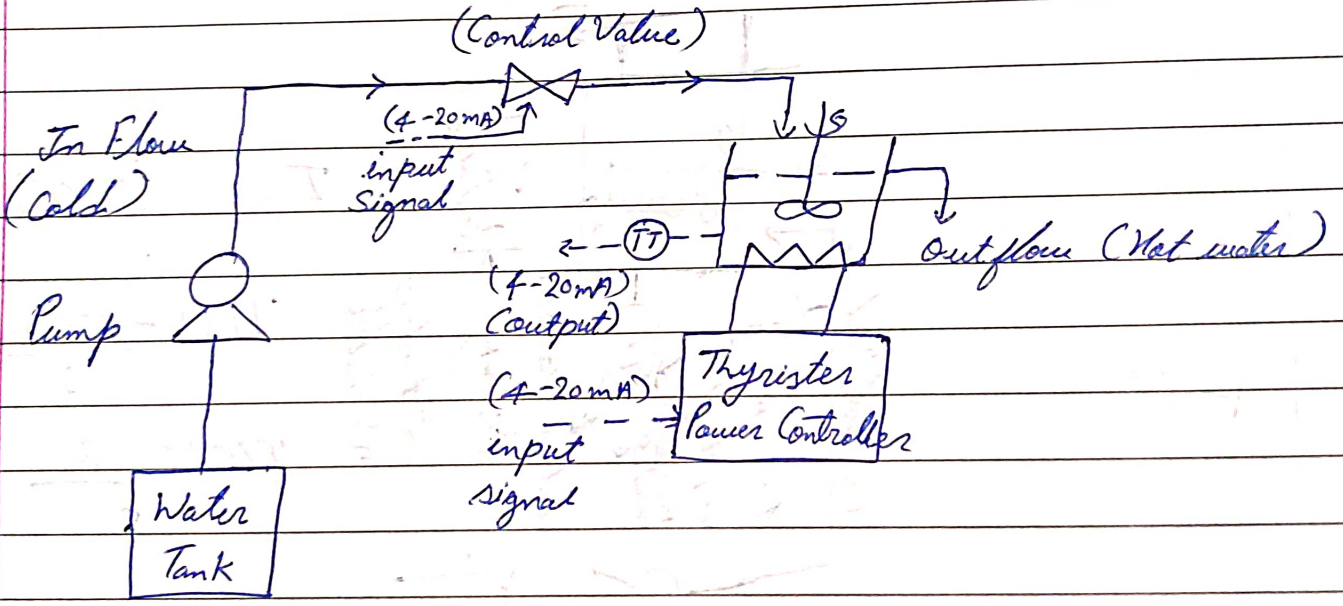
Open Loop Control

Aim :-

- To conduct step test experiments using stirred tank heater
- Use of experimental data for development of Transfer function models w.r.t manipulated input & disturbance unit
- Design P & PI controllers using Direct Synthesis Method.

Setup :-



Procedure :-

1) Switch on (pump & computer setup)
Set current at 4-20 mA. Examine the setup for steady-state. Height become constant with at time at steady state

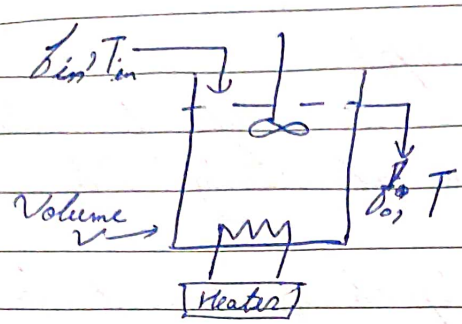
2) Now change the ~~flow rate~~ heat rate by step change current to 13.6 mA from 12 mA. Wait for the steady state to reach. Here $\Delta V = 1.6$ mA

3) Now again change the current to 12 mA & wait for steady state to reach.

4) Keep the flow constant & decrease the ~~flow~~ heat rate by changing current to 10.4 mA from 12 mA. & reach the steady state.
Again set the current value to 12 mA keeping the flow rate constant.

5) Repeat the above steps while keeping the heat rate constant & changing the flow rate. By giving step change of 1.6 mA (from 12 mA to 13.6 mA) & ~~10 mA~~ (from 12 mA to 10.4 mA).

Theory :-



Mass Balance

$$\rho \frac{dV}{dt} = f_{in} - f_o$$

At ~~stedt~~ steady state, $\frac{dV}{dt} = 0$

$$f_{in} = f_o$$

Energy Balance

Energy in − Energy out + Generation = Accumulation

$$\rho f_{in} C_p (T_{in} - T') - \rho f_o C_p (T - T') + Q = \rho V C_p \frac{dT}{dt}$$

Let $f_{in} = f_o = f$

$$\rho V C_p \frac{dT}{dt} = f C_p (T_{in} - T) + Q$$

$$\boxed{\frac{dT}{dt} = \frac{f}{V}(T_{in} - T) + \frac{Q}{\rho V C_p}}$$

U, manipulated variable = Q
D, Disturbance variable = f
Y, State variable = T

Date:

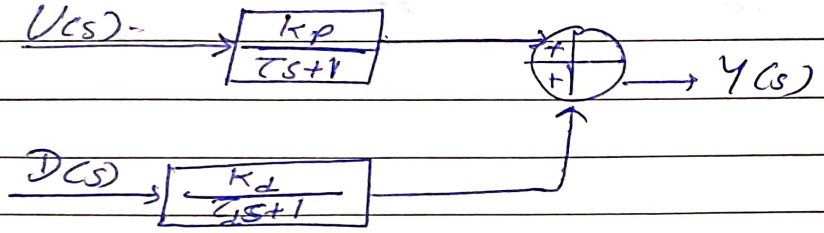
⇒ P & PI controller Design using Direct Synthesis

Consider a dynamic model of form

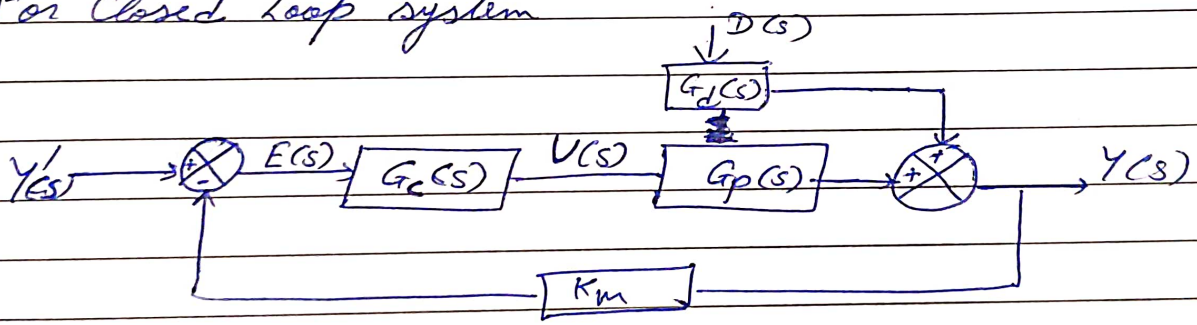
$$Y(s) = \frac{K_p}{\tau s+1} U(s) + \frac{K_d}{\tau_d s+1} D(s)$$

Block Diagram

→ Open Loop System



→ For Closed Loop system



~~E(s)~~ : For P controller, $G_c = K_c$

For PI controller, $G_c = K_c\left(1 + \frac{1}{\tau_I s}\right)$

$$G_p(s) = \frac{K_p}{\tau s+1} \qquad G_d = \frac{K_d}{\tau_d s+1} \qquad E(s) = Y'(s) - Y(s)$$

Sources of Error:-

→ External source of error like sources of disturbance, Tank temperature or shady environment

→ Heat dissipation from beaker & surrounding due to convection

→ Voltage function in power ~~sy~~ supply line that can be change RPM of motor

→ If sensor have time delay then it effect the output.

∵ Eqn is non-linear, ∴ we'll need to linearize it (by using taylor series expansion method)

here, derivation variables: $f' = f - \bar{f}$, $T' = T - \bar{T}$, $\phi' = \phi - \bar{\phi}$

($\bar{f}, \bar{\phi}, \bar{T}$ are steady state values)

$$f(T, f, \phi) = \rho V C_p \frac{dT}{dt}$$

$$\therefore \rho V C_p \frac{dT}{dt} = 0 + \frac{\partial}{\partial T}\left(f C_p (T_i - T) + \phi\right)\Big|_{T, \bar{f}, \phi} T' + \text{other similar term}$$

$$\rho V C_p \frac{dT}{dt} = 0 + (-\bar{f} C_p) T' + C_p (T_i - \bar{T}) f' + 1\phi'$$

by Taking Laplace.

$$\rho V_c [s T(s) - 0) - \bar{f} C_p T'(s) + C_p (T_i - \bar{T}) \cdot f'(s) + \phi'(s)$$

$$T'(s)\left[\rho V C_p s + \bar{f} C_p\right] = C_p (T_i - T') w'(s) + \phi'(s)$$

$$T'(s) = \frac{C_p (T_i - \bar{T})}{\bar{f} C} \cdot \frac{1}{\left(\frac{\rho V}{\bar{f}} s + 1\right)} f'(s) + \frac{1}{\bar{f} C_p} \frac{1}{\left(\frac{\rho V}{\bar{f}} s + 1\right)} \phi'(s)$$

$$\tau = \frac{\rho V}{\bar{f}}, \quad k_p = \frac{1}{\bar{f} C} \quad k_d = \frac{C_p (T_i - \bar{T})}{\bar{f} C_p}$$

Sample calculation →

1) from graphical method values of K & τ

for +ve step change in heater → $K_{ph} = 1.2248$, $\tau_{ph} = 148.7950$

for −ve step change in heater → $K_{nh} = 1.4114$, $\tau_{nh} = 122.4530$

2) from optimization method →

$$(K^*, \tau^*) = \min \sum_{T=T_0}^{T_f} \left[\int Y(T_i) - \int \hat{Y}(T_i) \right]^2$$

Where $\int \hat{Y}(T_i) = K \Delta U \left[1 - \exp\left(-T_i/\tau\right) \right]$

i) step change in flow rate, constant heater input

for +ve step change $K_{pf} = -3.2205$, $\tau_{pf} = 130.4005$

for −ve step change $K_{n,f} = -7.4475$, $\tau_{n,f} = 301.4070$

$$K_f^* = \frac{K_{pf} + K_{nf}}{2} \Rightarrow K_f^* = -5.3340$$

$$\tau_f^* = \frac{\tau_{pf} + \tau_{nf}}{2} \Rightarrow \tau_f^* = 215.9037$$

* Units of process gain K_p is in °C/mA
* unit of time constant (τ) is in sec.

ii) Similarly step changes in heater in put

for positive step change $K_{ph} = 1.2970$, $\tau_{ph} = 172.4255$

for Negative step change $K_{nh} = 1.4837$, $\tau_{nh} = 124.5298$

$K_p^* = 1.3903$, $\tau_p = 148.427$

* P & PI controls designs:

parameters.	P	PI
	$\alpha_1 = 1.6$	$\alpha = 2.15$
	$\alpha_2 = 1.65$	$\xi_1 = 0.8$
		$\xi_2 = 1.21$

for P- controller Design →

$$K_c = \frac{\alpha - 1}{K_p}, \quad K_p = K_p^* = 1.3903$$

$$K_{c_1} = \frac{\alpha_1 - 1}{K_p^*} = \frac{0.6}{1.3903} = 0.4315$$

$$K_{c_2} = \frac{\alpha_2 - 1}{K_p^*} = \frac{1.65}{1.3903} = 1.1867$$

for PI controller Design →

$\alpha = 2.15$

$$K_c = \frac{2\alpha\xi - 1}{K_p} \qquad \tau_I = \frac{(2\alpha\xi_2 - 1)\tau_p}{\alpha^2}$$

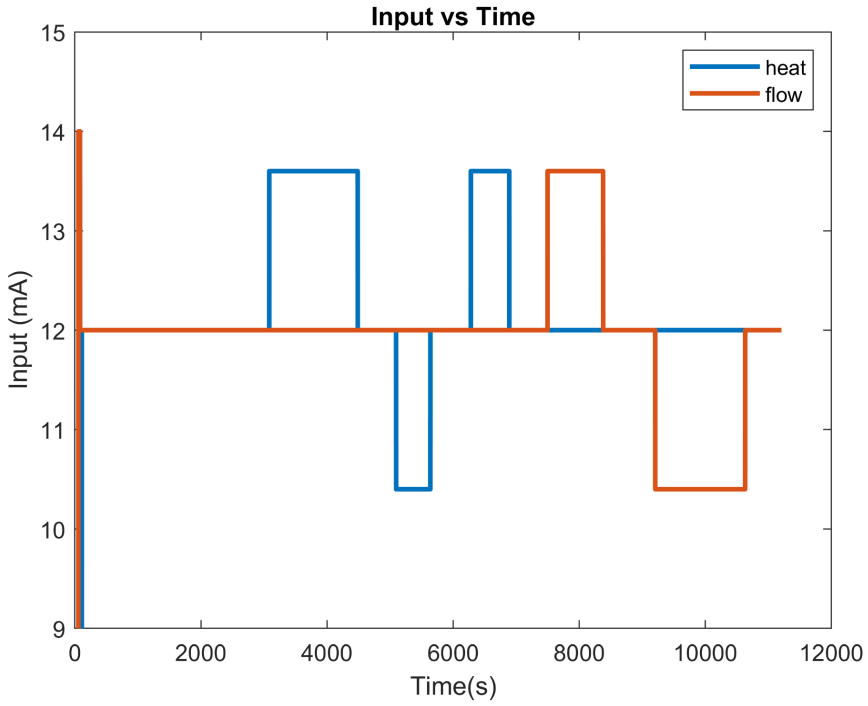
$$K_{c_1} = \frac{2 \times 2.15 \times 0.8 - 1}{1.3903} \qquad \tau_{I_1} = \frac{(2 \times 2.15 \times 0.8 - 1)}{(2.15)^2}\tau_p$$

