NTNU – Trondheim
Norwegian University of
Science and Technology

TKP4140 Process Control

Project E: Van de Vusse reaction in an isothermal CSTR

Part 1, Part 2 & Part 3

Group E-2

Tara Modarrespanah
Andrea Ritossa
Kira Trier Wang

November 18, 2022

Contents

List of Figures

List of Tables

1 Part 1: Modeling and Simulation

In this part we have modelled the system described in Project E, developed a nonlinear model for it and simulated it in Matlab and Simulink. In the end, we forced a single step increase for the input q and the three disturbances, in order to analyze the behaviour of our model and analyze the results.

1.1 Project description

Figure 21 presents a schematic description of an isotherm continuous stirred tank reactor (CSTR) with a feed stream q_1, which consists of pure component A, and a product stream q_2 consisting of A, B, C and D. Inside the reactor the following reactions occur:

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

$$2A \xrightarrow{k_3} D$$

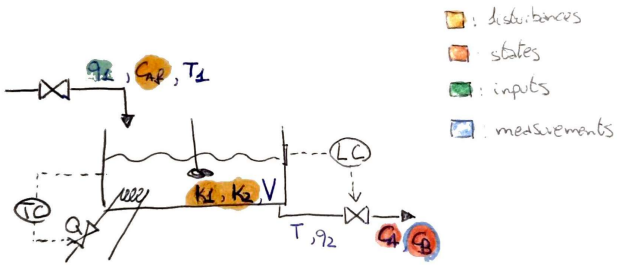

Figure 1: Process diagram for the system modelled

The rate of reaction of each component is:

$$r_A = -k_1 C_A - k_3 C_A^2$$

$$r_B = k_1 C_A - k_2 C_B$$

$$r_C = k_2 C_B$$

$$r_D = 0.5 k_3 C_A^2$$

1.1.1 Classification of variables

The variables in the model can be classified as:

- states (**x**): C_A, C_B
- inputs (**u**): q_1
- outputs (measurements) (**y**): C_B
- disturbances (**d**): C_{Af}, k_1, k_2

The concentrations of C_C and C_D are derived from the states by algebraic manipulation.

1.2 Nonlinear model

We derived the nonlinear model, based on the lessons of Process Control (1).

1.2.1 Assumptions

In order to get the non-linear model we considered the following quantities constant:

- Temperature: The temperature is assumed to be controlled by another, already installed controlled and this can therefore be assumed to be constant. Thanks to the constant temperature we can conclude that the rates of reaction are constant too, based on the Arrhenius equation: $k = e^{\frac{-E}{RT}}$
- Density: As we are considering liquid with a constant temperature, the density can be assumed to be constant.
- Volume: As the system has a level controller implemented, the volume can be considered constant.
- Perfect mixing

1.2.2 Model

For modeling this system there is no need for an energy balance because we assume the temperature to be constant. This implies that the variation of the temperature in time is zero. Therefore, the entire **energy balance** equals zero.

Mass balance: The general mass balance for the system looks like this:

$$\frac{d(m)}{dt} = m_{in} - m_{out}$$

Considering the constant density and implementing m = ρV and q = $\frac{\mathrm{m}}{\rho}$.
The mass balance can be shortened to become:

$$\frac{d(V)}{dt} = q_1 - q_2$$

As the volume in this system is constant, and hence $\frac{\mathrm{dV}}{\mathrm{dt}} = 0$ the mass balance can be simplified further to

$$q_1 = q_2 \tag{1}$$

Component balance:
The general component balance is:

$$\frac{dn_i}{dt} = F_{(i,in)} - F_{(i,out)} + r_i V$$

with i $\in$ [A, B, C, D]

The specific component balances for the four components in the system is:

$$A : \frac{d(c_A V)}{dt} = c_{Af} q_1 - c_A q_2 - V c_A (k_1 + c_A k_3)[\frac{mol}{min}] \tag{2a}$$

$$B : \frac{d(c_B V)}{dt} = -c_B q_2 + k_1 V c_A - k_2 V c_B [\frac{mol}{min}] \tag{2b}$$

$$C : \frac{d(c_C V)}{dt} = -c_C q_2 + k_2 V c_B [\frac{mol}{min}] \tag{2c}$$

$$D : \frac{d(c_D V)}{dt} = -c_D q_2 + \frac{k_3}{2} V c_A^2 [\frac{mol}{min}] \tag{2d}$$

1.2.3 Steady state data

In order to simulate the system, steady-state data is needed for all variables. In table 3 below, the steady state data from the project description can be found. In table 2 the steady-state data calculated based on the steady-state component balances can be seen.

Table 1: Steady state data

Variable	**Description**	**Steady state value**	**Units**
V	Volume	10000	L
k_1^*	Rate of reaction 1	25/6	1/min
k_2^*	Rate of reaction 2	5/3	1/min
k_3^*	Rate of reaction 3	1/6	L/(min mol)
q_1^*	Flow	5000	L/min
C_{Af}^*	Concentration of A in the feed	10	$\mathrm{mol\,L^{-1}}$

Table 2: Calculated steady-state data

Variable	**Description**	**Steady state value**	**Units**
C_A^*	Concentration of component A	1.0333	$\mathrm{mol\,L^{-1}}$
C_B^*	Concentration of component B	1.9871	$\mathrm{mol\,L^{-1}}$
C_C^*	Concentration of component C	6.6237	$\mathrm{mol\,L^{-1}}$
C_D^*	Concentration of component D	0.1780	$\mathrm{mol\,L^{-1}}$

1.3 Simulation results

We simulated the nonlinear model in Simulink. We applied a step change in each of the input variables and disturbances, with an increase of 10% from the nominal values *. The following graphs were obtained:

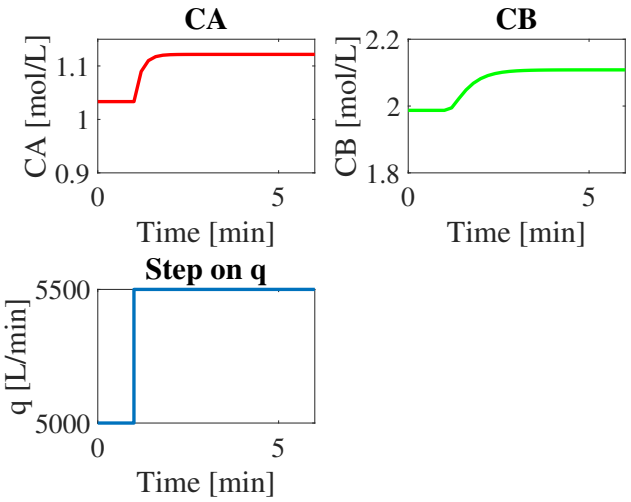

Figure 2: Simulation of step in q

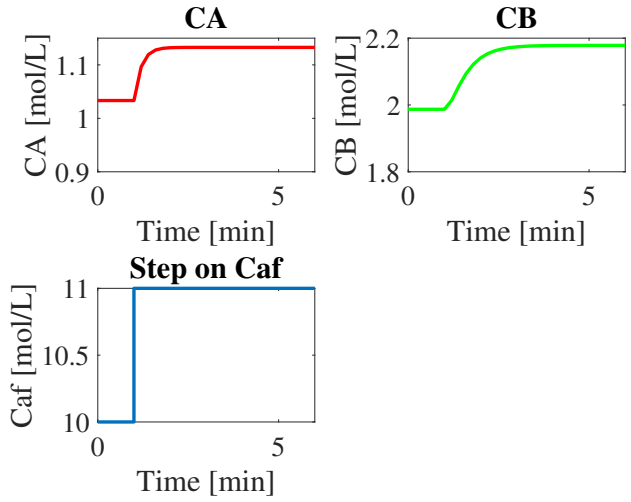

Figure 3: Simulation of step in Caf

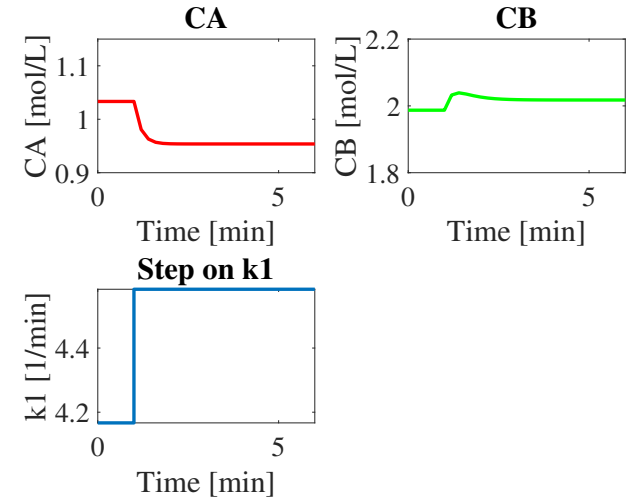

Figure 4: Simulation of step in k1

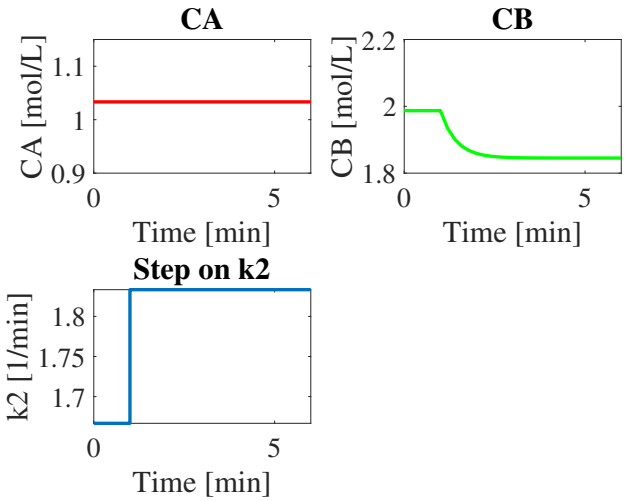

Figure 5: Simulation of step in k2

1.4 Steady state gains

We calculated the corresponding steady-state gains for our output C_b for a 10% step response for the input and each disturbance in the system. As all the simulations reach a steady state, finding the slope is in this case not necessary.

