0.9996 & 0.0292 \\ -0.0006 & -0.1474 \end{bmatrix}}_{A} x_n + \underbrace{\begin{bmatrix} 0.00002 \\ -0.0014 \end{bmatrix}}_{B} u_n \\ y_n &= \underbrace{\begin{bmatrix} 16.1739 & -0.9862 \end{bmatrix}}_{C} x_n + \underbrace{0}_{D} u \end{aligned} \tag{2.3.1}$$

Tuning this controller we use SIMC-tuning as given in Equation 2.1.1 and from Figure 2.3.2 we read off the parameters

$$\begin{aligned} y(\infty) &= -0.6696 \\ \tau &= 2476 \\ \theta &= 60 \end{aligned} \tag{2.3.2}$$

which gives

$$\begin{aligned} k &= \frac{y(\infty)}{\Delta u} = -0.6696 \\ k' &= \frac{k}{\tau} = -2.704 \cdot 10^{-4} \end{aligned} \tag{2.3.3}$$

Then using the rules for SIMC-tuning we get the set of parameters shown in Table 2.2. Choosing $\tau_c = 5\theta$ we get the response shown in Figure 2.3.3, which is good. Before applying these parameters, we as earlier need to compensate for internal gain in the controller. Doing a step change to reference we get the values of internal gain and

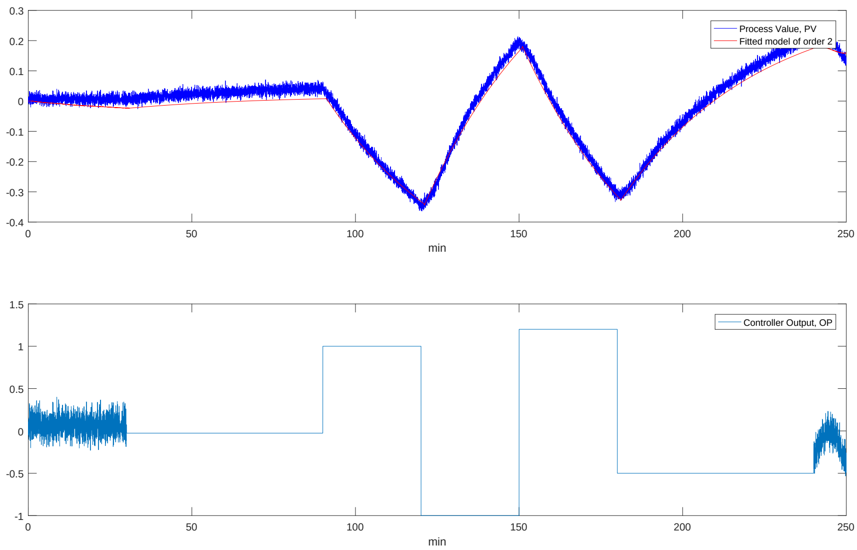

Figure 2.3.1: Process value, fitted model and controller output for `24_TC1088` when running the experiment

Table 2.2: Set of parameters for different choices of τ_c

τ_c	K_c	T_i
θ	-30.8	480
3θ	-15.4	960
5θ	-10.3	1440

applied gain shown in Equation 2.3.4, before we apply these parameters to the K-Spice simulation, and get the response shown in Figure 2.3.4. Here we see that the controller hits the reference value with good accuracy without overshooting, which when taking the purity constraints into consideration is a good thing. The response also reaches the setpoint fairly fast.

$$\begin{aligned} G &= \frac{-2.5}{57.97 - 52.84} = -0.487 \\ \implies K_{p,\text{applied}} &= -10.3 \cdot (-0.487) = 5.02 \end{aligned} \tag{2.3.4}$$