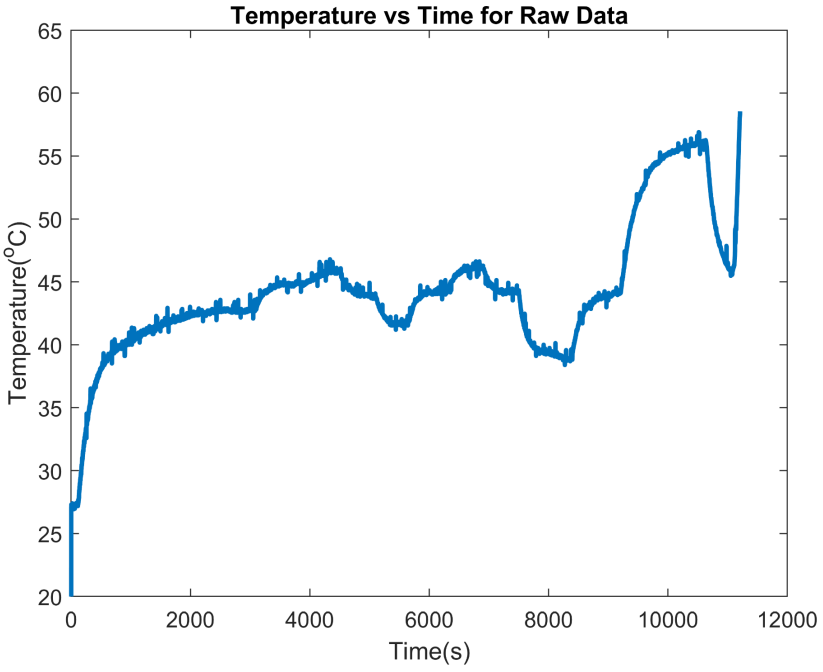
for $\xi_1 = 0.8 \rightarrow K_{c_1} = 1.7550 \quad \tau_1 = 78.3740$

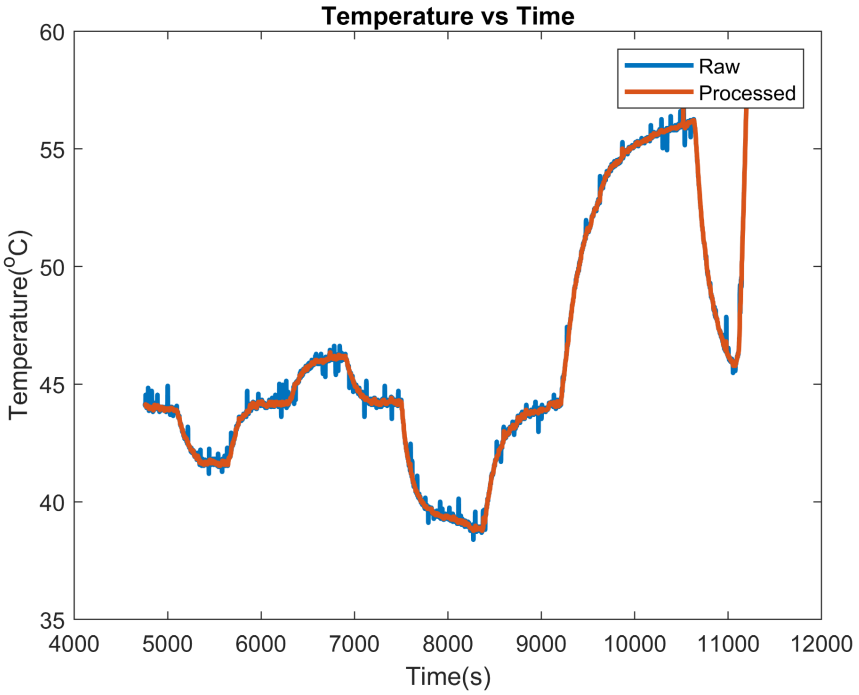
for $\xi_2 = 1.21 \rightarrow K_{c_2} = 3.0231 \quad \tau_2 = 135.0025$

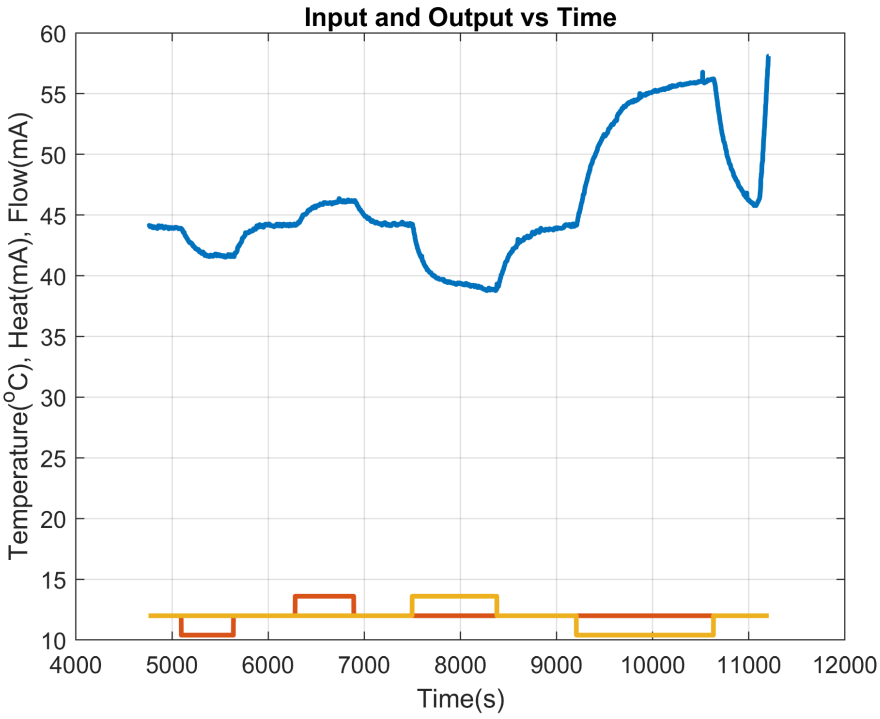
7. RESULTS

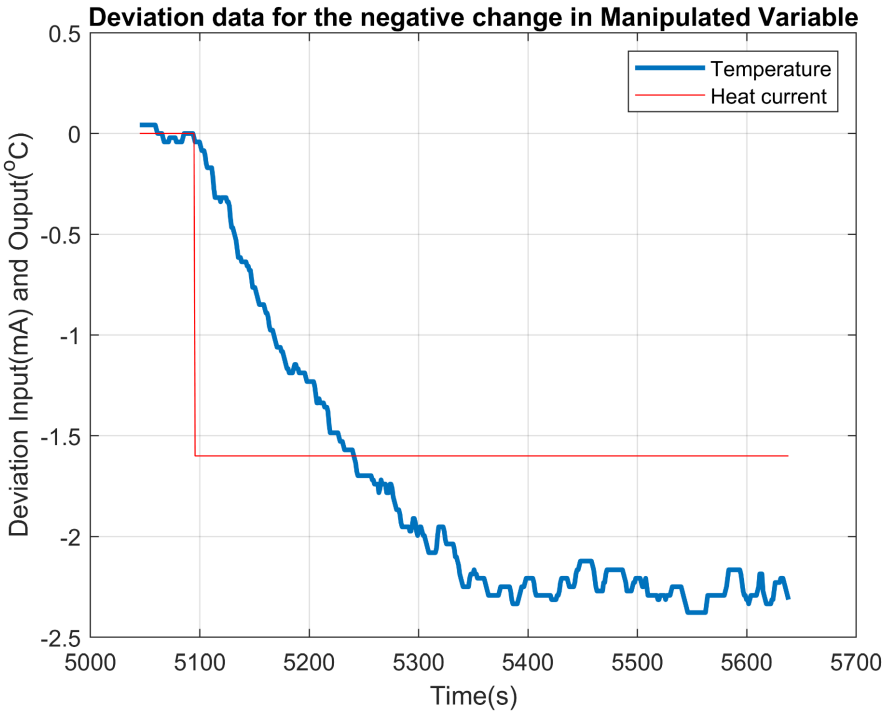
Step Change	Graphical Method				Optimisation Method			
	K	**K_{avg}**	**τ**	**τ_{avg}**	**K**	**K_{avg}**	**τ**	**τ_{avg}**
Manipulated Variable (Heat rate)								
+ve	1.2248	**1.3181**	148.7650	**135.609**	1.2970	**1.39035**	172.4255	**148.4776**
-ve	1.4114		122.4530		1.4837		124.5298	
Disturbance Variable (flow rate)								
+ve	-3.3613	**-5.4263**	134.5620	**228.132**	-3.2205	**-5.334**	130.4005	**215.9037**
-ve	-7.4914		321.7030		-7.4475		301.4070	

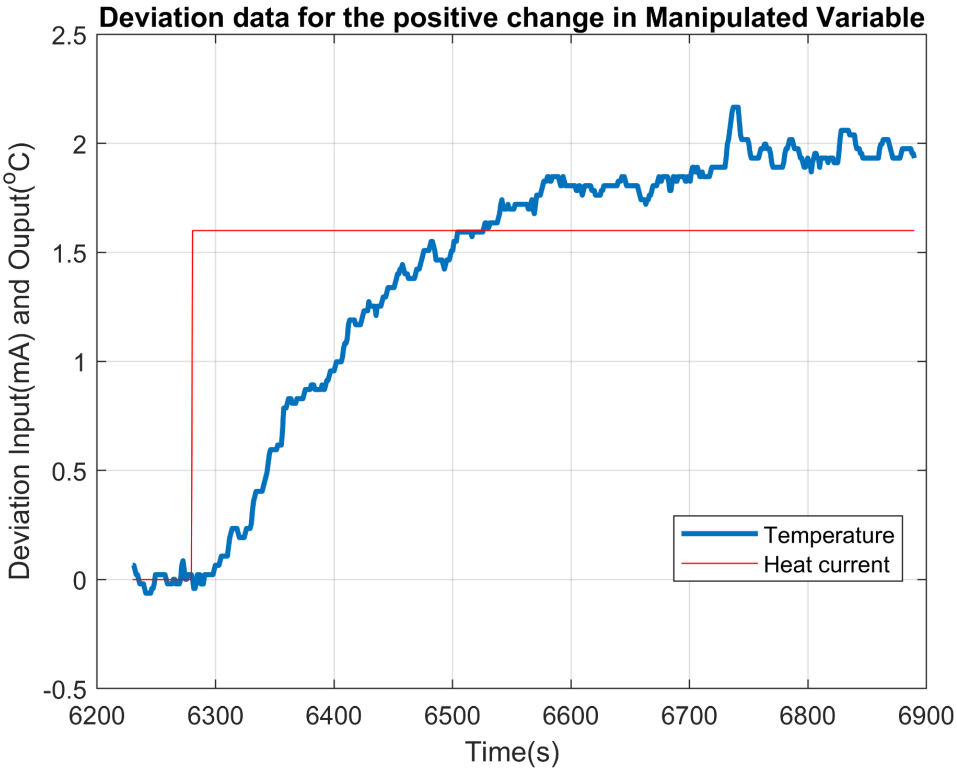

Input vs Time
heat
flow
Input (mA)
Time(s)
15
14
13
12
11
10
9
0
2000
4000
6000
8000
10000
12000

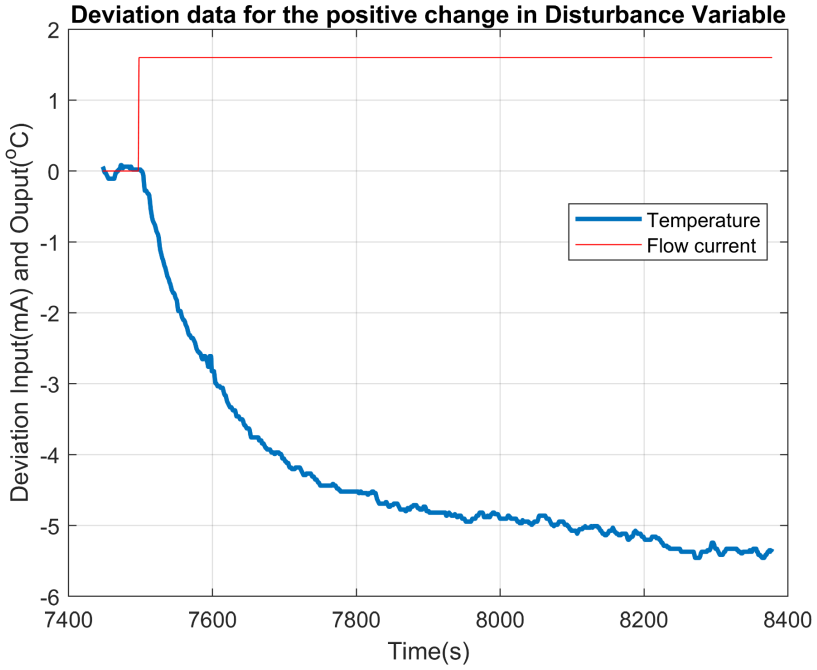

Temperature vs Time for Raw Data
Temperature(°C)
Time(s)
65
60
55
50
45
40
35
30
25
20
0
2000
4000
6000
8000
10000
12000