Table 3: Corresponding steady-state gain and slope

Value	Steady state gain	Units
q	0.000242	$\frac{L^2}{min \cdot mol}$
C_{Af}	0.191	—
k_1	0.0732	$\frac{L}{min \cdot mol}$
k_2	-0.855	$\frac{L}{min \cdot mol}$

1.5 Discussion

1.5.1 Making a step change on q_1

As we know the volume of the reactor is constant so increasing the entrance flow rate is equal to raising the exit flow rate as well. When we put a positive step on q_1, it means that the amount of A entering the reactor has much less time to remain in the reactor(residence time). Therefore, the reactant A doesn't have enough time to react as it used to. This leads to more concentration of A in the reactor. Due to the less residence time, both reactions 1 and 2 are going to be done less but reaction 2 is more affected and causing C_B to increase. This can be because of the fact that meanwhile C_A is increasing and lets B to produce more. These can also be observed by the equations for C_A and C_B (2a).

Here, in the first seconds after putting a step change for q_1, we can assume that C_A is almost equal to its previous amount so that the term $q(c_{Af} - c_A)$ is becoming bigger which can lead the derivative $\frac{d(c_A V)}{dt}$ to increase and the result would be C_A's rising. The growth in C_A would counteract the effect of q_1 and C_A will reach a new steady state point. This is because A will be consumed to produce B and D and some of it will exit the reactor without any reaction(2b).

For B, after we put a step on q_1 the C_A is increasing. The growth in the number of moles

of B that are created ($k_1 V c_A$) defeats the increase of the number of moles of B exiting the reactor ($-c_B q$). This can be because of the large volume of the reactor. This would lead to a growth in C_B which would have an effect on the number of moles of B exiting the reactor and reacting to create C. In this case, growth in the C_B would counteract the step on q_1. As a result, C_B would increase until a new steady state point is reached. We can also observe that C_B will increase with some delay (due to waiting for C_A changes) and a smaller slope (because of the term of $-c_B q$).

1.5.2 Making a step change on C_{Af}

Here, when we increase the concentration of A in the feed, we have more A in the reactor to react and produce B; therefore, the concentration of B will increase until it reaches a steady state again. If we have a more accurate look at the equations, we can conclude that $\frac{d(c_A V)}{dt}$ will increase because of the gain in $q c_{Af}$ term. After that growth in C_A will lead to a greater number of moles of A exiting the reactor and reacting to produce B and D. This would counteract the step change of C_{Af} and after some time C_A will reach new steady state condition.

After some time that the C_A changes from its previous steady state amount, C_A's increase would cause a positive slope for C_B per time due to the $k_1 V c_A$ term. Again, C_B's growth is equal to the greater number of B exiting the reactor and reacting to produce C which can counteract the effect of C_{Af} increase after some time and therefore C_B will reach a new steady state.

1.5.3 Making a step change on k_1

Putting a step on k_1 is equal to a greater number of moles of A converted to B so that C_A would decrease. This can also be tracked by 2a, where increasing $k_1(-V c_A k_1)$ leads decreasing in $\frac{d(c_A V)}{dt}$. Regarding a reduction in the amount of C_A, a smaller number of moles of A will leave the reactor and are converted to B and D. This will create a balance and after some time C_A will stick out to a new steady state amount.

In the first seconds after putting on the step, C_B will increase due to the more converting of A to B ($k_1 V c_A$). After that due to the a greater amount of B exiting the reactor ($-c_B q$) and converting to C ($-k_2 V c_B$) and a decrease in C_A ($k_1 V c_A$) which counteracts the effect of k_1, it will reach a new steady state point. C_B will experience a peak and it has to do with the change in C_A (before C_A decreases a lot).

1.5.4 Making a step change on k_2

Putting a step on k_2 has nothing to do with C_A, because it is for the second reaction that converts B to C and their concentrations does not have any effect on C_A as seen in equation (2a). In this case, C_A remains constant. For B, step on k_2 is equal to a greater number of moles of B converting to C, therefore C_B will decrease. It also can be concluded regarding (2b) that, in the first seconds, increasing k_2 ($-k_2 V c_B$) will cause a negative slope for C_B versus time. After decreasing C_B is started, a smaller number of moles of B will exit the reactor and even less would react to produce C. After all, a decrease in C_B will counteract the increase in k_2 and C_B will reach a new steady state amount.

2 Part 2: System Analysis

In this section we used the mathematical instruments that had been taught to us in TKP4140 lessons to improve our understanding of the system characteristics. This enabled us to see the differences between the nonlinear model and our approximation.In this part, the following inputs, output, disturbances and states used:

$$x = [C_A \;\; C_B]^T$$
$$u = [q]^T$$
$$d = [C_{Af} \;\; k_1 \;\; k_2]^T$$
$$y = [C_B]^T$$

2.1 Linearization

2.1.1 Linearization of C_A

$$\frac{dC_A}{dt} = \frac{C_{Af}q}{V} - \frac{C_Aq}{V} - C_Ak_1 - C_A^2k_3 \tag{3}$$

Introducing the deviation variables leads to

$$\frac{dC_A}{dt} = \frac{q^*}{V}\Delta C_{Af} + (-\frac{q^*}{V} - k_1^* - 2C_A^*k_3^*)\Delta C_A \tag{4}$$

From the linearization, the following parameters can be defined

$$a = \frac{q^*}{V}$$
$$b = \frac{-q^*}{V} - k_1^* - 2C_A^*k_3^*$$
$$c = \frac{C_Af^*}{V} - \frac{C_A^*}{V}$$
$$d = -C_A^*$$

The differential equation for the state C_A can then be simplified as

$$\frac{dC_A}{dt} = a\Delta C_{Af} + b\Delta C_A + c\Delta q + d\Delta k_1 \tag{5}$$

2.1.2 Linearization of C_B

$$\frac{dC_B}{dt} = \frac{-C_Bq}{V} + k_1C_A - k_2C_B \tag{6}$$

Deviation leads to

$$\frac{dC_B}{dt} = k_1\Delta C_A + (-\frac{q^*}{V} - k_2^*)\Delta C_B + (\frac{-C_B^*}{V})\Delta q + C_A^*\Delta k_1 + (-C_B^*)\Delta k_2 \tag{7}$$

From the linearizations, the following parameters are defined

$$e = k_1^*$$
$$f = \frac{-q^*}{V} - k_2^*$$
$$g = \frac{C_B^*}{V}$$
$$h = C_A^*$$
$$I = C_B^*$$

The differential equation for the state C_A can then be simplified as

$$\frac{dC_B}{dt} = e\Delta C_A + f\Delta C_B + g\Delta q + h\Delta k_1 + I\Delta k_2 \tag{8}$$

The calculated values for the parameters can be found in table 4

Parameter	Value	Unit
a	0.5	min^{-1}
b	-5.0111	min^{-1}
c	$8.9667 \cdot 10^{-4}$	$\frac{\text{mol}}{\text{L}^2}$
d	-1.0333	$\frac{\text{mol}}{\text{L}}$
e	$\frac{25}{6}$	min^{-1}
f	-2.16	min^{-1}
g	$-1.9871 \cdot 10^{-4}$	$\frac{\text{mol}}{\text{L}^2}$
h	1.0333	$\frac{\text{mol}}{\text{L}}$
I	-1.9871	$\frac{\text{mol}}{\text{L}}$