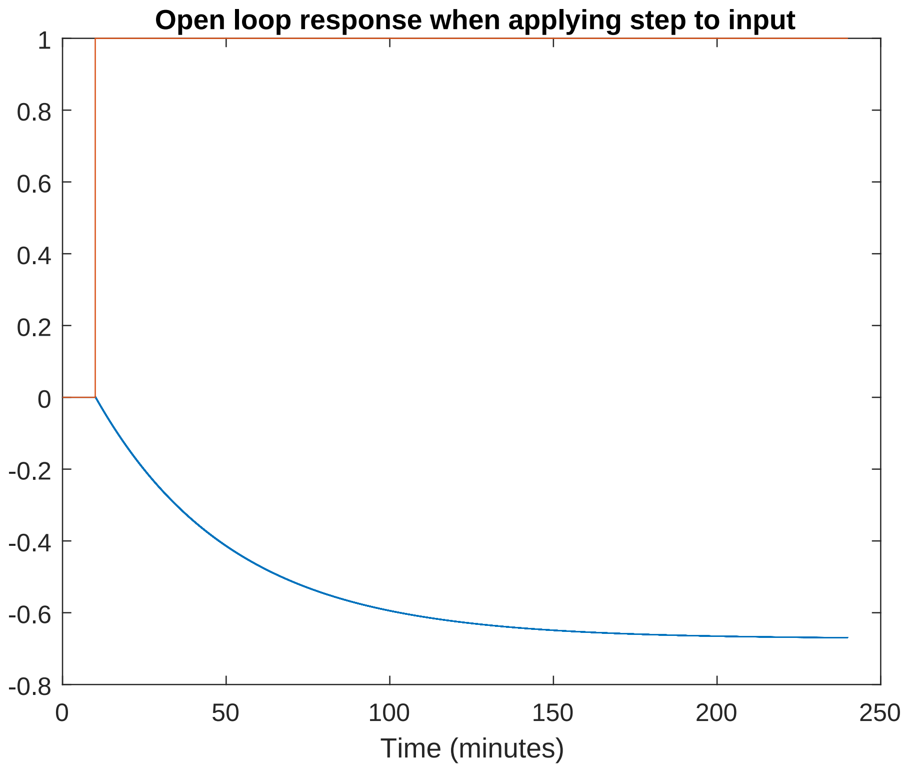

Figure 2.3.2: Open loop response of the fitted model when applying unit step to input