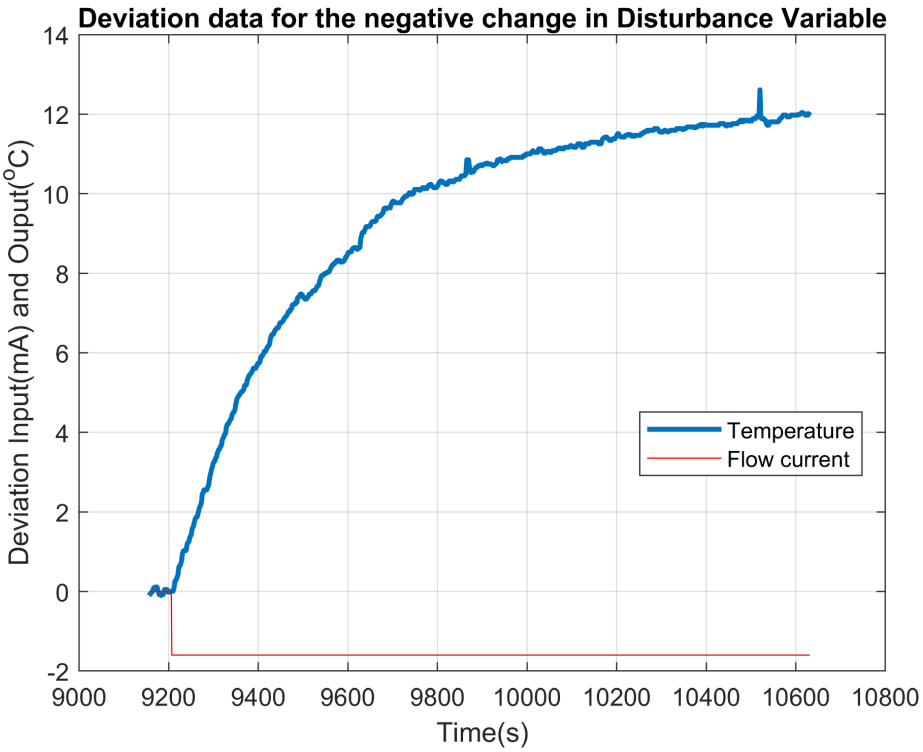

Temperature vs Time
Raw
Processed
Temperature(°C)
Time(s)
60
55
50
45
40
35
4000
5000
6000
7000
8000
9000
10000
11000
12000

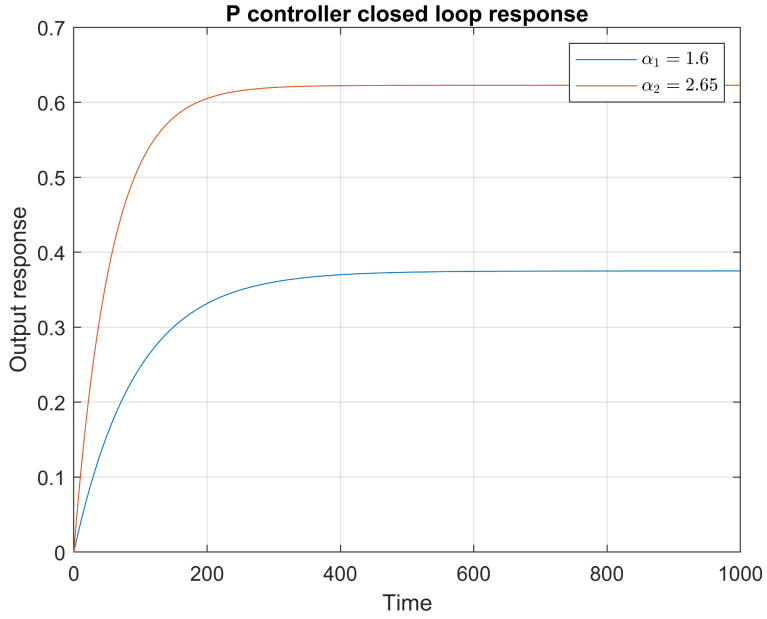

Input and Output vs Time
Temperature(°C), Heat(mA), Flow(mA)
Time(s)
60
55
50
45
40
35
30
25
20
15
10
4000
5000
6000
7000
8000
9000
10000
11000
12000

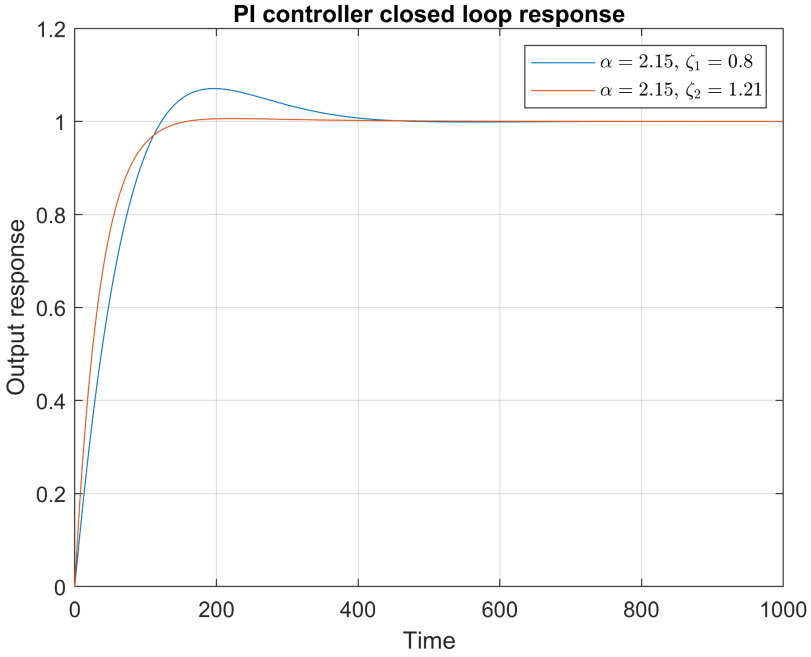

Deviation data for the negative change in Manipulated Variable
Temperature
Heat current
Deviation Input(mA) and Ouput(°C)
Time(s)
0.5
0
-0.5
-1
-1.5
-2
-2.5
5000
5100
5200
5300
5400
5500
5600
5700

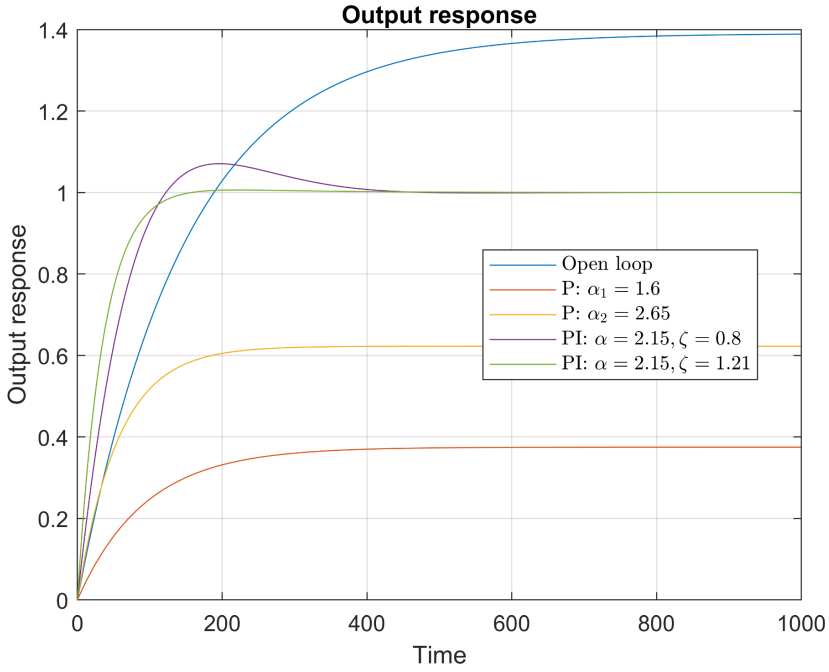

Deviation data for the positive change in Manipulated Variable
Temperature
Heat current
Deviation Input(mA) and Ouput(°C)
Time(s)
2.5
2
1.5
1
0.5
0
-0.5
6200
6300
6400
6500
6600
6700
6800
6900

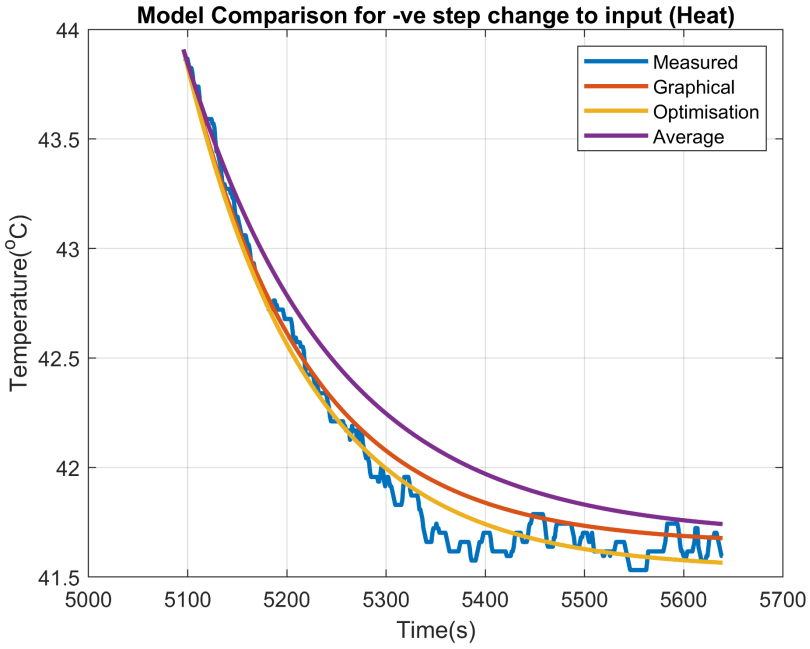

Deviation data for the positive change in Disturbance Variable
Deviation Input(mA) and Ouput(°C)
Time(s)
Temperature
Flow current
2
1
0
-1
-2
-3
-4
-5
-6
7400
7600
7800
8000
8200
8400

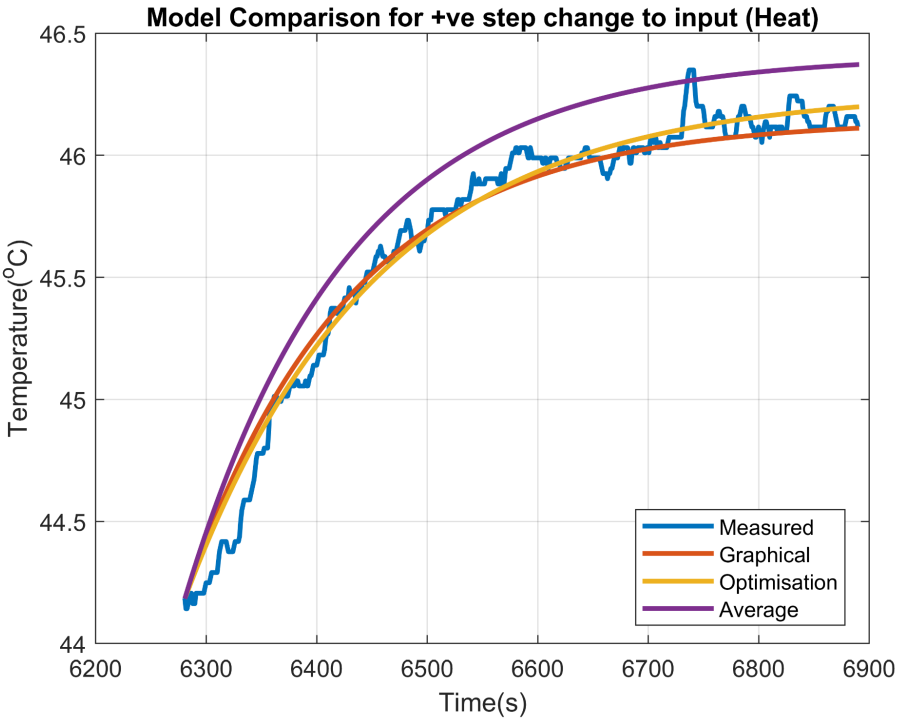

Deviation data for the negative change in Disturbance Variable
Deviation Input(mA) and Ouput(°C)
Time(s)
Temperature
Flow current
14
12
10
8
6
4
2
0
-2
9000
9200
9400
9600
9800
10000
10200
10400
10600
10800