Table 4: Calculated parameters for the differential equation system

The standard state-space form for the system is generally written as:

$$\dot{x} = Ax + Bu + Ed \tag{9a}$$
$$y = Cx + Du \tag{9b}$$

This can in our case be rewritten to

$$\dot{X} = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} b & 0 \\ e & f \end{bmatrix} \begin{bmatrix} C_A \\ C_B \end{bmatrix} + \begin{bmatrix} c \\ g \end{bmatrix} q + \begin{bmatrix} a & d & 0 \\ 0 & h & I \end{bmatrix} \begin{bmatrix} C_{Af} \\ k_1 \\ k_2 \end{bmatrix} \tag{10a}$$

$$y = \begin{bmatrix} C_A \\ C_B \end{bmatrix} \tag{10b}$$

Filling in the numbers earlier calculated, we get:

$$\dot{X} = \begin{bmatrix} -5.0111 & 0 \\ 4.16 & -2.16 \end{bmatrix} \begin{bmatrix} C_A \\ C_B \end{bmatrix} + \begin{bmatrix} 8.9667 \cdot 10^{-4} \\ 1.9871 \cdot 10^{-4} \end{bmatrix} q + \begin{bmatrix} 0.5 & -1.03 & 0 \\ 0 & 1.03 & -1.987 \end{bmatrix} \begin{bmatrix} C_{Af} \\ k_1 \\ k_2 \end{bmatrix} \tag{11a}$$

$$y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} C_A \\ C_B \end{bmatrix} \tag{11b}$$

The transfer function $G(s)$ can then be constructed:

$$G(s) = C(sI - A)^{-1}B + D \tag{12}$$

Inserting the matrices found previously, we get

$$G(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \left(s \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} -5.01 & 0 \\ 4.16 & -2.16 \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} 8.97 \cdot 10^{-4} \\ -1.99 \cdot 10^{-4} \end{bmatrix} \tag{13}$$

$$G(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s + 5.01 & 0 \\ -4.16 & s + 2.16 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 8.97 \cdot 10^{-4} \\ -1.99 \cdot 10^{-4} \end{bmatrix} \tag{14}$$

$$G(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \frac{1}{(s + 5.01)(s + 2.16)} \cdot \begin{bmatrix} s + 2.16 & 0 \\ -4.16 & s + 5.01 \end{bmatrix} \cdot \begin{bmatrix} 8.97 \cdot 10^{-4} \\ -1.99 \cdot 10^{-4} \end{bmatrix} \tag{15}$$

$$G(s) = \begin{bmatrix} \frac{(s+2.16) \cdot 8.97 \cdot 10^{-4}}{(s+5.01)(s+2.16)} \\ \frac{4.16 \cdot 8.977 \cdot 10^{-4} - 1.99(s+5.01)}{(s+5.01)(s+2.16)} \end{bmatrix} \tag{16}$$

$$G(s) = \begin{bmatrix} \frac{1.79 \cdot 10^{-4}}{(0.1996s+1)} \\ \frac{2.524 \cdot 10^{-4}(1-0.0726s)}{(0.1996s+1)(0.4615s+1)} \end{bmatrix} \tag{17}$$

The transfer function for the disturbance can as well be found:

$$G_d(s) = C(sI - A)^{-1}E \tag{18}$$

With the matrices earlier found, we get

$$G_d(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \left(s \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} -5.01 & 0 \\ 4.16 & -2.16 \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} 0.5 & -1.03 & 0 \\ 0 & 1.03 & -1.987 \end{bmatrix} \tag{19}$$

$$G_d(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s+5.01 & 0 \\ -4.16 & s+2.16 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 0.5 & -1.03 & 0 \\ 0 & 1.03 & -1.987 \end{bmatrix} \tag{20}$$

$$G_d(s) = \begin{bmatrix} s+2.16 & 0 \\ -4.16 & s+5.01 \end{bmatrix} \cdot \frac{1}{(s+5.01)(s+2.16)} \cdot \begin{bmatrix} 0.5 & -1.03 & 0 \\ 0 & 1.03 & -1.987 \end{bmatrix} \tag{21}$$

$$G_d(s) = \begin{bmatrix} \frac{0.5 \cdot (s+2.16)}{(s+5.01)(s+2.16)} & \frac{-1.03(s+2.16)}{(s+5.01)(s+2.16)} & 0 \\ \frac{0.5 \cdot 4.16}{(s+5.01)(s+2.16)} & \frac{-1.03 \cdot 4.16 + 1.03(s+5.01)}{(s+5.01)(s+2.16)} & \frac{-1.987(s+5.01)}{(s+5.01)(s+2.16)} \end{bmatrix} \tag{22}$$

$$G_d(s) = \begin{bmatrix} \frac{0.0998}{0.1996s+1} & \frac{-0.2056}{0.1996s+1} & 0 \\ \frac{0.19192}{(0.1996s+1)(0.4615s+1)} & \frac{0.08(1+1.18s)}{(0.1996s+1)(0.4615s+1)} & \frac{-0.917}{0.4615s+1} \end{bmatrix} \tag{23}$$

2.1.3 Zeros, poles and gains

The poles of the system can be calculated as the eigenvalues of the matrix A.

$$A = \begin{bmatrix} -5.01 & 0 \\ 4.16 & -2.16 \end{bmatrix} \tag{24}$$

The characteristic polynomial can be found as the determinant of $(\lambda I - A)$:

$$det\left([\lambda I - A]\right) = (\lambda + 5.01)(\lambda + 2.16) \tag{25}$$

This can be put equal to zero and the poles are determined to be

$$(\lambda + 5.01)(\lambda + 2.16) = 0 \qquad \leftrightarrow \qquad \lambda = -5.01 \quad \& \quad \lambda = -2.16 \tag{26}$$

As well, the poles and zeros were found from the transfer functions, using the ones on zero/pole form (equation (16) & (22)) for the poles and zeros and using the time-constant form to find the gain (equation (17) & (23)). The results can be seen in table 5 below:

g	**Zeros**	**Poles**	**Gain**
g_1	———-	-5.01 s^{-1}	$1.79 \cdot 10^{-4} \frac{L^2}{min \cdot mol}$
g_2	13.76	-5.01 s^{-1} & -2.16 s^{-1}	$2.524 \cdot 10^{-4} \frac{L^2}{min \cdot mol}$
$g_{d,11}$	———	-5.01 s^{-1}	0.0998
$g_{d,21}$	———-	-5.01 s^{-1} & -2.16 s^{-1}	0.19192
$g_{d,12}$	——	-5.01 s^{-1}	$-0.2056 \frac{L}{min \cdot mol}$
$g_{d,22}$	-0.85	-5.01 s^{-1} & -2.16 s^{-1}	$0.08 \frac{L}{min \cdot mol}$
$g_{d,13}$	———-	———-	———-
$g_{d,23}$	———-	-2.16 s^{-1}	-0.917 $\frac{L}{min \cdot mol}$

Table 5: Values for zeros, gain and poles for each transfer function

2.2 Simulations of the system

The system was simulated using Matlab and Simulink. A step change of 10% was applied to all disturbances and the input, and the non-linear system was plotted against the linearized system. The graphs obtained can be found in figure 6 - 9 below.