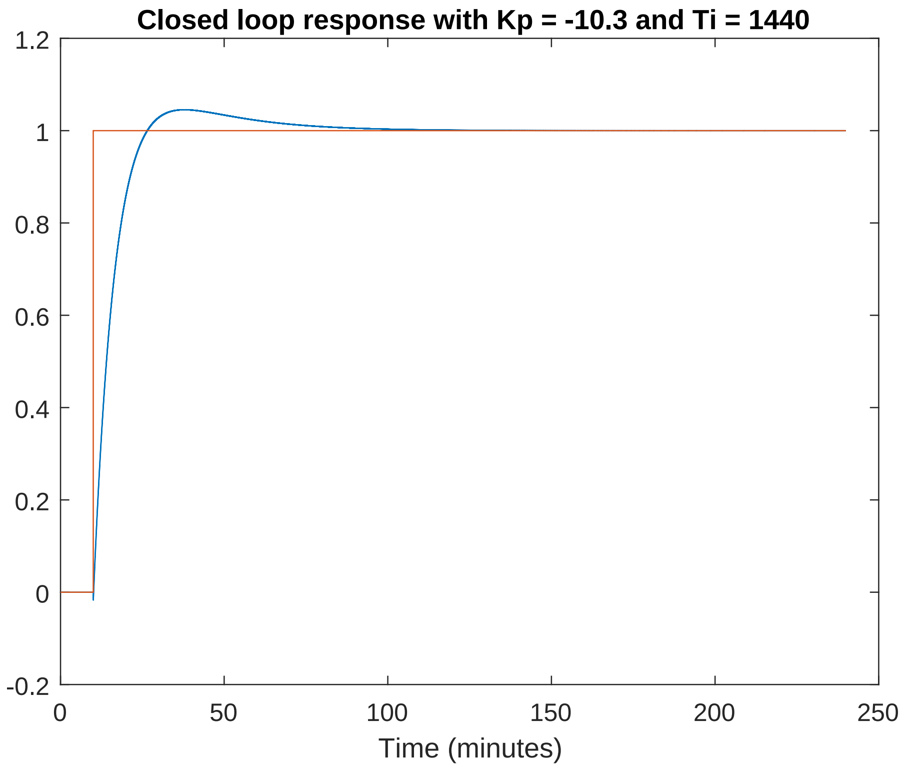

Figure 2.3.3: Tuned response of the model with $K_p = -10.3$ and $T_i = 1440$

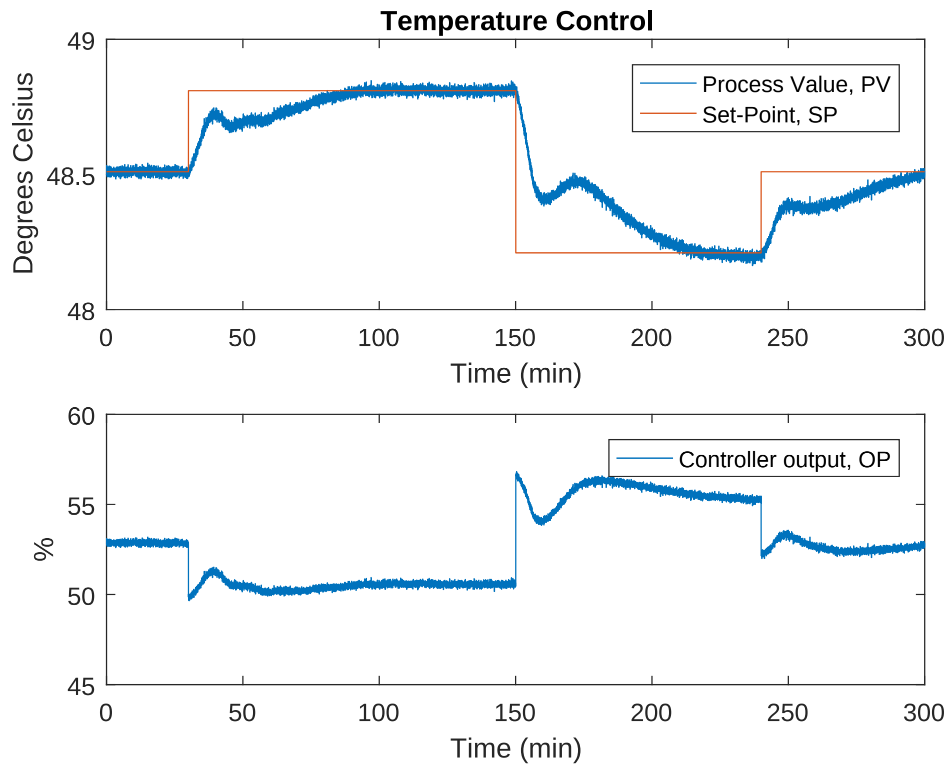

Figure 2.3.4: Response of K-Spice simulation of `24_TC1088` when applying steps to reference

2.3.b Identification and tuning of temperature controller `24_TC1015`

We now move on to the final controller we are going to tune for now, namely the composition controller for the amount of n-butane in the top product. The maximum allowed impurity is specified to 4%, which is achieved at 35.85 °C in tray 88. To stay within this constraint we aim at controlling the temperature to the set point of 35.3 °C which results in 2% n-butane in the top product. Doing a similar experiment as for the previous controller we get the response shwen in Figure 2.3.5 and the identified second order state space model given by

$$
\begin{aligned}
A &= \begin{bmatrix} 1 & 0.0255 \\ 0.0008 & -0.0659 \end{bmatrix} \\
B &= \begin{bmatrix} -0.0072 \\ 0.1938 \end{bmatrix} \cdot 10^{-3} \\
C &= \begin{bmatrix} 19.2362 & -0.9464 \end{bmatrix} \\
D &= 0
\end{aligned}
\tag{2.3.5}
$$

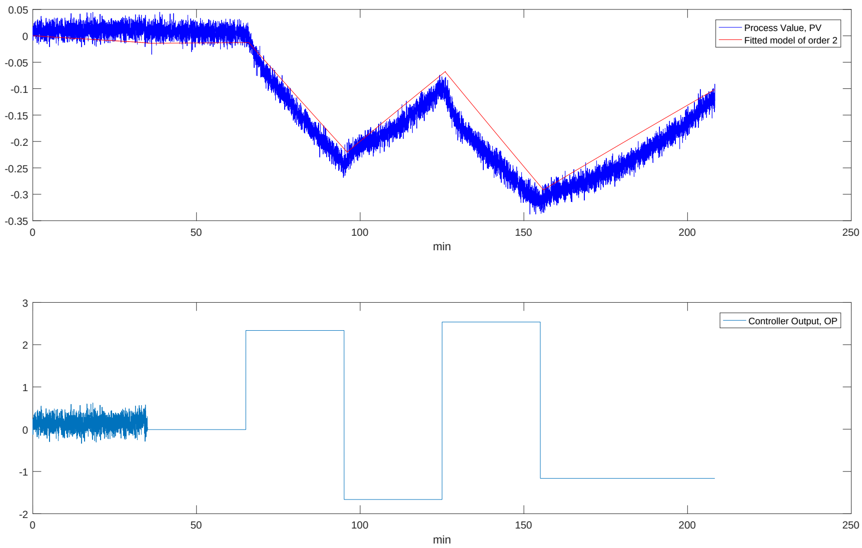

Figure 2.3.5: Response of `24_TC1015` after several steps in controller output together with identified model of second order

Observing the open loop step response to this system we get the response shown in Figure 2.3.6, which we see is an integrating response. Applying the SIMC-tuning method as in Equation 2.1.1 to this output we get values

$$k' = \frac{\Delta y}{\Delta t \Delta u} = \frac{-0.2137 - (-0.0201)}{(5000 - 4000) \cdot 1} \\ = -4.84 \cdot 10^{-5} \tag{2.3.6}$$

which with different choices of τ_c for $\theta = 60\,\text{s}$ gives the controller parameters table shown in Table 2.3.