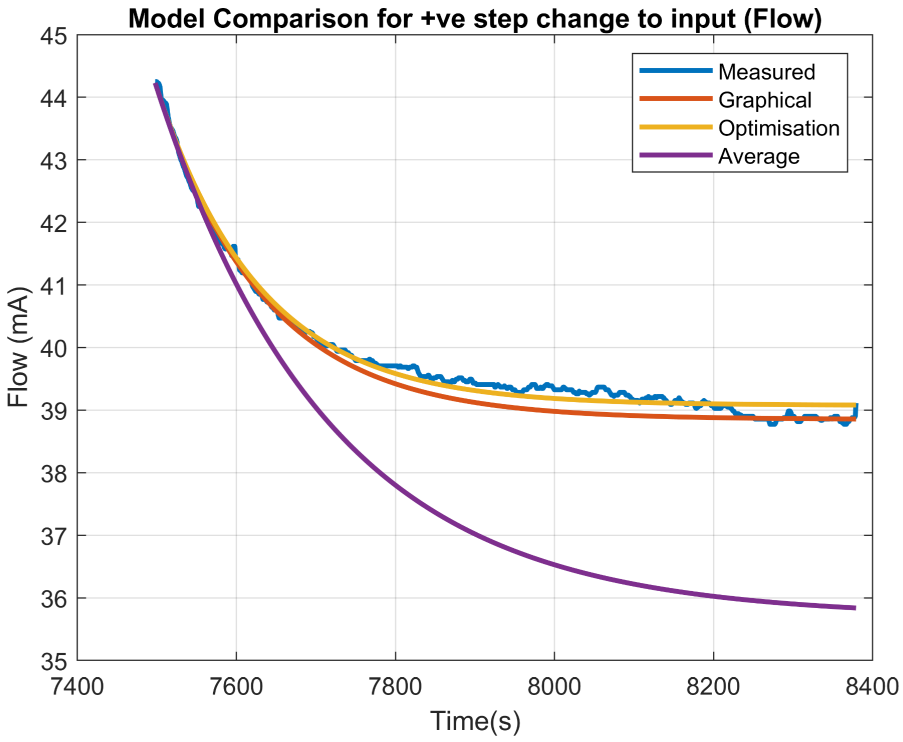

P controller closed loop response
$\alpha_1 = 1.6$
$\alpha_2 = 2.65$
Output response
Time
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
200
400
600
800
1000

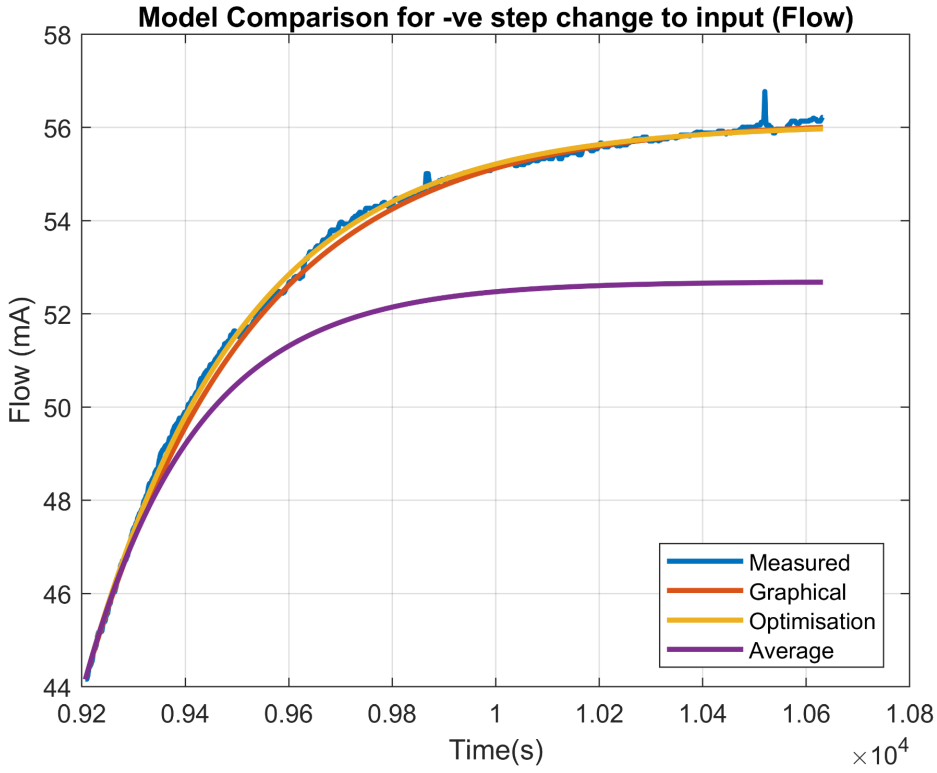

PI controller closed loop response
$\alpha = 2.15, \zeta_1 = 0.8$
$\alpha = 2.15, \zeta_2 = 1.21$
Output response
Time
0
0.2
0.4
0.6
0.8
1
1.2
0
200
400
600
800
1000


Output response
Output response
Time
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
200
400
600
800
1000
Open loop
P: $\alpha_1 = 1.6$
P: $\alpha_2 = 2.65$
PI: $\alpha = 2.15, \zeta = 0.8$
PI: $\alpha = 2.15, \zeta = 1.21$


Model Comparison for -ve step change to input (Heat)
Measured
Graphical
Optimisation
Average
Temperature(°C)
Time(s)
44
43.5
43
42.5
42
41.5
5000
5100
5200
5300
5400
5500
5600
5700


Model Comparison for +ve step change to input (Heat)
Measured
Graphical
Optimisation
Average
Temperature(°C)
Time(s)
46.5
46
45.5
45
44.5
44
6200
6300
6400
6500
6600
6700
6800
6900


Model Comparison for +ve step change to input (Flow)
Measured
Graphical
Optimisation
Average
Flow (mA)
Time(s)
45
44
43
42
41
40
39
38
37
36
35
7400
7600
7800
8000
8200
8400


Model Comparison for -ve step change to input (Flow)
Measured
Graphical
Optimisation
Average
Flow (mA)
Time(s)
×10^4
58
56
54
52
50
48
46
44
0.92
0.94
0.96
0.98
1
1.02
1.04
1.06
1.08

Hypothesis and Discussion:

→ Possible reasons for assymmetric behaviour.

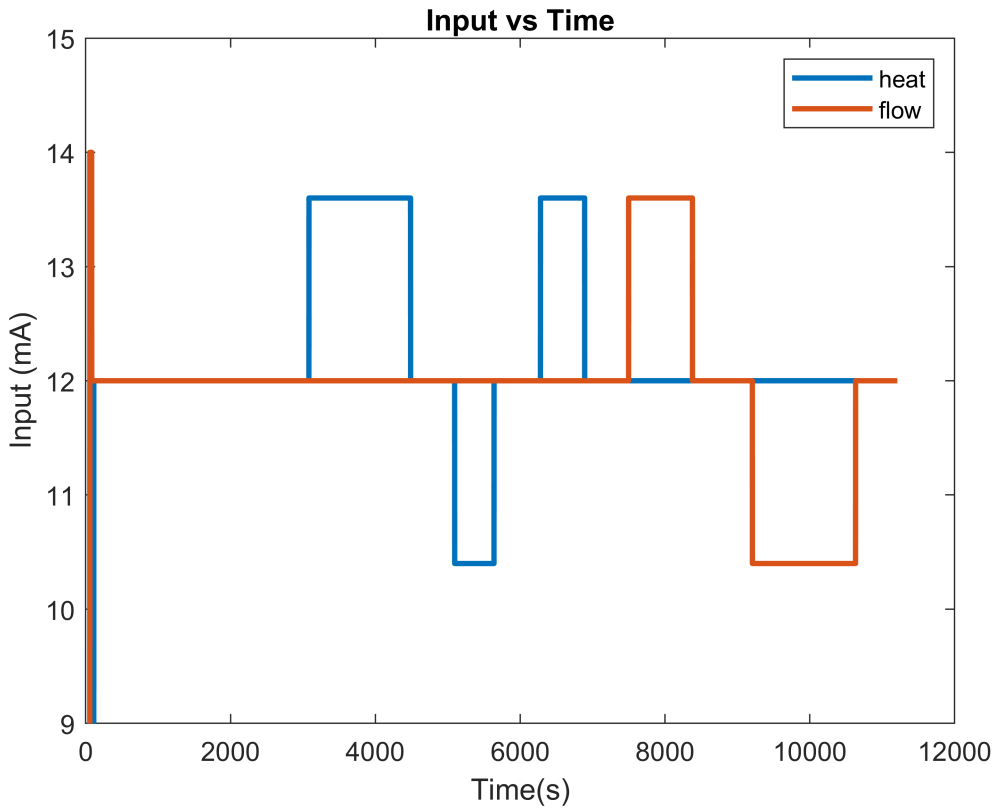
1] Non linearity of the process equation and the linearization throug Taylor expansion around steady state to obtain the transfer function.

2] Non linearity in other component such as valve and the Time delay of the sensors.

→ Closed loop have smaller settling time than open loop as observed from the comparison graph of output response with time.

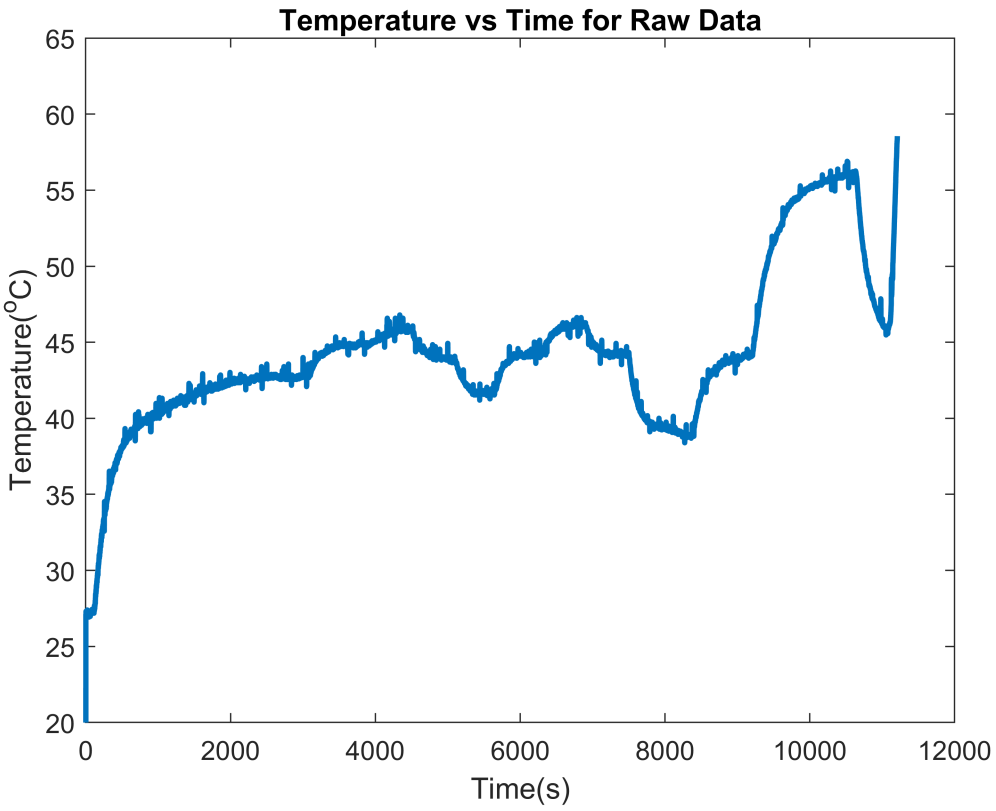
Also for the PI controller where $\zeta = 0.8$ or $(\zeta < 1)$ the response shows oscillatory behaviour due to the imaginary poles of the transfer function.

```
clear
clc
%%% Loading Data
data = load('B6A_OPENLOOP.txt');
n = 11078;  % number of rows of data
time = data(1:n,1);
temp = data(1:n,2);
heat = data(1:n,3);
flow = data(1:n,4);
dU = 1.6;
U=12;

plot(time,heat,LineWidth=2)
hold on
plot(time,flow,LineWidth=2)
hold off
legend('heat','flow')
ylim([9,15])
xlabel("Time(s)")
ylabel("Input (mA)")
title("Input vs Time")
```




```
plot(time,temp,LineWidth=2)
xlabel("Time(s)")
ylim([20,65])
```

```
ylabel("Temperature(^oC)")
title("Temperature vs Time for Raw Data")
```




```

```

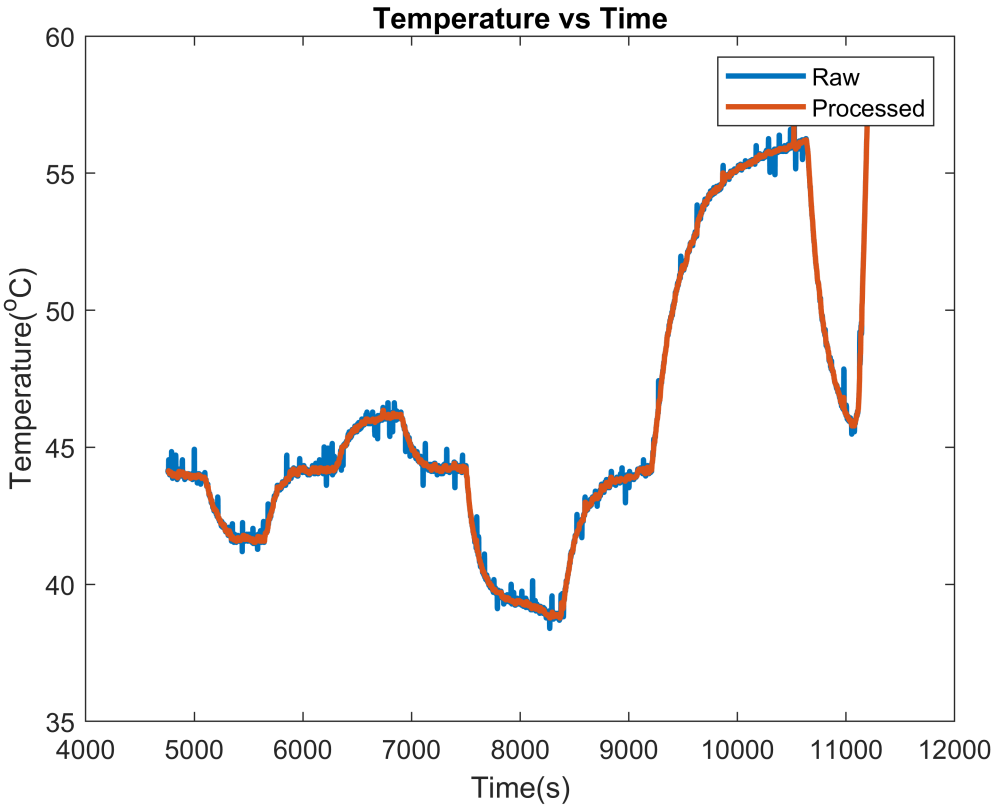
Our Initial Setting up values were errorneous, so slicing the initial part.

```
a = 4700;
time = data(a:n,1);
temp = data(a:n,2);
heat = data(a:n,3);
flow = data(a:n,4);

plot(time,temp,LineWidth=2)
hold on
xlabel("Time(s)")
%ylim([20,65])
ylabel("Temperature(^oC)")
title("Temperature vs Time")
```