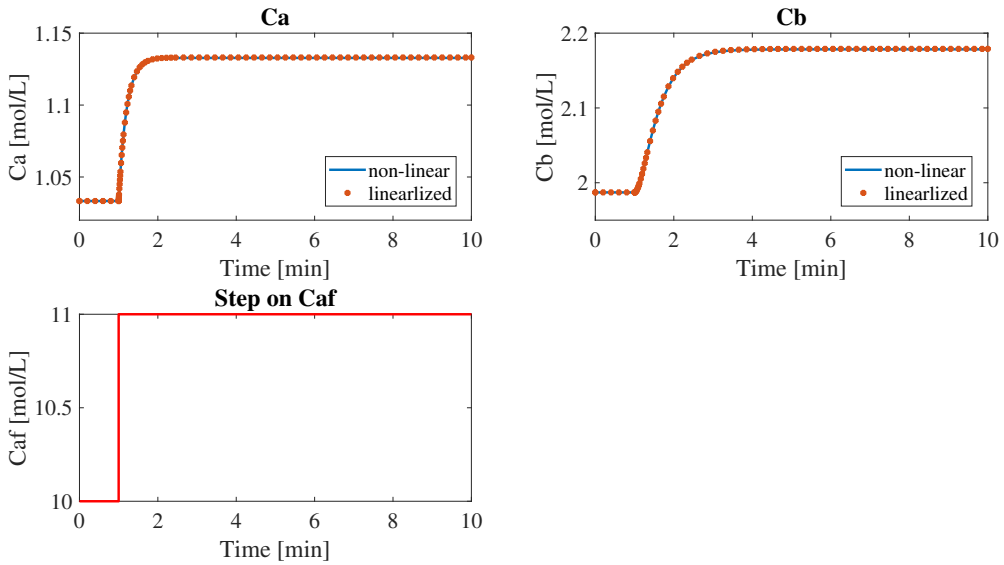

Figure 6: Simulation of a 10% step in C_{Af}

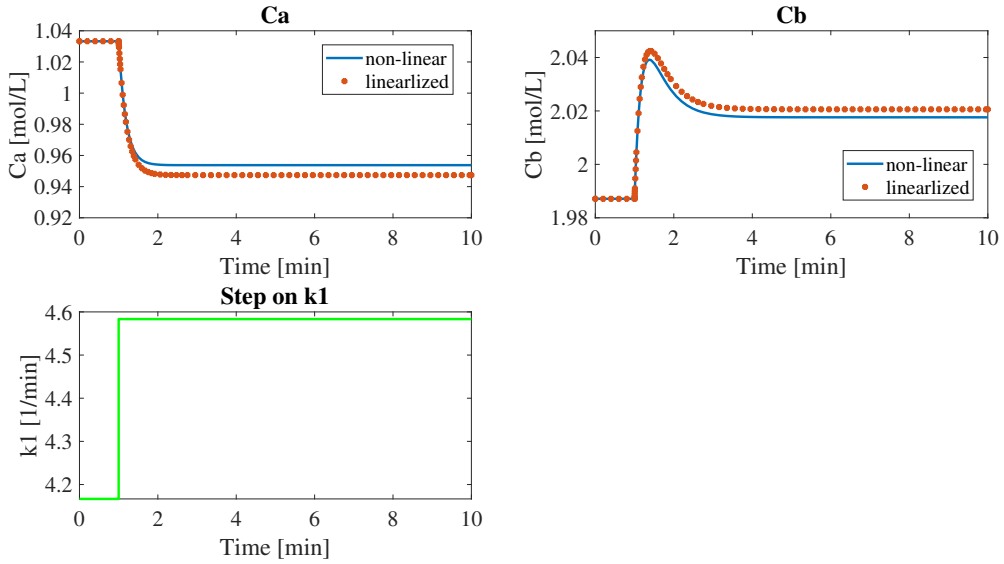

Figure 7: Simulation of a 10% step in k_1

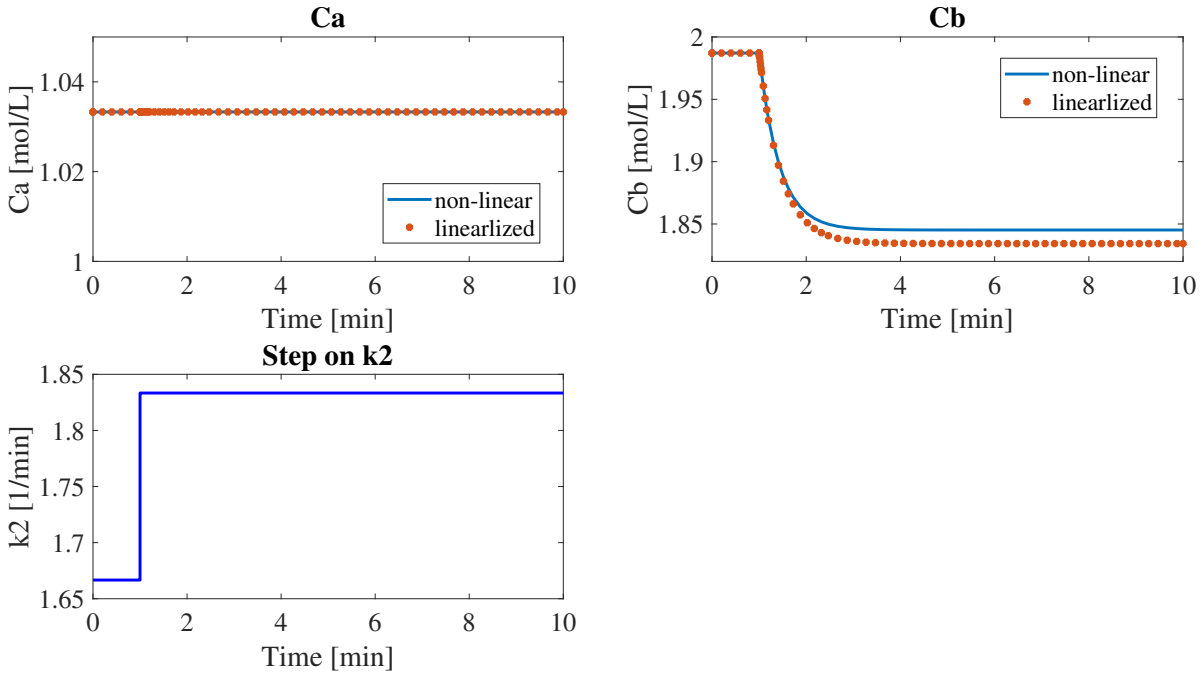

Figure 8: Simulation of a 10% step in k_2

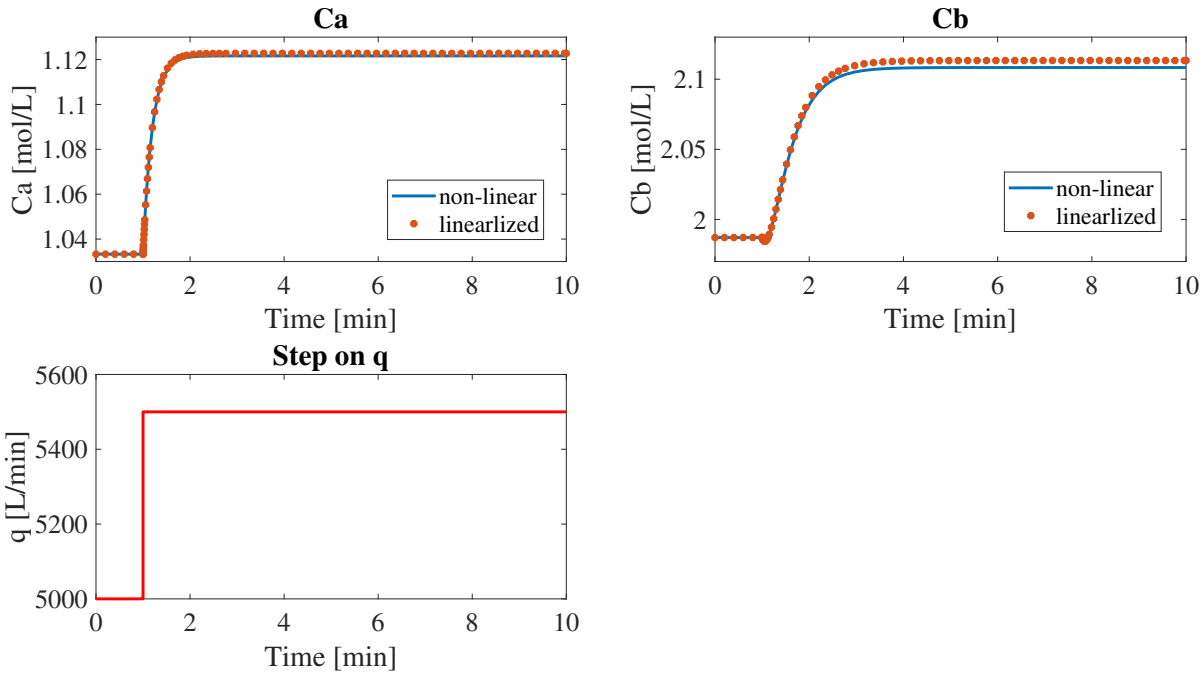

Figure 9: Simulation of a 10% step in q

From figure 6 it is seen that the linearized and the non-linearized model gives the same response. This means that a step change in C_{Af} does not affect the linearized model in a different way than it affects the non-linear model and hence that the model for C_{Af} was linear from the beginning.

Looking at the step change in k_1 in figure 7 it is seen that C_A reaches a steady state that is a bit smaller for the linearized model than for the non-linearized model. As well, it is seen that C_B in the same way reaches a slightly higher steady state. This suggests that the states C_A and C_B in this case are dependent on a change in k_1, which as well can be seen in equation (5) and (8).

In figure 8 it is seen that only the linearized model for C_B varies from the non-linear model, whilst the linearized and non-linearized models for C_A are unchanged. It is seen how the linearized curve for C_B follows the non-linearized curve for the first approx. 2 minutes. After this, the non-linear model reaches a steady state while the linear curve falls a bit more before reaching a new, lower steady state.

The effect of a 10% step change in q can be seen in figure 9. Here, it is seen how the curve for C_A is following the linearized curve whilst the curve for C_B varies very little. This suggests that the linearized model for C_B is dependent on the input, but that the effect is very little, compared to the change in the non-linear model.