Table 2.3: Set of parameters for different choices of τ_c

τ_c	K_c	T_i
θ	-452	480
3θ	-226	960
5θ	-150	1440
7θ	-113	1920

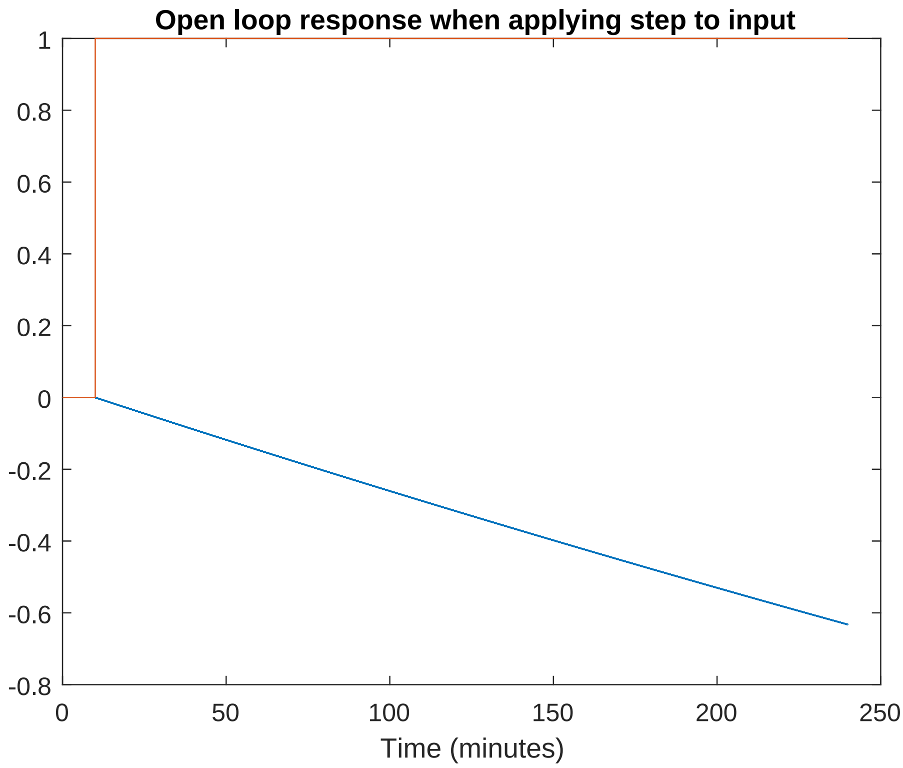

Figure 2.3.6: Open loop step response of `24_TC1015`

We here get the best result when choosing $\tau_c = 5\theta$ which gives the step response shown in Figure 2.3.7. From here we need to find the internal gain factor, which by doing a step reference change as always are found to be $G = -0.44$. This means that $K_{p,\text{applied}} = 66$. The response is shown in Figure 2.3.8. Here we see fast approach to an increase in the temperature setpoint, but somewhat slower when we apply a decrease in reference temperature. All in al this system behaves reasonably good with this controller tuning.