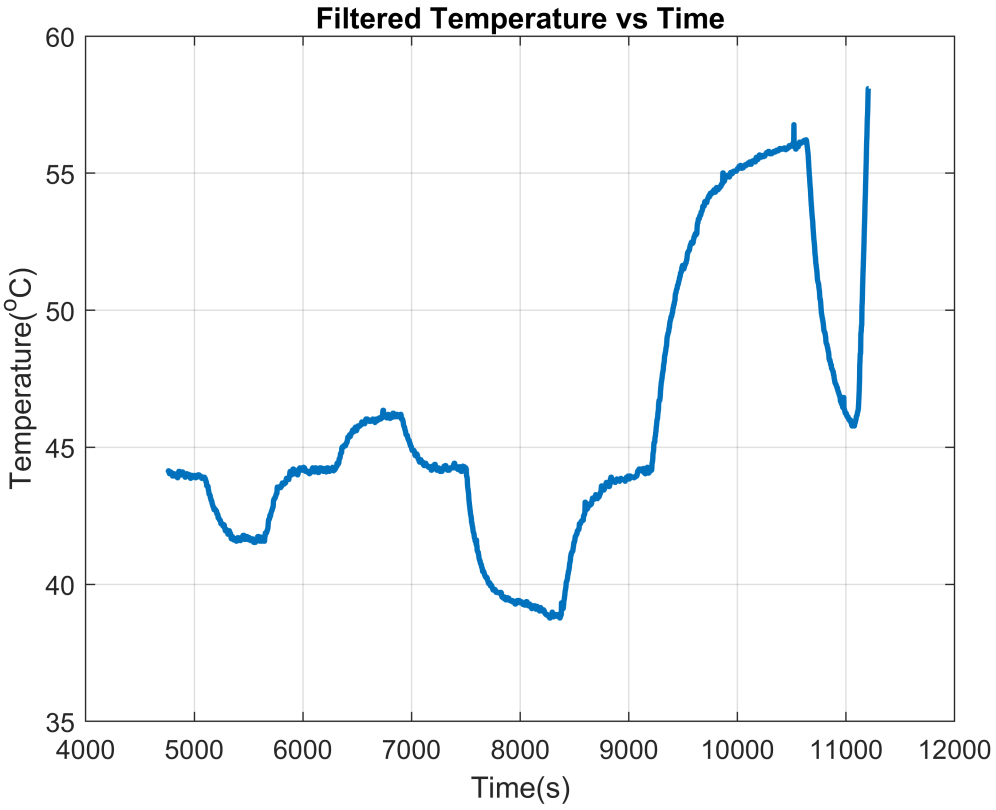
Sharp Spikes in the data is due to the noise in the measurement. Removing the error

```
% requires signal processing toolbox
temp_filtered = medfilt1(temp,10);
temp_filtered(1) =  temp_filtered(2);
plot(time,temp_filtered,LineWidth=2)
hold off
```

```
legend('Raw','Processed')
```

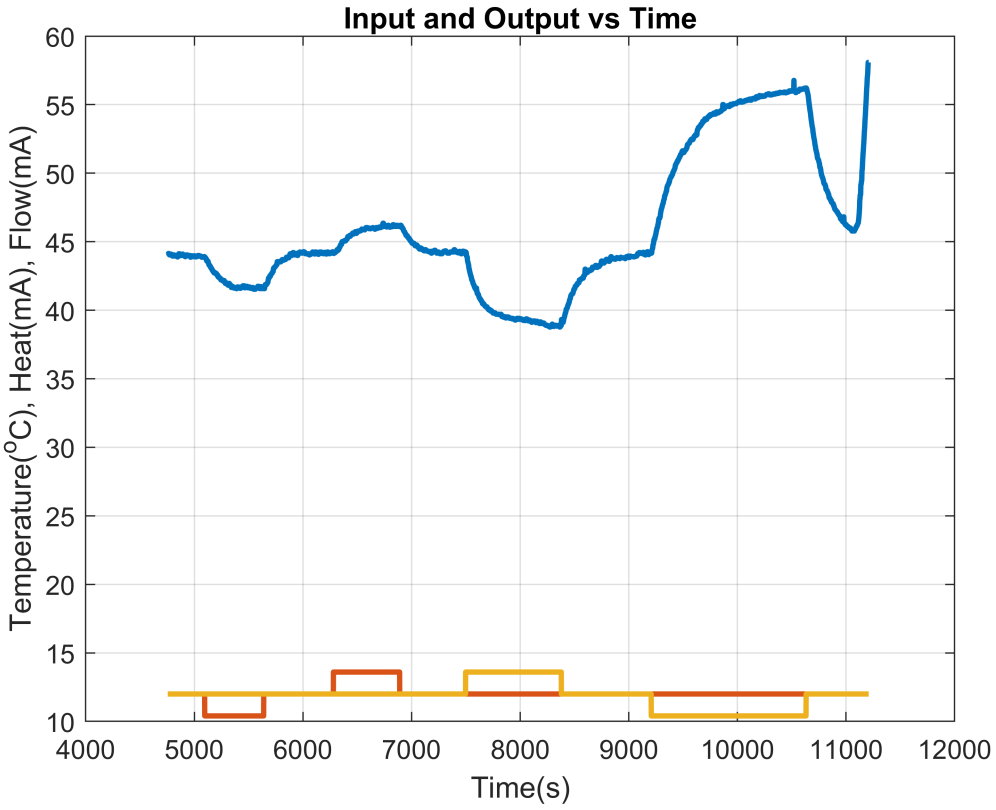



```
plot(time,temp_filtered,LineWidth=2)
title("Filtered Temperature vs Time")
grid on
xlabel("Time(s)")
ylabel("Temperature(^oC)")
```



A plot with input and output together

```
plot(time,temp_filtered,LineWidth=2)
hold on
title("Input and Output vs Time")
grid on
xlabel("Time(s)")
ylabel("Temperature(^oC), Heat(mA), Flow(mA)")
plot(time,heat,LineWidth=2)
plot(time,flow,LineWidth=2)
hold off
```



Determining the Steady State Values

```
SS_U1 = find(heat==U-dU,1);
temp_SS_U1 = mean(temp_filtered(SS_U1-50:SS_U1-1)) %steady-state temp
```

```
temp_SS_U1 = 43.9090
```

```
heat_SS_U1 = mean(heat(SS_U1-50:SS_U1-1)) %steady-state heat current
```

```
heat_SS_U1 = 12
```

For Changing Manipulated Variable (Heater)

```
SS_U_N_H = find(heat==U & time>5500,1);
temp_SS_U_N_H = mean(temp_filtered(SS_U_N_H-50:SS_U_N_H-1)) %steady-state temp
```

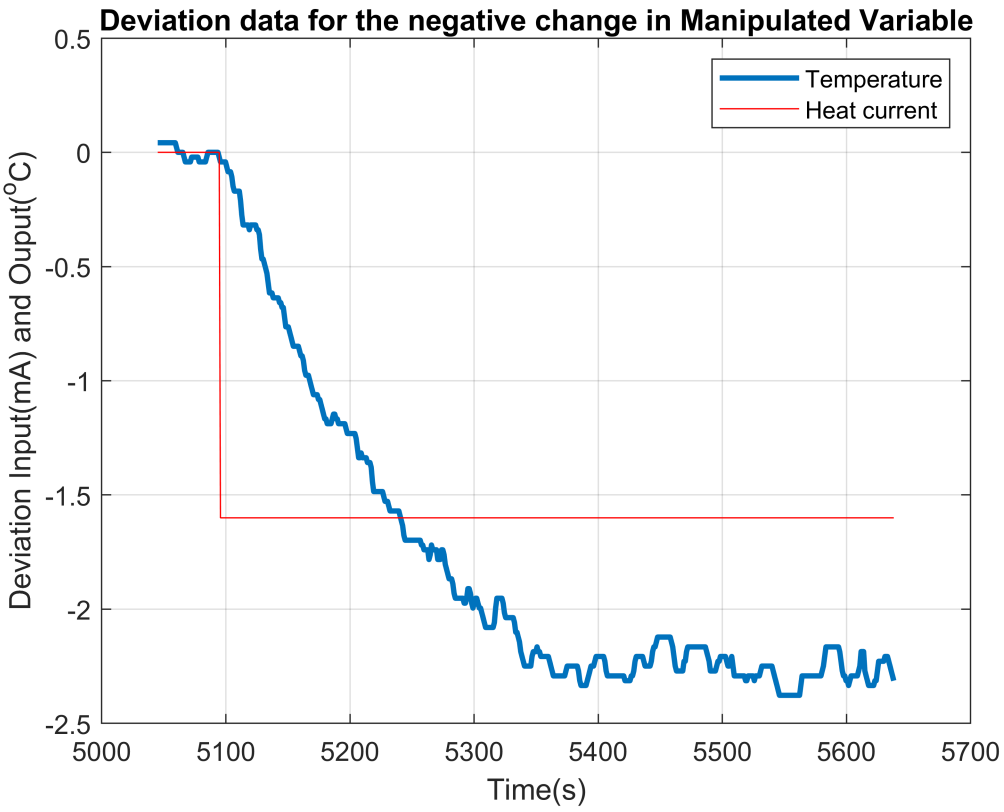
```
temp_SS_U_N_H = 41.6507
```

```
heat_SS_U_N_H = mean(heat(SS_U_N_H-50:SS_U_N_H-1)) %steady-state heat
```

```
heat_SS_U_N_H = 10.4000
```

PLOTTING for negative input change (Manipulated)

```
plot(time(SS_U1-50: SS_U_N_H-1),temp_filtered(SS_U1 - 50:SS_U_N_H-1)- temp_SS_U1,LineWidth=2)
hold on
plot(time(SS_U1-50: SS_U_N_H-1),heat(SS_U1-50:SS_U_N_H-1)- heat_SS_U1,"r")
hold off
title("Deviation data for the negative change in Manipulated Variable")
legend("Temperature","Heat current")
xlabel("Time(s)")
ylabel("Deviation Input(mA) and Ouput(^oC)")
grid on
```



Method : Calculating K and tau

```
% Graphical
K_N_H = (temp_SS_U_N_H-temp_SS_U1)/(heat_SS_U_N_H-heat_SS_U1)
```

```
K_N_H = 1.4114
```

```
yc = 0.633*(temp_SS_U_N_H-temp_SS_U1);

if(yc<=0)
 n_c = find(temp_filtered - temp_SS_U1 - yc <= 0,1);
else
 n_c = find(temp_filtered-temp_SS_U1 - yc >= 0,1);
end
t_c = time(n_c);
% t(find(heater-heater_s,1))
% t(idx_s)
```

```
tau_N_H= t_c-time(SS_U1)
```

```
tau_N_H = 122.4530
```

```
% Optimization Method
%del_N_H_heat = heat(SS_U1:SS_U_N_H)-heat_SS_U_N_H;
del_N_H_temp = temp_filtered(SS_U1:SS_U_N_H)-temp_SS_U1;
t_NH = time(SS_U1:SS_U_N_H)-time(SS_U1);

f = @(x)sum(((x(1).*(-dU).*(1-exp(-t_NH/x(2)))) - del_N_H_temp).^2);

x0=[K_N_H,tau_N_H];
[x,~] = fminunc(f,x0);
```

```
Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```

```
K_N_H_opt = x(1)
```

```
K_N_H_opt = 1.4837
```

```
tau_N_H_opt = x(2)
```

```
tau_N_H_opt = 124.5298
```

```
SS_U2 = find(heat==U+dU,1);
temp_SS_U2 = mean(temp_filtered(SS_U2-50:SS_U2-1)) %steady-state temp
```

```
temp_SS_U2 = 44.1837
```

```
heat_SS_U2 = mean(heat(SS_U2-50:SS_U2-1)) %steady-state heat current
```

```
heat_SS_U2 = 12
```

```
SS_U_P_H = find(heat==U & time>6500,1);
temp_SS_U_P_H = mean(temp_filtered(SS_U_P_H-50:SS_U_P_H-1)) %steady-state temp
```