2.3 Half-rule approximation

In order to tune PI controllers for controlling the system we need to have a first order plus delay model. So we can approximate the transfer functions in G(17) using the half rule:

$$G(s) = \begin{bmatrix} \frac{1.79 \cdot 10^{-4}}{(0.1996s+1)} \\ \\ \frac{2.524 \cdot 10^{-4}(1-0.0726s)}{(0.1996s+1)(0.4615s+1)} \end{bmatrix}$$

- The transfer function of Ca is already linear, so we will keep it as:

$$G_{Ca}(s) = 1.79 \cdot 10^{-4} \frac{1}{0.1996s + 1} \tag{27}$$

- The transfer function of C_b, instead, doesn't fit the desired shape, so we will apply the half rule to it:

$$k = 2.524 * 10^{-}4 \quad \tau_1 = 0.4615 \quad \tau_2 = 0.1996 \quad T_1 = -0.0726$$

First order half rule approximation:

$$k' = k \quad \tau' = \tau_1 + \frac{\tau_2}{2} = 0.5163 \quad \theta' = \frac{\tau_2}{2} - T_1 = 0.1724$$

$$G_{Cb}(s) = 2.524 \cdot 10^{-4} \frac{e^{-0.1724s}}{0.5163s + 1} \tag{28}$$

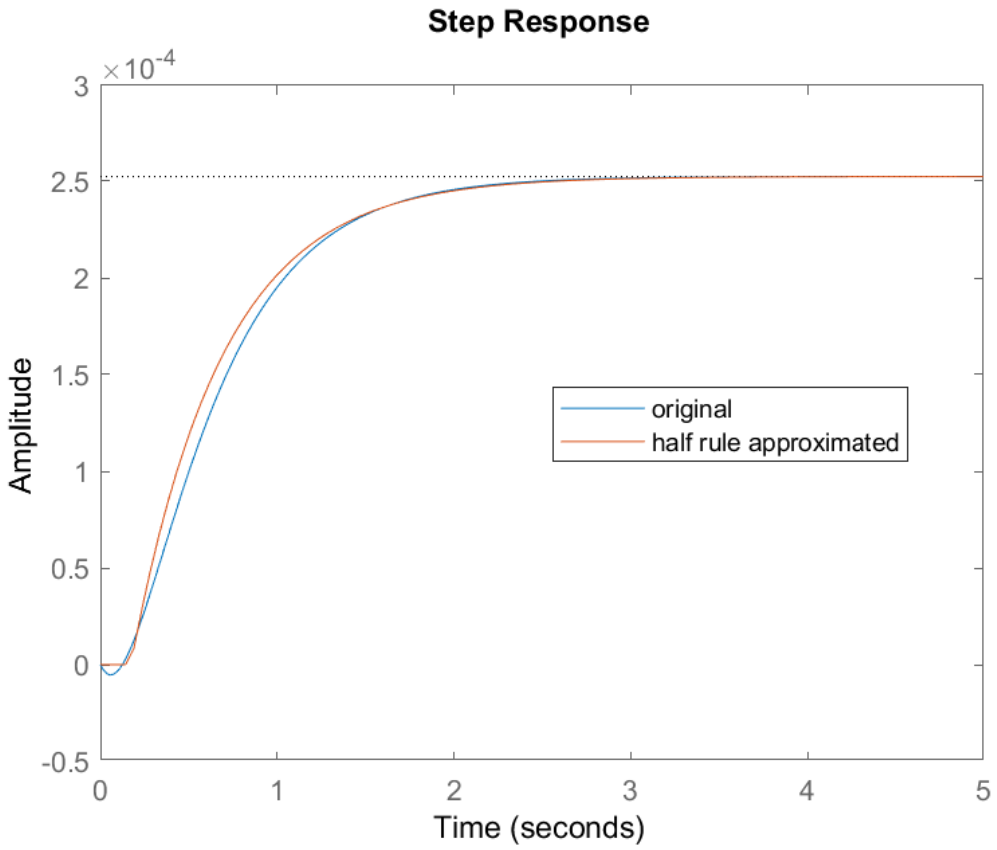

Figure 10: Approximation of G_{C_b}

To sum up the final values that we are going to use:

	k	τ_1	θ
G_{C_a}	$1.79 \cdot 10^{-4}$	0.1996	0
G_{C_b}	$2.524 \cdot 10^{-4}$	0.5163	-0.1724

As an exercise we have approximated the transfer function of the disturbances, for trying the LHP-zero approximation, but for clarity we will not add our results in this report.

2.4 Discussion

2.4.1 Properties of the transfer functions

The properties of the transfer functions such as: zeros, poles, and gain (RHP, LHP, time constants τ, time delays θ, etc.). Do these properties correspond to the simulations?
We make a comparison one by one for each situation and its plot:

Step on input (q):

- Results for C_A: We just have one pole which is real and negative and it means the system is stable and without any oscillation. The time constant is 0.1996 seconds. We can see from the graph that the time that C_A reaches 0.63 of its changes is so similar to this amount. The gain we calculated is $1.79 \cdot 10^{-4}$ which is almost equal to the amount we derive from the graph ($0.08/500 = 1.6 \cdot 10^{-4}$).

- Results for C_B: We have two poles which are real and negative and it means the system is stable and without any oscillation. The time constants are 0.1996 and 0.4615 seconds which if we use a half rule for first-order approximation would be 0.5613 seconds. We can see from the graph that the time that C_B reaches 0.63 of its changes is so similar to this amount. The gain we calculated is $2.524 \cdot 10^{-4}$ which is almost equal to the amount we derive from the graph (approximately $0.1129/500 = 2.258 \cdot 10^{-4}$). The transfer function also has a zero equal to 13.76 so the response is moderate in speed as it is shown in the graph.

Step on disturbance ($C_A f$):

- Results for C_A: We just have one pole which is real and negative and it means the system is stable and without any oscillation. The time constant is 0.1996 seconds. We can see from the graph that the time that C_A reaches 0.63 of its changes is so similar to this amount. The gain we calculated is 0.0998 which is almost equal to the amount we derive from the graph (approximately $0.1/1 = 0.1$).

- Results for C_B: We have two poles which are real and negative and it means the system is stable and without any oscillation. The time constants are 0.1996 and 0.4615 seconds which if we use a half rule for first-order approximation would be 0.5613 seconds. We can see from the graph that the time that C_B reaches 0.63 of its changes is so similar to this amount. The gain we calculated is 0.19192 which is almost equal to the amount we derive from the graph (approximately 0.191).

Step on disturbance (k_1):

- Results for C_A: We just have one pole which is real and negative and it means the system is stable and without any oscillation. The time constant is 0.1996 seconds. We can see from the graph that the time that C_A reaches 0.63 of its changes is so similar to this amount. The gain we calculated is -0.2056 which is almost equal to the amount we derive from the graph ($-0.085/0.42 = -0.2024$).

- Results for C_B: We have two poles which are real and negative and it means the system is stable and without any oscillation. The time constants are 0.1996 and 0.4615 seconds which if we use a half rule for first-order approximation would be 0.5613 seconds. We can see from the graph that the time that C_B reaches 0.63 of its changes is so similar to this amount. The gain we calculated is 0.08 which is almost equal to the amount we derive from the graph (approximately $0.0329/0.42 = 0.078$). The transfer function also has a zero equal to -0.85 so it changed the shape of the response(as it is negative) and caused a jump in the first moments as it is shown in the graph.

Step on disturbance (k_2):

- Results for C_A: As we can observe from equation (5), C_A doesnât have to do with k_2. Therefore, the plot is just a constant horizontal line and does not change with k_2.
- Results for C_B: We just have one pole which is real and negative and it means the system is stable and without any oscillation. The time constant is 0.4615 seconds. We can see from the graph that the time that C_B reaches 0.63 of its changes is so similar to this amount. The gain we calculated is -0.917 which is almost equal to the amount we derive from the graph (-0.148/0.167=-0.886).

2.4.2 Linearization

We approximated applying the first order Taylor polynome for multi-variable functions, obtaining (5), (8).
Ca original model:

$$\frac{dC_A}{dt} = \frac{C_{Af}q}{V} - \frac{C_Aq}{V} - C_Ak_1 - C_A^2k_3$$

Ca linearized form:

$$\frac{dC_A}{dt} = \frac{q^*}{V} \cdot \Delta C_{Af} + (\frac{-q^*}{V} - k_1^* - 2C_A^*k_3^*) \cdot \Delta C_A + (\frac{C_Af^*}{V} - \frac{C_A^*}{V}) \cdot \Delta q - C_A^* \cdot \Delta k_1$$

Preliminarly to the simulation we observe that every addend has exactly two non constant factors and that the main recurrent operand, so the value that will be affected the most by the approximation, is C_A.
The steps in the disturbances confirm our prior assumptions (9,8,7,6):

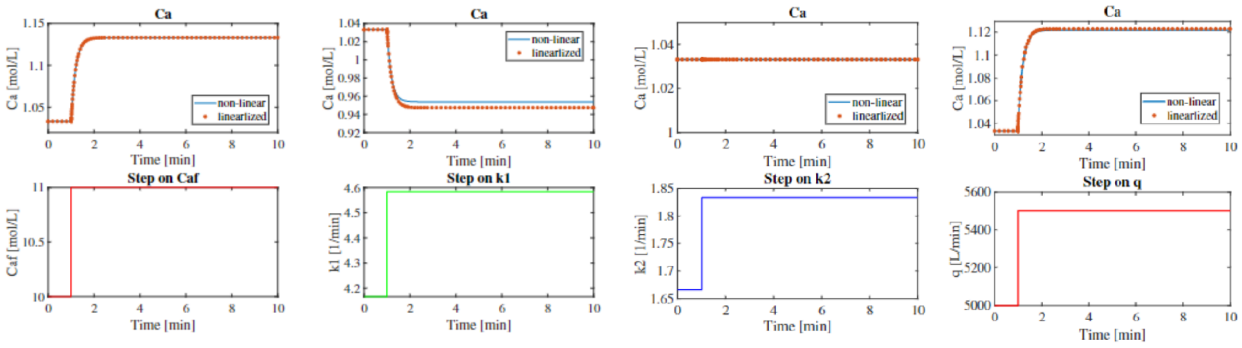

Figure 11: Comparison of the steps in C_A

- C_A is affected by the step in C_Af, but we don't measure a substantial difference between the model and the approximation because of the low recurrence of the disturbance.
- The step on k_1 produces the wider difference in value between the original and the linearized model because of the approximation of C_A to its steady state value.
- We don't see any effect of the step in k2 because there it has no dependecy with r_2, so it's not relevant to the discussion.
- For the step on q we measure a little variation between the two models.