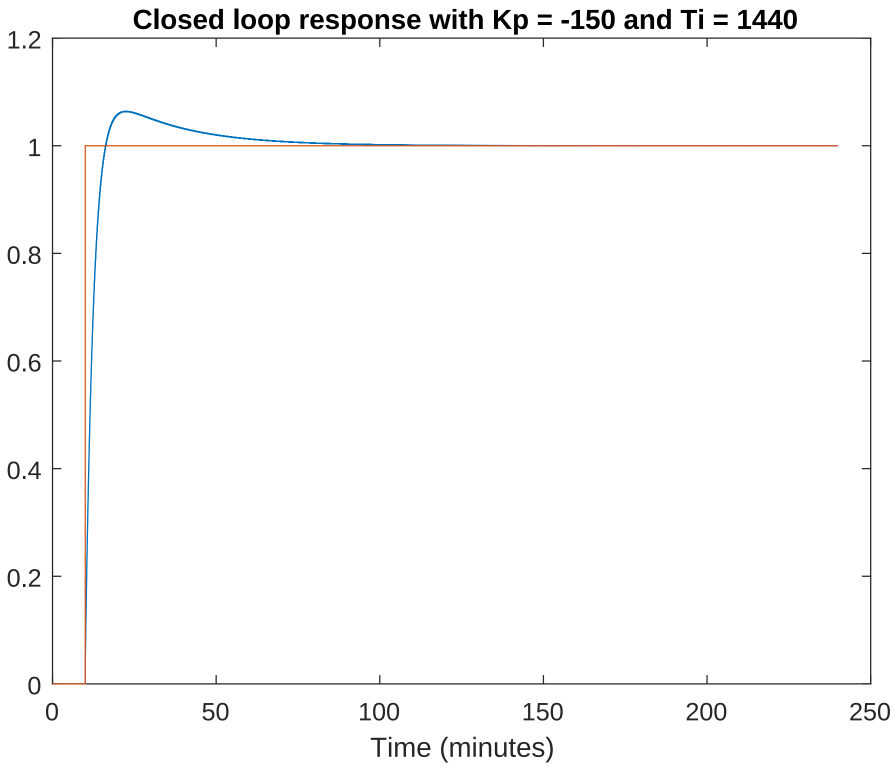

Figure 2.3.7: Stop response of the tuned model of `24_TC1015`

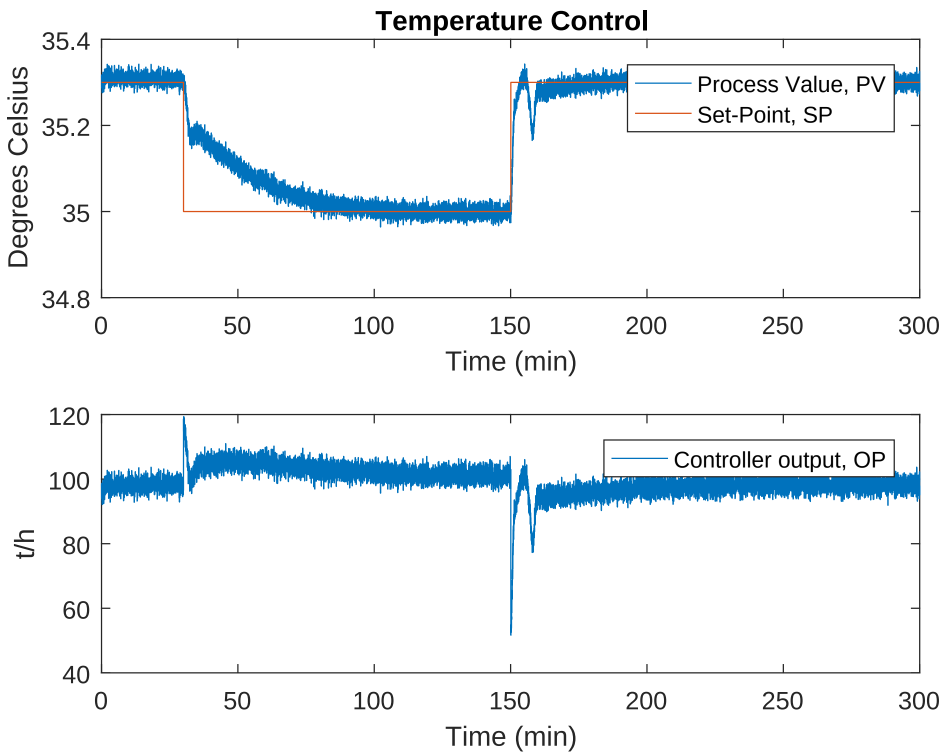

Figure 2.3.8: Response of the controller 24_TC1015 when doing step changes in reference temperature

3 Results

In this chapter we will look at the performance of the fully tuned system in different simulation scenarios. Firstly we will look at how the temperature performs when starting at a stable state, then we move on to scenarios where we do different changes in the setpoint of the controller before we finish by looking at how the system behaves when doing a cold start.

3.1 Stable startup

When loading the conditions where the initial temperatures are about 0.3 °C below deisred set point we get the response shown in Figure 3.1.1. Here we see that both the temperatures settles towards desired temperature within reasonable time, without any significant oscillations, and continues to work around this setpoint without large deviations due to noise. The controllers are also operating in an area where they are far from saturating, which is desirable.

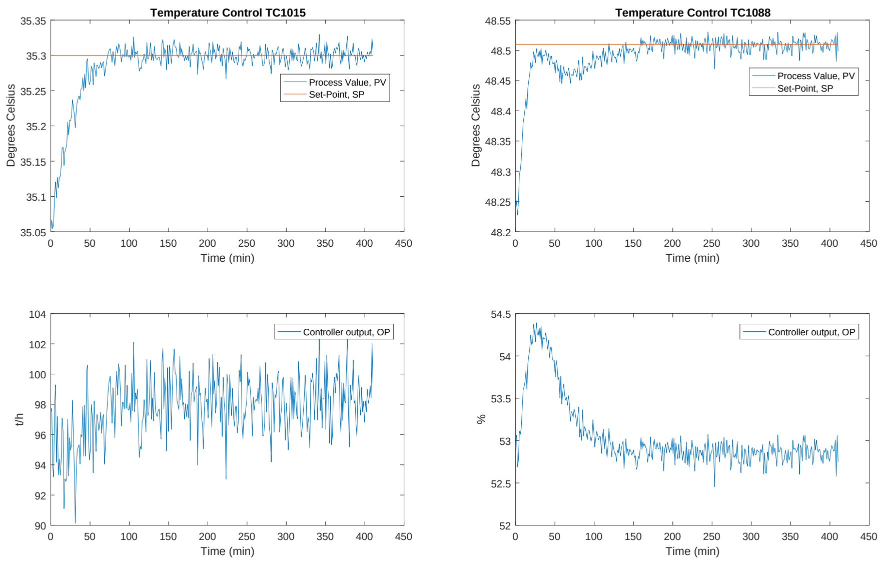

Figure 3.1.1: Response of both temperature controllers with initial conditions not at set point

3.2 Cold start

When doing a cold startup, the process values are far off the desired checkpoints. This applies to both the temperature and level controllers. To reach the desired checkpoint we therefore need a lot of time. At a 15 hour simulation we see that we reach desired temperature in the top column fairly fast, while the response in the bottom column is slower. This is due to physical limitations in the system, as we see that the controller output from `24_TC1088` are saturated at minimum. This output is the reference value for the controller `24_LC1028` which controls the heat exhange area for the boilup part. As this is saturated as shown in Figure 3.2.2 we see that there are no more room for more heat exchange, and this is a limitation for the system. It will take som time for the full plant to be in running condition after a cold start, as we need to wait for the temperature in the bottom column to reach its set point.