```
temp_SS_U_P_H = 46.1434
```

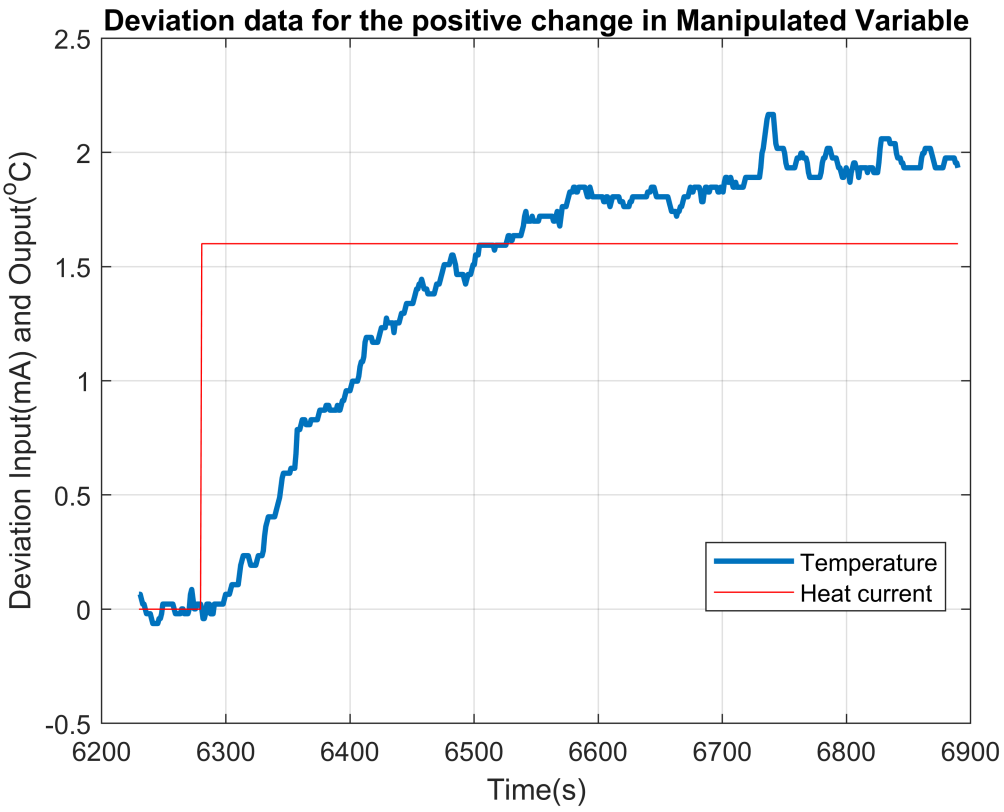
```
heat_SS_U_P_H = mean(heat(SS_U_P_H-50:SS_U_P_H-1)) %steady-state heat
```

```
heat_SS_U_P_H = 13.6000
```

```
```

PLOTTING for positive input change (Manipulated)

```
plot(time(SS_U2-50: SS_U_P_H-1),temp_filtered(SS_U2 - 50:SS_U_P_H-1)- temp_SS_U2,LineWidth=2)
hold on
plot(time(SS_U2-50: SS_U_P_H-1),heat(SS_U2-50:SS_U_P_H-1)- heat_SS_U2,"r")
hold off
title("Deviation data for the positive change in Manipulated Variable")
legend("Temperature","Heat current",Location="best" )
xlabel("Time(s)")
ylabel("Deviation Input(mA) and Ouput(^oC)")
grid on
```



Method : Calculating K and tau

```
% Graphical
K_P_H = (temp_SS_U_P_H-temp_SS_U2)/(heat_SS_U_P_H-heat_SS_U2)
```

```
K_P_H = 1.2248
```

```
yc = 0.633*(temp_SS_U_P_H-temp_SS_U2);

if(yc<=0)
 n_c = find(temp_filtered - temp_SS_U2 - yc <= 0,1);
else
 n_c = find(temp_filtered-temp_SS_U2 - yc >= 0,1);
end
t_c = time(n_c);
% t(find(heater-heater_s,1))
% t(idx_s)
```

```
tau_P_H= t_c-time(SS_U2)
```

```
tau_P_H = 148.7650
```

```
% Optimization Method
%del_P_H_heat = heat(SS_U2:SS_U_P_H)-heat_SS_U_P_H;
del_P_H_temp = temp_filtered(SS_U2:SS_U_P_H)-temp_SS_U2;
t_PH = time(SS_U2:SS_U_P_H)-time(SS_U2);

f = @(x)sum(((x(1).*(dU).*(1-exp(-t_PH/x(2)))) - del_P_H_temp).^2);

x0=[K_P_H,tau_P_H];
[x,~] = fminunc(f,x0);
```

```
Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```

```
K_P_H_opt = x(1)
```

```
K_P_H_opt = 1.2970
```

```
tau_P_H_opt = x(2)
```

```
tau_P_H_opt = 172.4255
```

```
SS_U3 = find(flow==U+dU,1);
temp_SS_U3 = mean(temp_filtered(SS_U3-50:SS_U3-1)) %steady-state temp
```

```
temp_SS_U3 = 44.2291
```

```
heat_SS_U3 = mean(heat(SS_U3-50:SS_U3-1)) %steady-state heat current
```

```
heat_SS_U3 = 12
```

```
flow_SS_U3 = mean(flow(SS_U3-50:SS_U3-1)) %steady-state flow
```

```
flow_SS_U3 = 12
```

Now For Disturbance (Flow)

```
SS_U_P_F = find(flow==U & time>8000,1);
temp_SS_U_P_F = mean(temp_filtered(SS_U_P_F-50:SS_U_P_F-1)) %steady-state temp
```

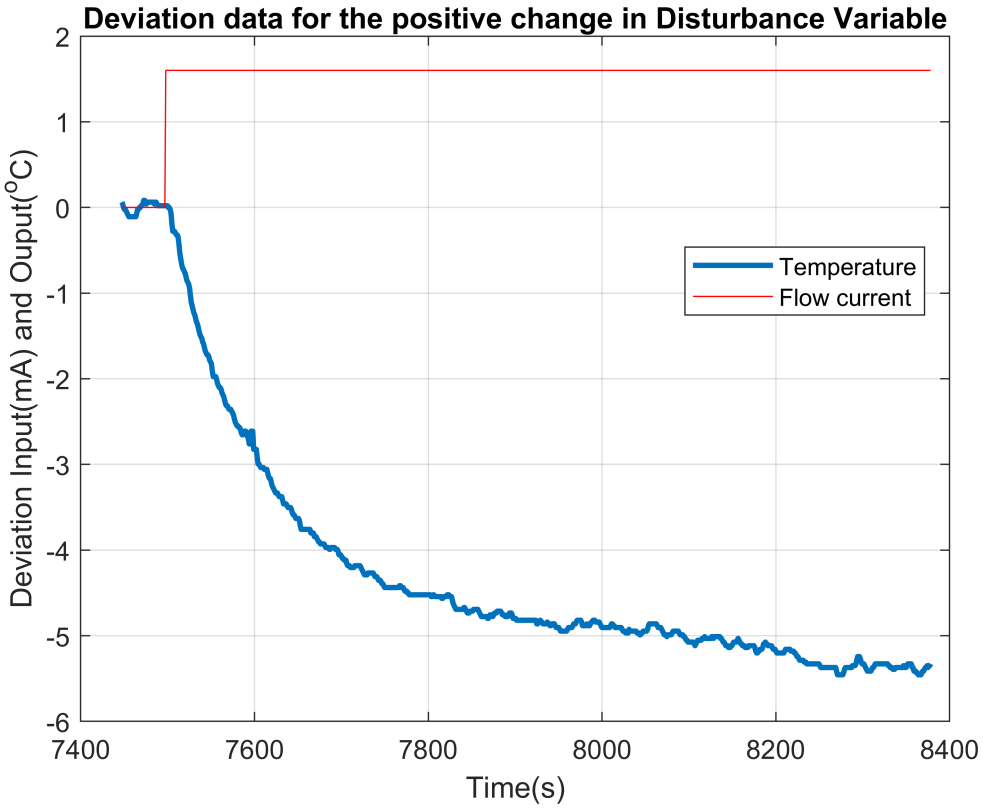
```
temp_SS_U_P_F = 38.8510
```

```
flow_SS_U_P_F = mean(flow(SS_U_P_F-50:SS_U_P_F-1)) %steady-state flow
```

```
flow_SS_U_P_F = 13.6000
```

PLOTTING for positive input change (Disturbance)

```
plot(time(SS_U3-50: SS_U_P_F-1),temp_filtered(SS_U3 - 50:SS_U_P_F-1)- temp_SS_U3,LineWidth=2)
hold on
plot(time(SS_U3-50: SS_U_P_F-1),flow(SS_U3-50:SS_U_P_F-1)- flow_SS_U3,"r")
hold off
title("Deviation data for the positive change in Disturbance Variable")
legend("Temperature","Flow current",Location="best" )
xlabel("Time(s)")
ylabel("Deviation Input(mA) and Ouput(^oC)")
grid on
```



Method : Calculating K and tau

```
% Graphical
K_P_F = (temp_SS_U_P_F-temp_SS_U3)/(flow_SS_U_P_F-flow_SS_U3)
```

```
K_P_F = -3.3613
```

```
yc = 0.633*(temp_SS_U_P_F-temp_SS_U3);

if(yc<=0)
 n_c = find(temp_filtered - temp_SS_U3 - yc <= 0,1);
else
 n_c = find(temp_filtered-temp_SS_U3 - yc >= 0,1);
```

```
end
t_c = time(n_c);
% t(find(heater-heater_s,1))
% t(idx_s)
tau_P_F= t_c-time(SS_U3)
```

```
tau_P_F = 134.5620
```

```

% Optimization Method
%del_P_F_heat = heat(SS_U3:SS_U_P_F)-heat_SS_U_P_F;
del_P_F_temp = temp_filtered(SS_U3:SS_U_P_F)-temp_SS_U3;
t_PF = time(SS_U3:SS_U_P_F)-time(SS_U3);

f = @(x)sum(((x(1).*(dU).*(1-exp(-t_PF/x(2)))) - del_P_F_temp).^2);

x0=[K_P_F,tau_P_F];
[x,~] = fminunc(f,x0);
```

```
Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```

```

K_P_F_opt = x(1)
```

```
K_P_F_opt = -3.2205
```

```
tau_P_F_opt = x(2)
```

```
tau_P_F_opt = 130.4005
```

```

```

```
SS_U4 = find(flow==U-dU,1);
temp_SS_U4 = mean(temp_filtered(SS_U4-50:SS_U4-1)) %steady-state temp
```

```
temp_SS_U4 = 44.1567
```

```
flow_SS_U4 = mean(flow(SS_U4-50:SS_U4-1)) %steady-state flow
```

```
flow_SS_U4 = 12
```

```

SS_U_N_F = find(flow==U & time>9500,1);
temp_SS_U_N_F = mean(temp_filtered(SS_U_N_F-50:SS_U_N_F-1)) %steady-state temp
```

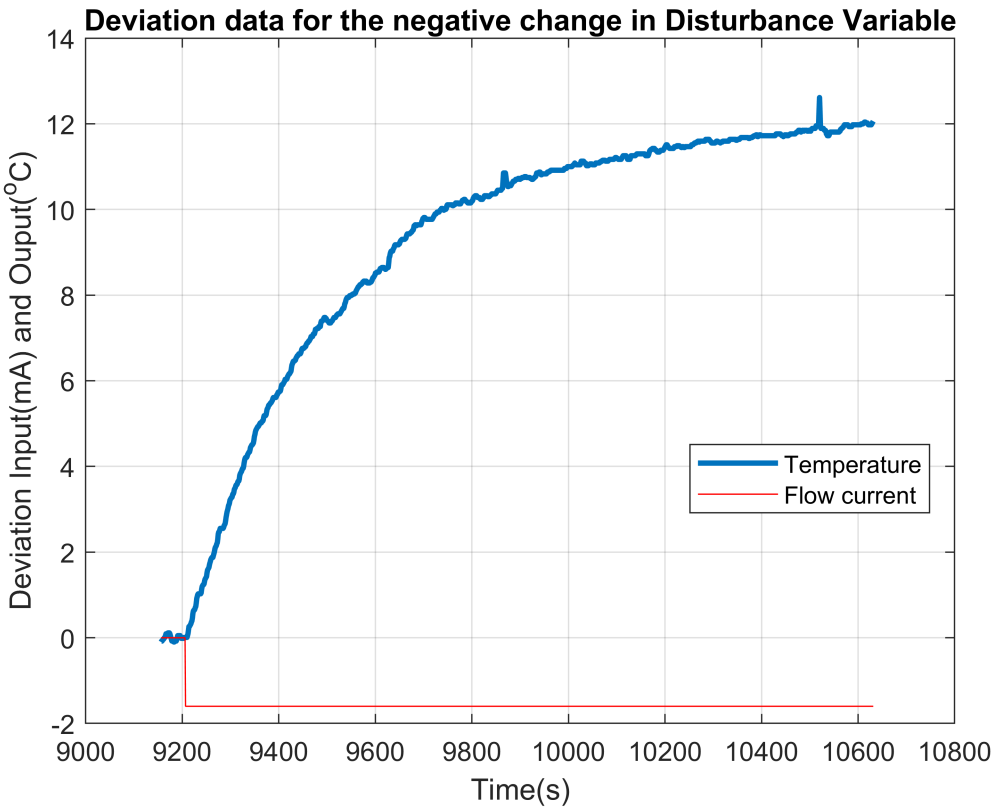
```
temp_SS_U_N_F = 56.1430
```

```
flow_SS_U_N_F = mean(flow(SS_U_N_F-50:SS_U_N_F-1)) %steady-state flow
```

```
flow_SS_U_N_F = 10.4000
```

PLOT for negative deviation in Disturbance

```
plot(time(SS_U4-50: SS_U_N_F-1),temp_filtered(SS_U4 - 50:SS_U_N_F-1)- temp_SS_U4,LineWidth=2)
hold on
plot(time(SS_U4-50: SS_U_N_F-1),flow(SS_U4-50:SS_U_N_F-1)- flow_SS_U4,"r")
hold off
title("Deviation data for the negative change in Disturbance Variable")
legend("Temperature","Flow current",Location="best" )
xlabel("Time(s)")
ylabel("Deviation Input(mA) and Ouput(^oC)")
grid
```



Method : Calculating K and tau

```
% Graphical
K_N_F = (temp_SS_U_N_F-temp_SS_U4)/(flow_SS_U_N_F-flow_SS_U4)
```

```
K_N_F = -7.4914
```

```
yc = 0.633*(temp_SS_U_N_F-temp_SS_U4);
```

```
if(yc<=0)
 n_c = find(temp_filtered - temp_SS_U4 - yc <= 0,1);
else
 n_c = find(temp_filtered-temp_SS_U4 - yc >= 0,1);
end
t_c = time(n_c);
% t(find(heater-heater_s,1))
% t(idx_s)
tau_N_F= t_c-time(SS_U4)
```

```
tau_N_F = 321.7030
```

```
% Optimization Method
%del_N_F_heat = heat(SS_U4:SS_U_N_F)-heat_SS_U_N_F;
del_N_F_temp = temp_filtered(SS_U4:SS_U_N_F)-temp_SS_U4;
t_NF = time(SS_U4:SS_U_N_F)-time(SS_U4);

f = @(x)sum(((x(1).*(-dU).*(1-exp(-t_NF/x(2)))) - del_N_F_temp).^2);

x0=[K_N_F,tau_N_F];
[x,~] = fminunc(f,x0);
```

```
Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.