C_B original model:

$$\frac{dC_B}{dt} = \frac{-C_Bq}{V} + k_1C_A - k_2C_B$$

C_B linearized form:

$$\frac{dC_B}{dt} = k_1 * \Delta C_A + (-\frac{q^*}{V} - k_2^*)\Delta C_B + (\frac{-C_B^*}{V})\Delta q + C_A^*\Delta k_1 + (-C_B^*)\Delta k_2$$

Preliminarly we can infer from the formula that a deviation in the approximation will derive by the addend (C_A k1): C_A itself has an high dependecy with k1 that is remarked in this equation.

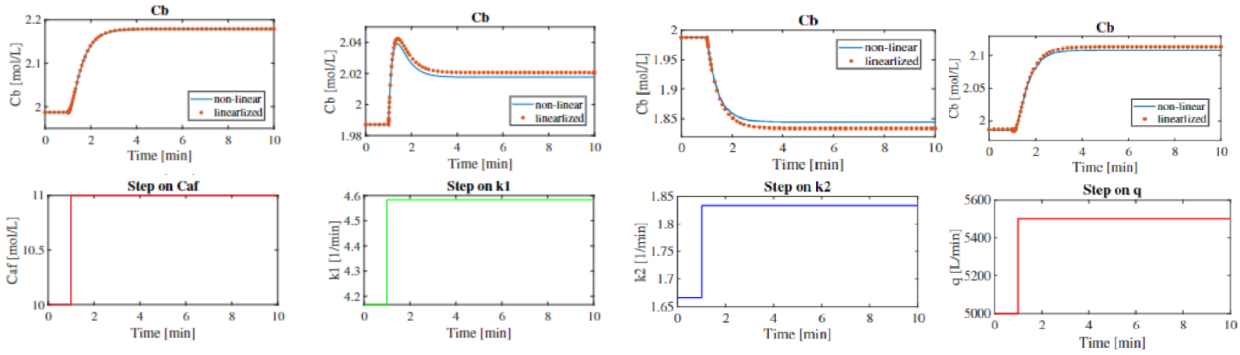

Figure 12: Comparison of the steps in C_A

- C_B is affected by the step in C_{Af} because of its impact on C_A, we don't measure a remarkable difference for the same conclusions made on C_A
- Observing the step on k_1 we can note the deviation between the models.
- The step on k2 has the biggest difference between the models. This could be derived by a cascade impact of the approximation in the reactions: r2 relies on the product of r1.
- For the step on q we measure a little variation between the two models.

2.4.3 Half-rule

Transfer function of C_B:

$$G_{C_B}(s) = \frac{2.524 \cdot 10^{-4}(1 - 0.0726s)}{(0.1996s + 1)(0.4615s + 1)}$$

The half rule approximation of the transfer function of C_B:

$$G_{C_B}(s) = 2.524 \cdot 10^{-4} \frac{e^{-0.1724s}}{0.5163s + 1}$$

- for the first 150-200ms our approximation doesn't react due to the delay, evicting the initial undershooting of the model
- until the first 1.5 seconds the approximation will slightly overcome the original function
- and from then the approximation and the half rule are quite indistinguishable

The half rule gives us a good approximation of the function C_B.

3 Part 3: Controller Design

In this part of the project we designed, tested and analysed a control structure for the process.

3.1 Control structure

Assuming the perfect level control we have constant volume in the tank. Taking into account that the temperature is manipulated by another controller with perfect mixing, we are left with q as manipulated variable, C_B as output, and C_{Af} as a disturbance. k_1 and k_2 rely on the temperature change so assuming the temperature to be constant allows us to regard C_{Af} as the only important disturbance with regards to the differences between the two control structures.

We prioritize a feedforward controller over a cascade controller measurement.This is mainly because we don′t have a variable in the system that responds faster to changes than the control variable C_B.
We came up with two different control structures:

- Feedback only controller:

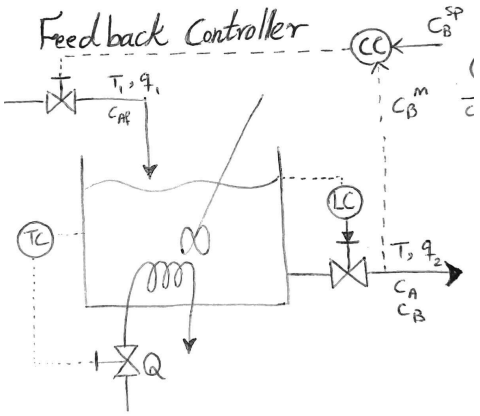

Figure 13: Process diagram with feedback controller

  - Pro: Less complicated control structure (as no model is needed)
  - Pro: Correcting itself when negative feedback
  - Con: Could be late in the response due to the delay in the measurement of the concentration
  - Con: Might cause instability in the case of controller overreaction

- Feedback and feedforward together

  - Pro: Better response for changes in C_{Af}
  - Con: More complex and expensive control structure
  - Small con: Only works for known and measurable disturbances

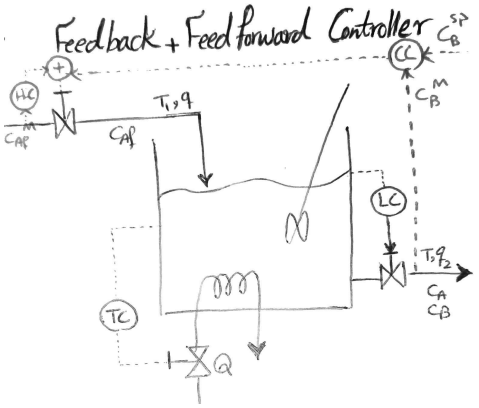

Figure 14: Process diagram with feedback & feedforward

3.2 Controller tuning

In part 2, the half rule approximation of the transfer function of C_B was derived:

$$G_{C_B}(s) = 2.524 \cdot 10^{-4} \frac{e^{-0.1724s}}{0.5163s + 1}$$

Using SIMC rules, the PI controller can be tuned and the result is shown below (for a tight control: τ_c=θ)

$$K_C = \frac{\tau_1}{k} \cdot \frac{1}{(\tau_c + \theta)} = \frac{0.5163}{2.524 \cdot 10^{-4} \cdot (2 \cdot 0.1724)} = 5909$$

$$\tau_I = min(\tau_1, 4(\tau_c + \theta)) = min(0.5163, 4(2 \cdot (0.1724)) = 0.5163$$

so the PI-controller based on first-order model would be:

$$c(s) = K_C(1 + \frac{1}{\tau_I s}) = K_C \cdot \frac{\tau_I s + 1}{\tau_I s} = \frac{5909(0.5163s + 1)}{0.5163s} \tag{29}$$

Our plan was adding a feed-forward controller on the feed-back one which measures C_{Af} as a disturbance. For that goal we used the equations below and assumed the controller to be ideal:

$$y = (g \cdot c_{FF} \cdot g_{dm} + g_d) \cdot d$$

ideal controller: $(g \cdot c_{FF} \cdot g_{dm} + g_d) = 0$

assuming $g_{dm} = 1$:

$$c_{FF} = \frac{-g_d}{g} = \frac{\frac{-0.19192}{(0.1996s+1)(0.4615s+1)}}{\frac{2.524 \cdot 10^{-4}(1-0.0726s)}{(0.1996s+1)(0.4615s+1)}} = \frac{-7.6 \cdot 10^2}{(1 - 0.0726s)}$$

As we can observe the transfer function we derived for feed-forward controller is non-realizable since its real part of denominator root is positive. In this case we will use a proportional controller with the gain k = -760.

3.3 Simulations

For testing which of the control structures is most efficient for our system, we implemented both a single feedback controller and a combined structure of a feedback and feedforward controller in the Simulink file used in part 1.
A 10% step change in the disturbances C_{Af}, k_1, k_2 and the input q is applied to the model. The output graphs can be seen in figure 15-18.