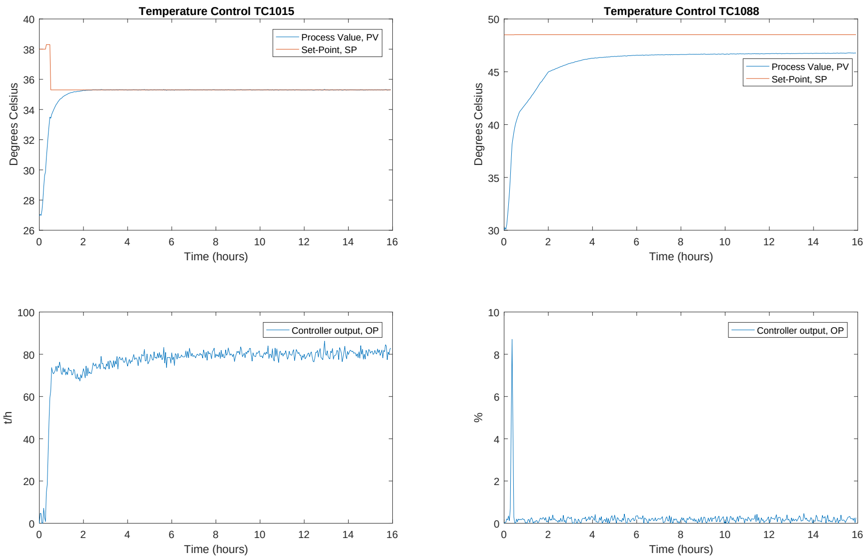

Figure 3.2.1: Output of both temperature controllers when doing a cold start of the system

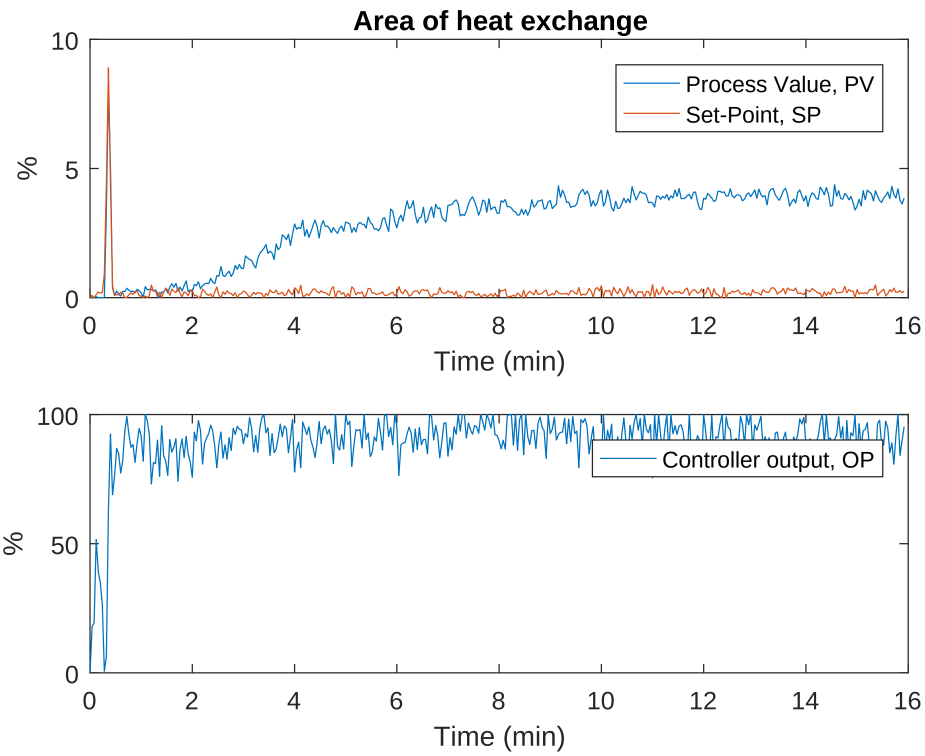

Figure 3.2.2: Controller for the heat exhange in the boilup part of the bottom column in saturation

3.3 Rapid step changes

We now move on to a 12 hour simulation doing four step changes of $\pm 0.3\,^\circ$C, two on each controller, we get the output shown in Figure 3.3.1. Here wee see that both controllers follow reference good, although we see that when doing a change in the top column, this affects the bottom column quite alot more than the opposite direction. This is probably due to the fact that K_p for `24_TC1015` is severely larger than for `24_TC1088` which again leads to more agressive tuning, which also can be seen in Figure 3.3.1. Although this interaction we observe that both process values stays well within the specifications and there are no noticeable oscillations for this simulation. When looking at the time frame of this simulation, it's also fair to mention that we do many changes within a short time period and that the response with this set of controller parameters performs pretty decent.