<stopping criteria details>
```

```
K_N_F_opt = x(1)
```

```
K_N_F_opt = -7.4475
```

```
tau_N_F_opt = x(2)
```

```
tau_N_F_opt = 301.4070
```

The Final Steady State after resetting the Disturbance was not obtained due to shutting off of the air supply to valve due to external factors.

But an assumption based on prior steady state values is taken as.

```
SS_U5 = mean([temp_SS_U1,temp_SS_U4,temp_SS_U2,temp_SS_U3])
```

```
SS_U5 = 44.1196
```

```
```

Controller Response

```
K_p = 0.5*(K_P_H_opt+K_N_H_opt)
```

```
K_p = 1.3903
```

```
tau_p = 0.5*(tau_P_H_opt+tau_N_H_opt)
```

```
tau_p = 148.4777
```

```
K_d = 0.5*(K_P_F_opt+K_N_F_opt)
```

```
K_d = -5.3340
```

```
tau_d = 0.5*(tau_P_F_opt+tau_N_F_opt)
```

```
tau_d = 215.9037
```

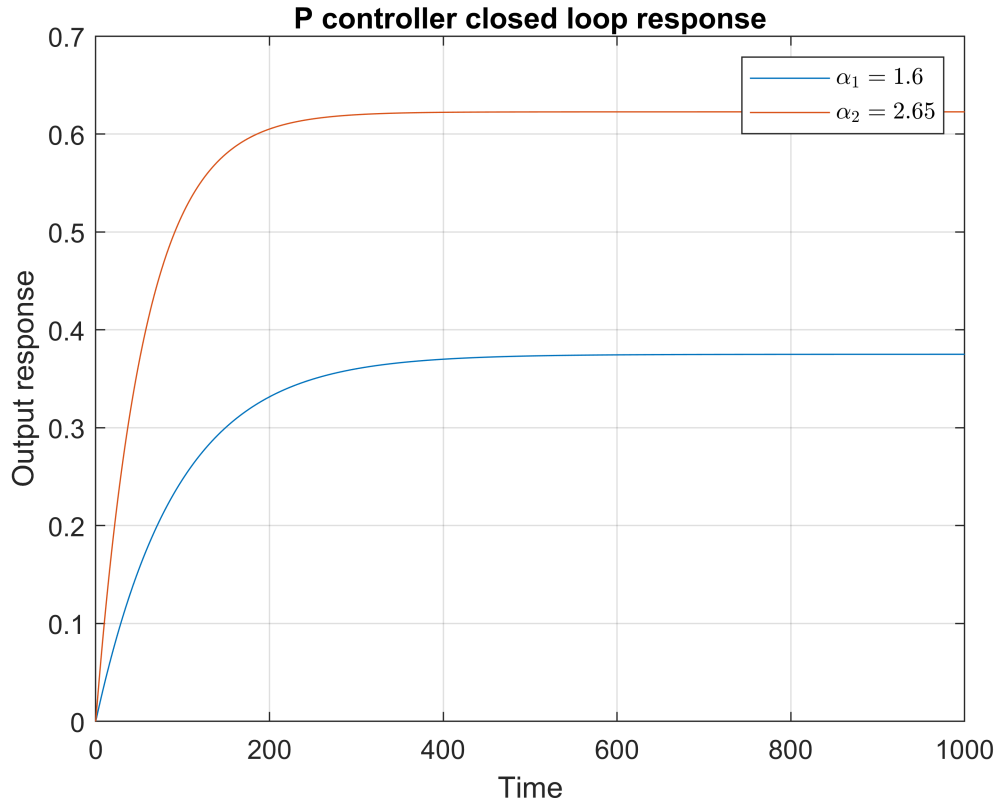
```
% TODO
alpha1 = 1.6
```

```
alpha1 = 1.6000
```

```
alpha2 = 2.65
```

```
alpha2 = 2.6500
```

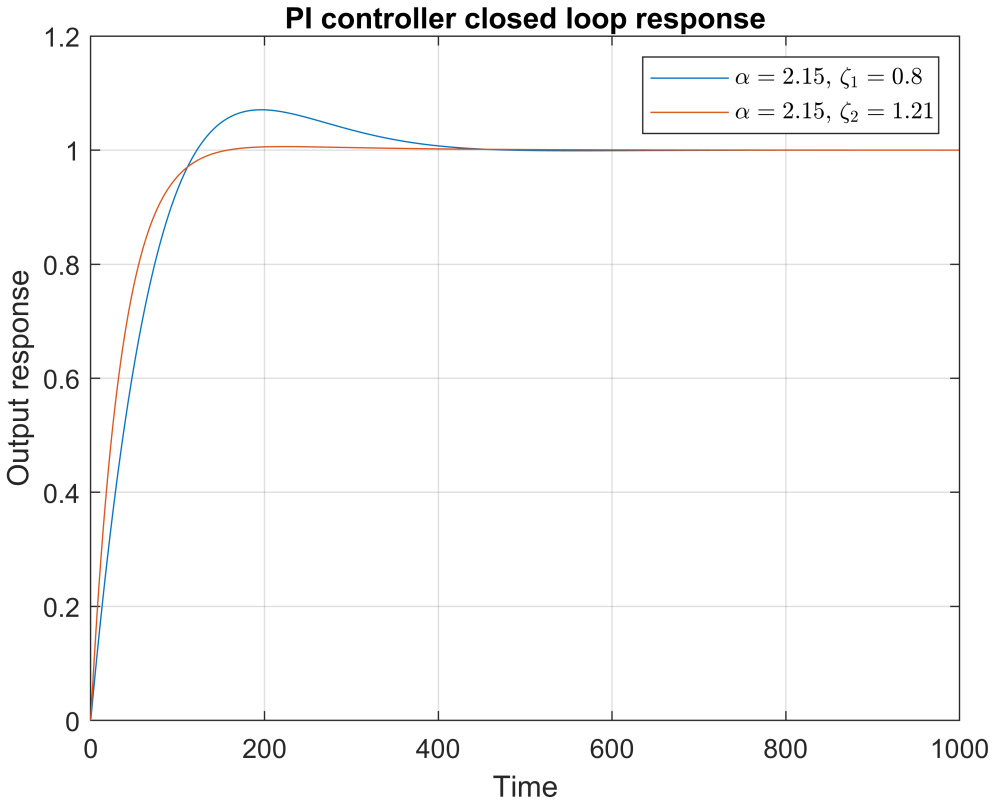
```
K_c1 = (alpha1-1)/K_p;
K_c2 = (alpha2-1)/K_p;


syms t s
H1 = K_c1*K_p*(1/s)/(tau_p*s+ 1+ K_c1*K_p);
H2 = K_c2*K_p*(1/s)/(tau_p*s+ 1+ K_c2*K_p);
h1=matlabFunction(ilaplace(H1));
h2=matlabFunction(ilaplace(H2));
t = 0:1.003:1000;

plot(t, h1(t))
hold on
plot(t,h2(t))
hold off
grid
title('P controller closed loop response')
leg1 = legend('$\alpha_1 = 1.6$', '$\alpha_2 = 2.65$');
set(leg1,'Interpreter','latex');
xlabel('Time')
ylabel('Output response')
```



PI Controller

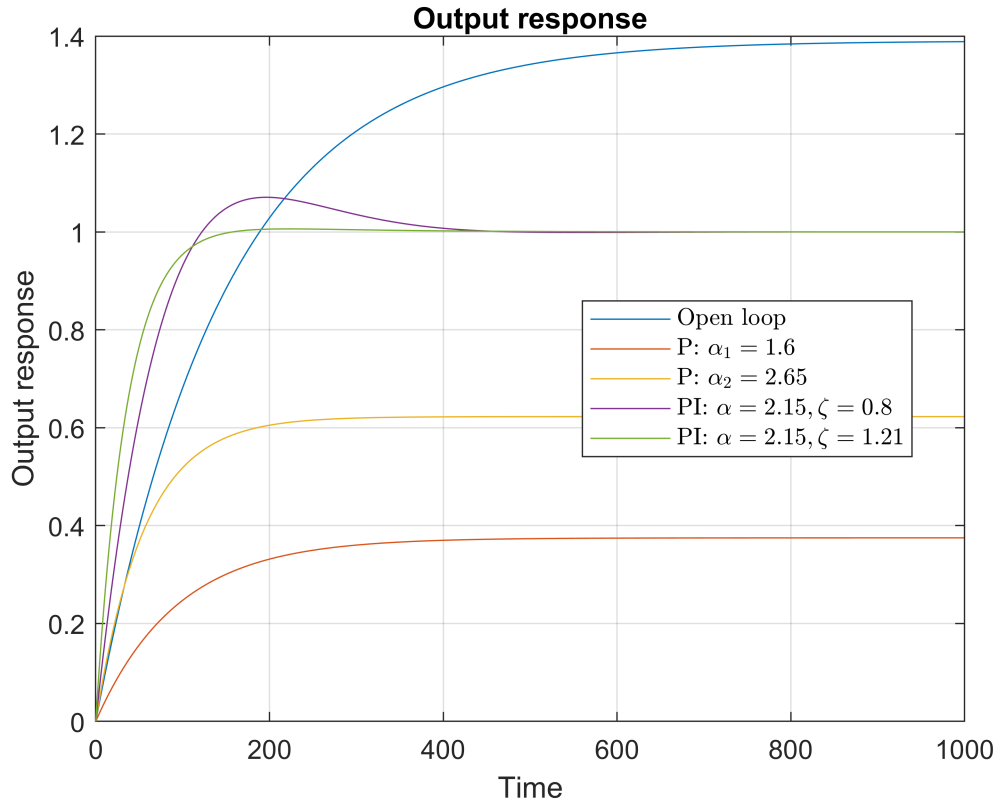
```
alpha_PI = 2.15;

%TODO
zeta_1 = 0.8; %(underdamped)
zeta_2 = 1.21; %(overdamped)

K_PI_c1 = (2*alpha_PI*zeta_1 - 1)/K_p;
K_PI_c2 = (2*alpha_PI*zeta_2 - 1)/K_p;

tau_i_1= (2*alpha_PI*zeta_1 - 1)*tau_p/(alpha_PI^2);
tau_i_2= (2*alpha_PI*zeta_2 - 1)*tau_p/(alpha_PI^2);

syms t3 s
t3 = 0:1.031:1000;
H_PI_1 = (tau_i_1*s + 1)*(1/s)/ ((tau_p*tau_i_1*s^2/(K_PI_c1*K_p))+(tau_i_1*s/(K_PI_c1*K_p/(1+K
H_PI_2 = (tau_i_2*s + 1)*(1/s)/ ((tau_p*tau_i_2*s^2/(K_PI_c2*K_p))+(tau_i_2*s/(K_PI_c2*K_p/(1+K
h1_PI=matlabFunction(ilaplace(H_PI_1));
h2_PI=matlabFunction(ilaplace(H_PI_2));
plot(t3,h1_PI(t3))
hold on
plot(t3,h2_PI(t3))
hold off
```