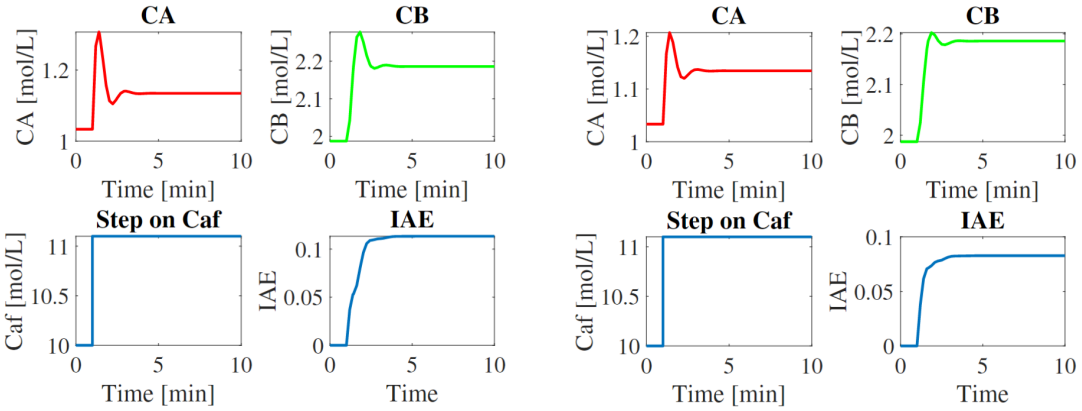

Figure 15: Simulation of a 10% step in C_{Af}: LH feedback only, RH feedback feedforward

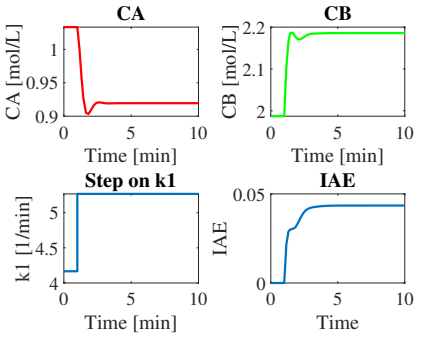

Figure 16: Simulation of a 10% step in k_1 with control

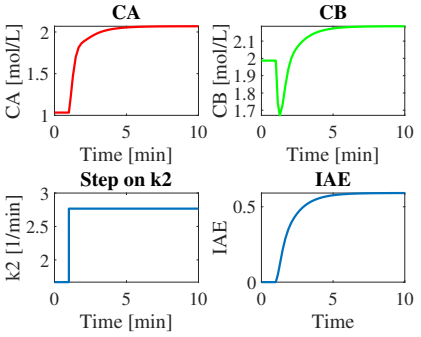

Figure 17: Simulation of a 10% step in k_2 with control

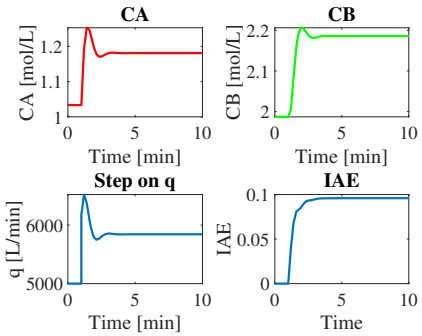

Figure 18: Simulation of a 10% step in q with control

When looking at the differences between a step in the feedback-only control structure and the combined feedback/feedforward control, only the step in C_{Af} causes a difference in the responses of C_A and C_B. Therefore, only this one is shown with two plots while only one graph is shown for the step changes in q, k_1 and k_2.

As can be seen from figure 15, the concentrations of both C_A and C_B responds to the step change by increasing quickly and then decreasing to a new steady state, which is higher than the initial steady state. The spike in concentration is bigger for the feedback-only control, while the combined feedback/feedforward control gives a smaller overshoot before settling at the steady state value. Thus, both control structures arrive at the same steady-state value.

In figure 16 it is seen how the concentration of A lowers with a step in k_1 while the concentration of B is rising. Both graphs either undershoot or overshoot a little bit before reaching the steady state.

Figure 17 shows how C_A increases steadily until reaching steady state, without any overshooting. From the same figure it is seen how the concentration profile of C_B initially drops and then increases steadily to a new steady state concentration.

Finally, it is seen from figure 18 shows a response which for C_A and C_B reminds a lot of the concentration profiles for a step in C_{Af}. Thus, it is seen how a step on q does not give a step response like the previous steps did. This is caused by q being the variable directly controlled by the control structure implemented.

3.3.1 Integral Absolute Error

In order to quantitatively analyze the response of both our systems we calculated the IAE (Integral Absolute Error) performance index.

$$IAE = \int_0^\infty |e(t)|dt \tag{30}$$

And our simulation works in the laplace domain, so:

$$\mathcal{L}\{IAE(t)\} = IAE(s) = \frac{|e(s)|}{s}$$

3.4 Bode plot

The transfer function which shall be used for the Bode Plot is L= C(s)g(s) for the feed-forward controller:

$$L(s) = C_{FB} \cdot G = \frac{5909(0.5163s+1)}{0.5163s} \cdot \frac{2.524 \cdot 10^{-4}(1-0.0726s)}{(0.1996s+1)(0.4615s+1)} = \frac{2.89 \cdot (1-0.0726s)(0.5163s+1)}{s \cdot (0.1996s+1)(0.4615s+1)} \tag{31}$$

- Poles and zeros in the Bode plot.
  Zeros are the roots of the numerator, hence here we have:

  1. (1-0.0726s) = 0 so z_1 = 13.77
  2. (0.5163s+1) = 0 so z_2 = -1.94

  Poles are the roots of the denominator, hence here we have:

  1. (0.1996s+1) = 0 so p_1 = -5.01
  2. (0.4615s+1) = 0 so p_2 = -2.16
  3. p_3 = 0

- Drawing the asymptotes in the Bode plot
  The bode plot we got from the Matlab has a shift value of 360 degrees for phase axis; therefor 180 deg of the plot's phase axis is equal to -180 deg.

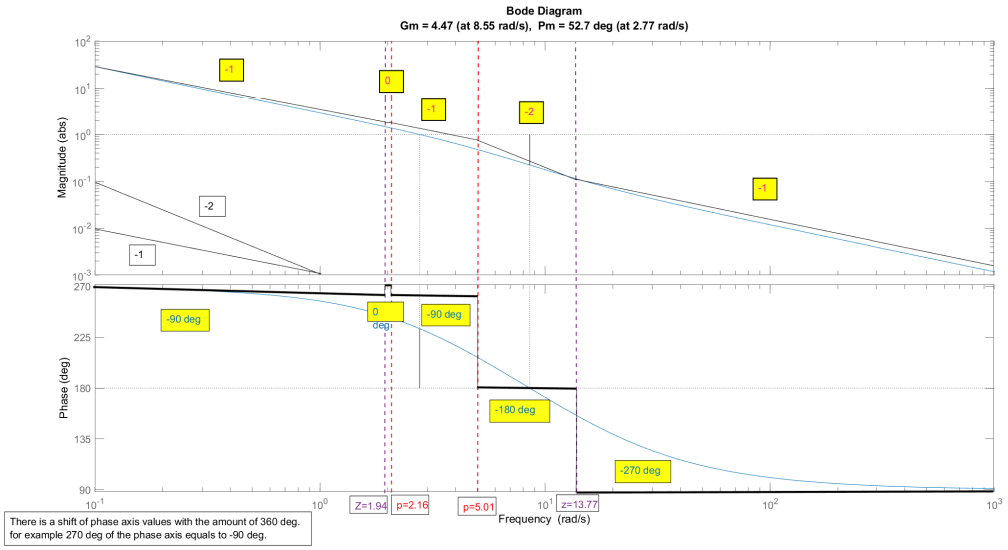

Figure 19: Indicating asymptotes and poles and zeros in the bode plot
For better image quality, please see attached files.

- Phase margin and gain margin indicated in the Bode plot

- Gain margin:

$$GM = \frac{1}{|L|_{(\omega_{180})}} = \frac{1}{0.224} = 4.47$$

- Phase margin:

$$PM = \phi_{(\omega_c)} + 180 = -127.3 + 180 = 52.7deg = 0.9198rad$$

- Delay margin:

$$DM = \frac{PM}{\omega_c} = \frac{0.9198}{2.77} = 0.33$$

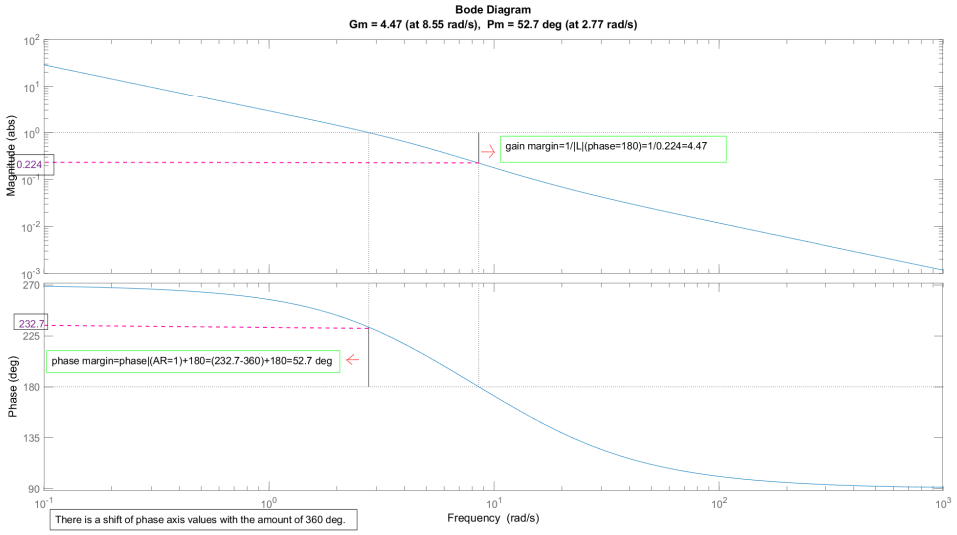

Figure 20: Indicating phase margin and gain margin in the bode plot
For better image quality,please see attached files.