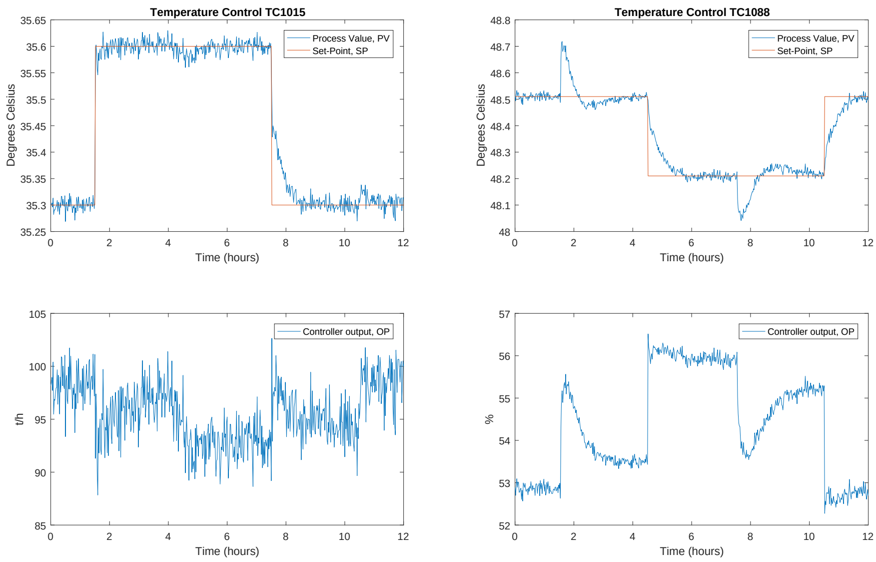

Figure 3.3.1: Output of both temperature controllers when doing several steps within short time

3.4 Summary

All in all the plant now performs reasonably within the specifications and the controllers perform well by them selves. The controller parameters for the two temperature controllers could perhaps have been chosen slightly different to counteract the interations between the system, but as the system performs at this point, the paramaeters will be kept as is, as the controllers works their way back from these deviations within reasonable time. When it comes to the cold start up, we saw that the time until the system is properly running is long due to physical limitations when it comes to heat exchange. In our case we don't see any solution to this, perhaps something could be done upstream to boost this process or perhaps not. This is anyway not what we aimed to do in this report. The plant is now running pretty good, that was what we aimed at.

Bibliography

[1] J.G. Balchen, T. Andresen, and B.A. Foss. *Reguleringsteknikk*. Intitutt for teknisk kybernetikk, NTNU, 2003. ISBN: 8247151472.

[2] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. Wiley, 2005. ISBN: 9780470011676. URL: https://books.google.no/books?id=3dxSAAAAMAAJ.

Appendix A: MATLAB code

Listing A.1: MATLAB code to import and identify system of order nx using the command `n4sid`, before plotting estimated response together with the actual response of the system

```
%% cleanup
close all
clear variables
clc

%% Get data from the log file
% Make sure to use the correct path for the log file
fileID=fopen('Logging.lst.txt','r'); % This loads the data log
   file
for m = 1:35
    String_Row=fgetl(fileID); % Ignore first 35 rows in the txt
        file
end

i = 1;
while(ischar(String_Row));          % Continue until the end of
   file
       String_Row=fgetl(fileID);   % Read row from txt file
    if ischar(String_Row) ~= 0
        Num_Vector = str2num(String_Row);    % Converts string
           number a vector
        Data(i,:) = Num_Vector;              % Store rows into a
            "Data" Matrix
    end
     i = i + 1;
end
fclose(fileID); % Close the data log file

%% Controller data
Time = Data(:,1);            % Time
PC1024 = Data(:,2:4);        % Controller: 24_PC1024
FC1005 = Data(:,5:7);        % Controller: 24_FC1005
```

```
FC1019 = Data(:,8:10);       % Controller: 24_FC1019
LC1016 = Data(:,11:13);      % Controller: 24_LC1016
LC1015 = Data(:,14:16);      % Controller: 24_LC1015
FC1015 = Data(:,17:19);      % Controller: 24_FC1015
LC1028 = Data(:,20:22);      % Controller: 24_LC1028
TC1015 = Data(:,23:25);      % Controller: 24_TC1015
TC1088 = Data(:,26:28);      % Controller: 24_TC1088
% Where the:
% First column is the Process Value, PV
% Second column is the Set-Point, SP
% Third column is the Control Signal, OP
% Ex.
% PC1024(:,1) = Process Value, PV
% PC1024(:,2) = Set-Point, SP
% PC1024(:,3) = Control signal, OP

%% normalize data
cut = 1; %Where to start reading the data from

controller = LC1015; %Choose the controller we want
controller_cut = controller(cut:end,:);
y = controller_cut(:,1) - controller_cut(1,1);
y_ref =  controller_cut(:,2) - controller_cut(1,2);
u = controller_cut(:,3) - controller_cut(1,3);
Time_cut = Time(cut:end) - Time(cut);


%% system identification
Ts = 1; %Sample time
data = iddata(y,u,Ts); %Generate data of type iddata needed for
    n4sid
nx = 1; %Order of model
sys = n4sid(data,nx, 'inputdelay', 60); %Compute model with
   eventual input delay

%%
y_sim = lsim(sys,u);

%% plot
figure
plot(Time_cut./60, y, 'b'); hold on;
plot(Time_cut./60, y_sim, 'r');
plot(Time_cut./60, y_ref);
%plot(Time_cut./60, u);
```

```
xlabel('min');
legend('Process Value, PV',['Fitted model of order ' num2str(nx
   )] , 'Reference value');
```