```
grid
title('PI controller closed loop response')
legend('$\alpha = 2.15$, $\zeta_1 = 0.8$', '$\alpha = 2.15$, $\zeta_2=1.21$','Interpreter','lat
xlabel('Time')
ylabel('Output response')
```



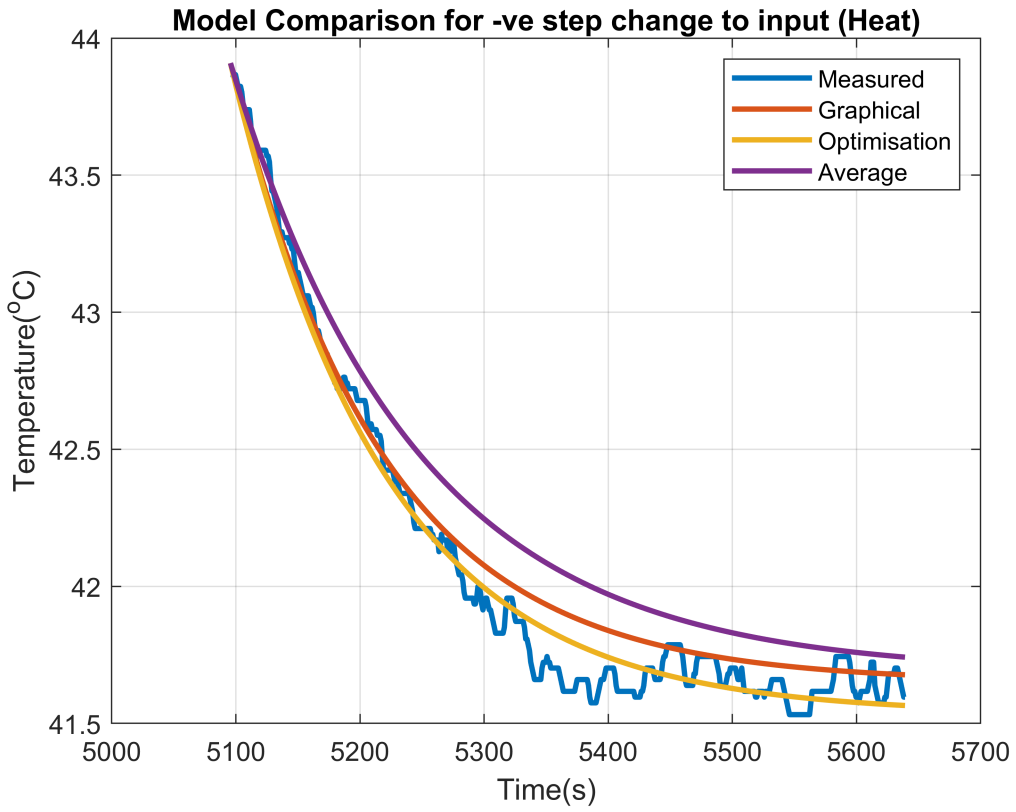
OPEN LOOP RESPONSES

```
syms t4 s
t4 = 0:1.031:1000;
hh_open = K_p*(1/s)/(tau_p*s + 1); %open loop response
h_open=matlabFunction(ilaplace(hh_open));
figure(3)
plot(t4,h_open(t4),t4,h1(t4),t4,h2(t4),t4,h1_PI(t4),t4,h2_PI(t4))
grid()
title('Output response')
legend('Open loop', 'P: $\alpha_1 = 1.6$', 'P: $\alpha_2 = 2.65$','PI: $\alpha = 2.15, \zeta =
xlabel('Time')
ylabel('Output response')
```

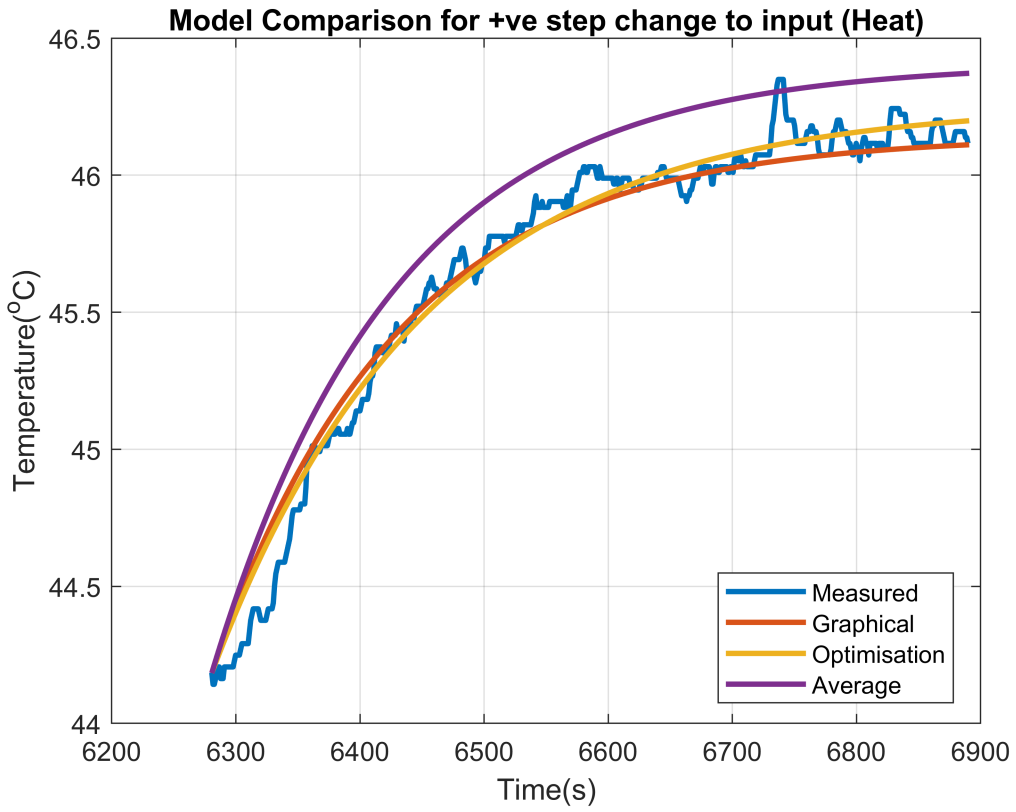


COMPARISON

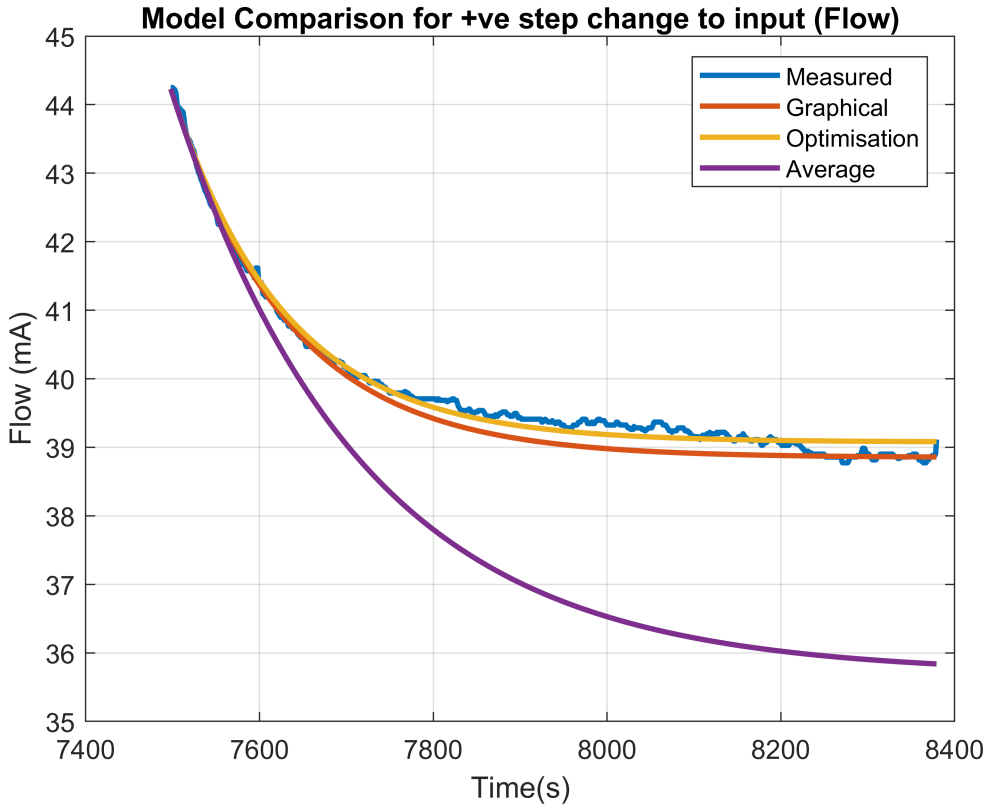
```
% For NEGATIVE MANIPULATED
graph =temp_SS_U1+ K_N_H.*(-dU).*(1-exp(-t_NH/tau_N_H));
opt = temp_SS_U1+K_N_H_opt.*(-dU).*(1-exp(-t_NH/tau_N_H_opt));
avg = temp_SS_U1+K_p.*(-dU).*(1-exp(-t_NH/tau_p));
t_plot = time(SS_U1:SS_U_N_H);
plot(t_plot,temp_filtered(SS_U1 :SS_U_N_H),LineWidth=2)
hold on
plot(t_plot,graph,LineWidth=2)
plot(t_plot,opt,LineWidth=2)
plot(t_plot,avg,LineWidth=2)
hold off
title("Model Comparison for -ve step change to input (Heat)")
ylabel("Temperature(^oC)")
xlabel("Time(s)")
legend("Measured","Graphical","Optimisation","Average")
grid
```



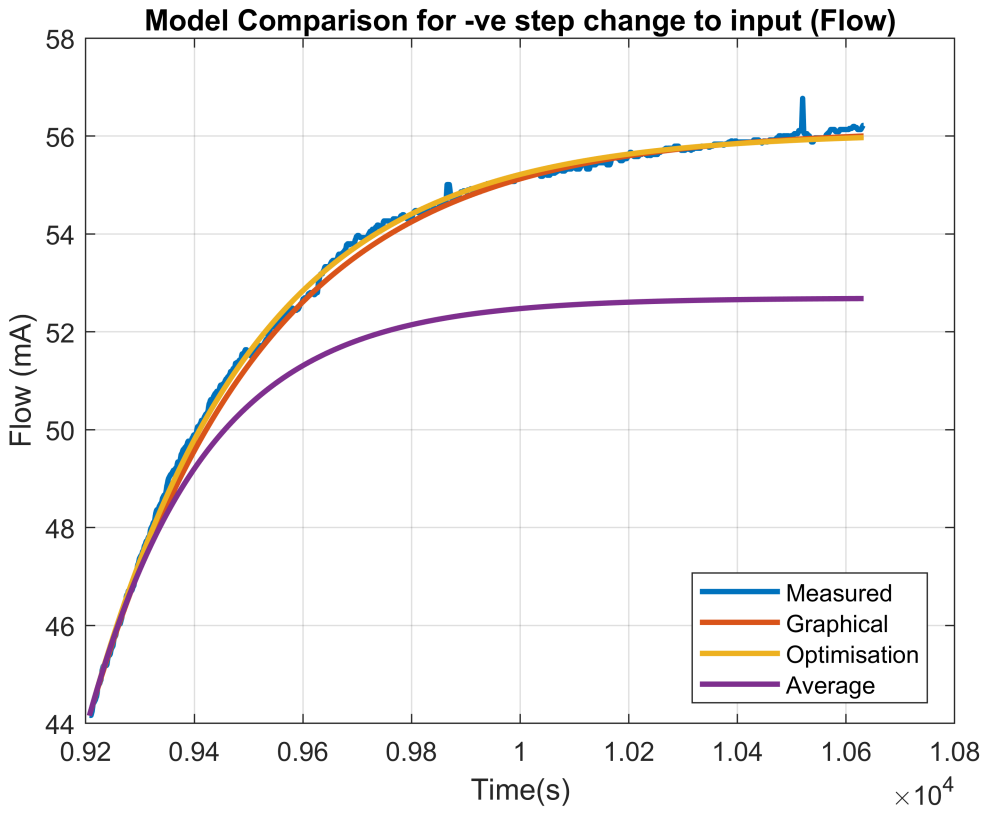

```
% For Positive MANIPULATED
graph =temp_SS_U2+ K_P_H.*(dU).*(1-exp(-t_PH/tau_P_H));
opt = temp_SS_U2+K_P_H_opt.*(dU).*(1-exp(-t_PH/tau_P_H_opt));
avg = temp_SS_U2+K_p.*(dU).*(1-exp(-t_PH/tau_p));
t_plot = time(SS_U2:SS_U_P_H);
plot(t_plot,temp_filtered(SS_U2 :SS_U_P_H),LineWidth=2)
hold on
plot(t_plot,graph,LineWidth=2)
plot(t_plot,opt,LineWidth=2)
plot(t_plot,avg,LineWidth=2)
hold off
title("Model Comparison for +ve step change to input (Heat)")
ylabel("Temperature(^oC)")
xlabel("Time(s)")
legend("Measured","Graphical","Optimisation","Average",Location="best")
grid
```




```
% For Positive Disturbance
graph =temp_SS_U3+ K_P_F.*(dU).*(1-exp(-t_PF/tau_P_F));
opt = temp_SS_U3+K_P_F_opt.*(dU).*(1-exp(-t_PF/tau_P_F_opt));
avg = temp_SS_U3+K_d.*(dU).*(1-exp(-t_PF/tau_d));
t_plot = time(SS_U3:SS_U_P_F);
plot(t_plot,temp_filtered(SS_U3 :SS_U_P_F),LineWidth=2)
hold on
plot(t_plot,graph,LineWidth=2)
plot(t_plot,opt,LineWidth=2)
plot(t_plot,avg,LineWidth=2)
hold off
title("Model Comparison for +ve step change to input (Flow)")
ylabel("Flow (mA)")
xlabel("Time(s)")
legend("Measured","Graphical","Optimisation","Average",Location="best")
grid
```




```
% For Negative Disturbance
graph =temp_SS_U4+ K_N_F.*(-dU).*(1-exp(-t_NF/tau_N_F));
opt = temp_SS_U4+K_N_F_opt.*(-dU).*(1-exp(-t_NF/tau_N_F_opt));
avg = temp_SS_U4+K_d.*(-dU).*(1-exp(-t_NF/tau_d));
t_plot = time(SS_U4:SS_U_N_F);
plot(t_plot,temp_filtered(SS_U4 :SS_U_N_F),LineWidth=2)
hold on
plot(t_plot,graph,LineWidth=2)
plot(t_plot,opt,LineWidth=2)
plot(t_plot,avg,LineWidth=2)
hold off
title("Model Comparison for -ve step change to input (Flow)")
ylabel("Flow (mA)")
xlabel("Time(s)")
legend("Measured","Graphical","Optimisation","Average",Location="best")
grid
```



https://github.com/MananMohnot/Chemical-Lab-4/tree/main/Lab%201