Appendix B: Simulink diagrams

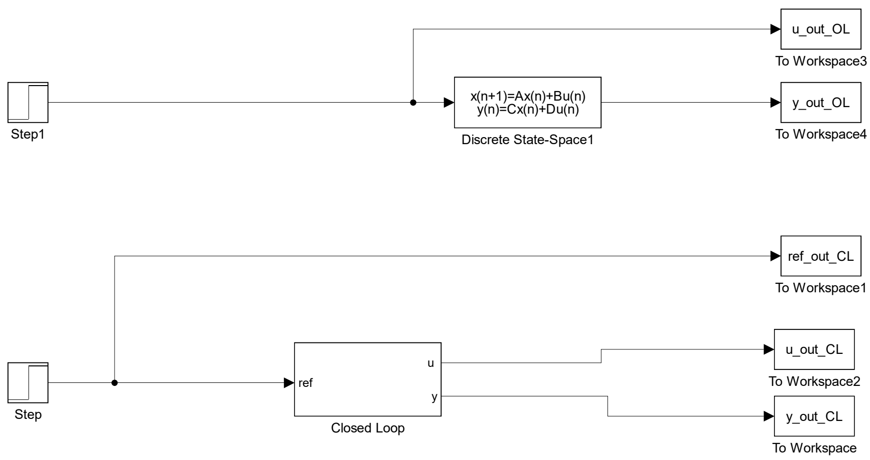

Figure B.1: General Simulink diagram of identified system with both open and closed loop system

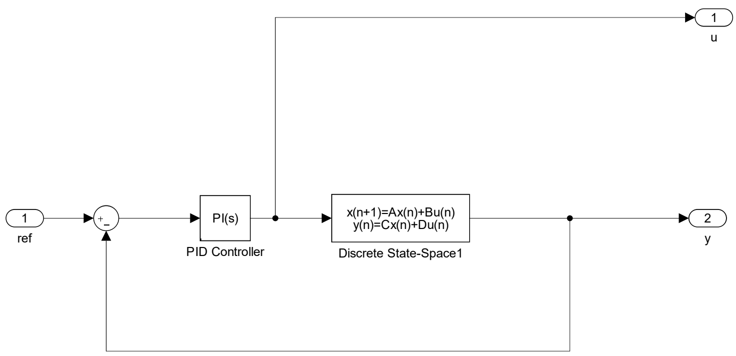

Figure B.2: Detailed diagram of the closed loop part of the system

Appendix C: MCL-scripts

Listing C.1: MCL script to run system for two hours and performing three steps to evaluate performance of controller

```
// Load inital Conditions
LoadConditions("ButaneSplitter")
// Run Model for 120 minutes
RunUntil([000:120:00.00])

// Set 24_TC1015 and 24_TC1088 in Manual Mode ("0" = Auto Mode, "1" = Manual Mode)
at ([01:00.00]) {24_TC1015:Mode} = 1
at ([01:00.00]) {24_TC1088:Mode} = 1

// Obtain value for Set-point / Controller Output
Internal_SP = {24_LC1016:InternalSetpoint}
// External_SP = {24_LC1016:ExternalSetpoint}
// Controller_Output = {24_LC1016:ControllerOutput}

at ([30:00.00]) {24_LC1016:InternalSetpoint} = Internal_SP + 0.5
at ([60:00.00]) {24_LC1016:InternalSetpoint} = Internal_SP - 0.5

// Set controllers back to their origninal operation
at ([75:00.00]) {24_LC1016:InternalSetpoint} = Internal_SP
// Set 24_TC1015 and 24_TC1088 in Auto Mode ("0" = Auto Mode, "1" = Manual Mode)
at ([115:00.00]) {24_TC1015:Mode} = 0
at ([115:00.00]) {24_TC1088:Mode} = 0
```

Listing C.2: MCL script to run system for five hours and performing several step changes in controller output of controller 24_TC1088.

```
// MCL Example

// Load inital Conditions
LoadConditions("ButaneSplitterStable")
// Run Model for 300 minutes
RunUntil([000:300:00.00])

// Set 24_TC1015 and 24_TC1088 in Manual Mode ("0" = Auto Mode, "1" = Manual Mode)
at ([60:00.00]) {24_TC1015:Mode} = 1
at ([60:00.00]) {24_TC1088:Mode} = 1


// Obtain value for Set-point / Controller Output
```

```
// Internal_SP = {24_TC1088:InternalSetpoint}
// External_SP = {24_TC1088:ExternalSetpoint}
Controller_Output = {24_TC1015:ControllerOutput}

at ([90:00.00]) {24_TC1015:ControllerOutput} = Controller_Output + 2
at ([120:00.00]) {24_TC1015:ControllerOutput} = Controller_Output - 2
at ([150:00.00]) {24_TC1015:ControllerOutput} = Controller_Output + 2.2
at ([180:00.00]) {24_TC1015:ControllerOutput} = Controller_Output - 1.5

// Set controllers back to their origninal operation
// at ([299:59.00]) {24_LC1015:InternalSetpoint} = Internal_SP
// Set 24_TC1015 and 24_TC1088 in Auto Mode ("0" = Auto Mode, "1" = Manual Mode)
at ([240:00.00]) {24_TC1015:Mode} = 0
at ([240:00.00]) {24_TC1088:Mode} = 0
```