3.5 Discussion

Among the choices we have for a controller, we selected the combined feedback and feedforward controller based on measurements of C_{Af}. In this case, we put a feedforward controller on the main feedback controller as seen in the following figure.

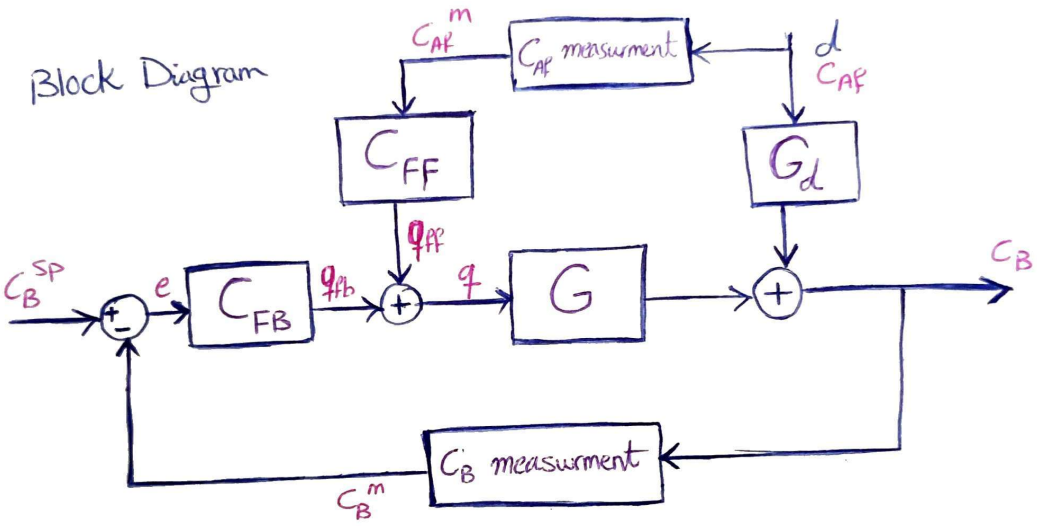

Figure 21: Process diagram for the system modelled

This controller structure improves the response upon the disturbance C_{Af}, as we assessed by the simulation of a step on this disturbance 15. Both the structures make the C_A and C_B converge to the same values in the same time but there is a substantial difference in the overshoot on the states: The difference in overshoot for C_A:
With feedback only:

$$OvershootC_A = C_{A_\infty} - C_{A_{max}} = 1.307\text{mol} \cdot \text{L}^{-1} - 1.124\text{mol} \cdot \text{L}^{-1} = 0.183\text{mol} \cdot \text{L}^{-1}$$

$$OvershootC_B = C_{B_\infty} - C_{B_{max}} = 2.278\text{mol} \cdot \text{L}^{-1} - 2.186\text{mol} \cdot \text{L}^{-1} = 0.092\text{mol} \cdot \text{L}^{-1}$$

With feedback and feedforward:

$$OvershootC_A = C_{A_\infty} - C_{A_{max}} = 1.207\text{mol} \cdot \text{L}^{-1} - 1.124\text{mol} \cdot \text{L}^{-1} = 0.083 \quad \text{mol} \cdot \text{L}^{-1}$$

$$Overshoot C_B = C_{B_\infty} - C_{B_{max}} = 2.202\text{mol} \cdot \text{L}^{-1} - 2.186\text{mol} \cdot \text{L}^{-1} = 0.016 \quad \text{mol} \cdot \text{L}^{-1}$$

$$C_{Aimprovement} = \frac{0.183}{0.083} = 220\%$$

$$C_{Bimprovement} = \frac{0.092}{0.016} = 575\%$$

A strong suggestion to go towards the feedback plus feedforward!

3.5.1 IAE

The main result that we can extract from the measurement of the Integral Absolute Error between the two implementations is from the step on C_{Af}: the IAE value resulted by the feeback only structure is 0.1133 while the one from feedback and feedforward is 0.0828. We can say that this bring a 136% improvement on the IAE measurement:

$$IAE_{Improvement} = \frac{0.1133}{0.0828} = 136\%$$

This is another indicator of a better robustness of the feedback plus feedforward controller!

3.5.2 Margins

Gain and phase margin are the amounts of change in open-loop gain and phase needed to make a closed-loop system unstable, respectively. More gain and phase margin means more stability and less oscillations, but the system has a sluggish response. In general, the gain margin of 2-10 dB and the phase margin of 30-60 degrees are desirable in the closed-loop system design. As we can observe, in our Bode plot for ϕ equals -180 deg, the ω_{180} has the AR equal to 0.224 and it is less than 1. This means that our system is stable. Also, for AR equals 1, the ω_c is 2.77 rad/s. In which the ϕ is -127.3 deg and it is more than -180 deg. This means that our system is stable. Meanwhile, our gain and phase margins which are equal to 4.47 and 52.7 deg(respectively) are placed in the desirable region of margins amounts meaning that we have almost a good control system.

Bibliography

[1] S. Skogestad. *Chemical and Energy Process Engineering.* CRC Press, 1. edition, 2009.

A Additional calculations

A.1 Part 1

Steady state gains:

- 10% step on q:

$$k = \frac{2.108 - 1.987}{5500 - 5000} = 0.000242[\frac{molmin}{l^2}]$$

$$k' = \frac{(2.0223 - 1.9946)/0.2}{5500 - 5000} = 0.000277[\frac{mol}{l^2}]$$

- 10% step on C_{Af}:

$$k = \frac{2.178 - 1.987}{11 - 10} = 0.191[1]$$

$$k' = \frac{(2.0566 - 2.0135)/0.2}{11 - 10} = 0.21[\frac{1}{min}]$$

- 10% step on C_{k1}:

$$k = \frac{2.0176 - 1.987}{4.5833 - 4.1666} = 0.0732[\frac{molmin}{l}]$$

$$k' = \frac{(2.039 - 2.0325)/0.2}{4.5833 - 4.1666} = 0.079136[\frac{mol}{l}]$$

- 10% step on C_{k2}:

$$k = \frac{1.845 - 1.987}{1.833 - 1.667} = -0.855[\frac{molmin}{l}]$$

$$k' = \frac{(1.901 - 1.934)/0.2}{1.833 - 1.667} = -0.994[\frac{mol}{l}]$$

B Simulink

B.1 Simulink Part 1

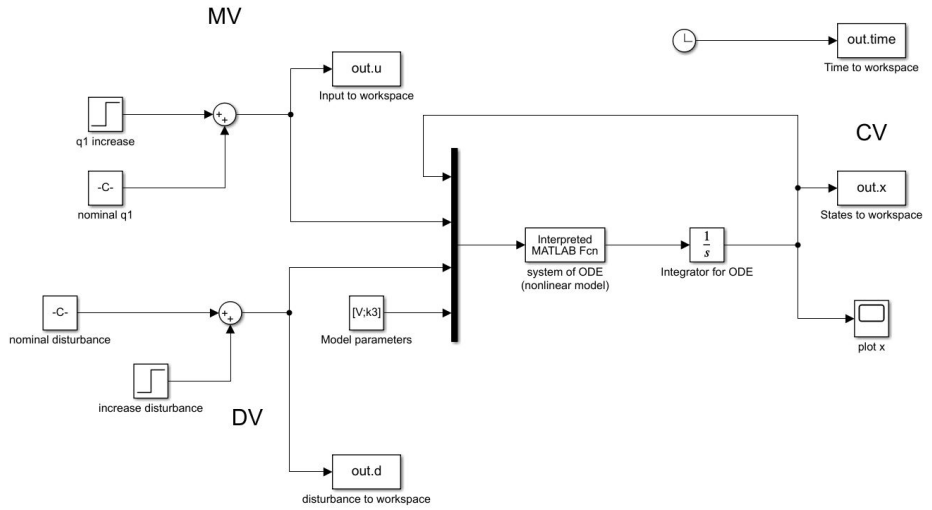

Figure 22: Screenshot of Simulink Block Diagram for part 1

B.2 Simulink Part 2

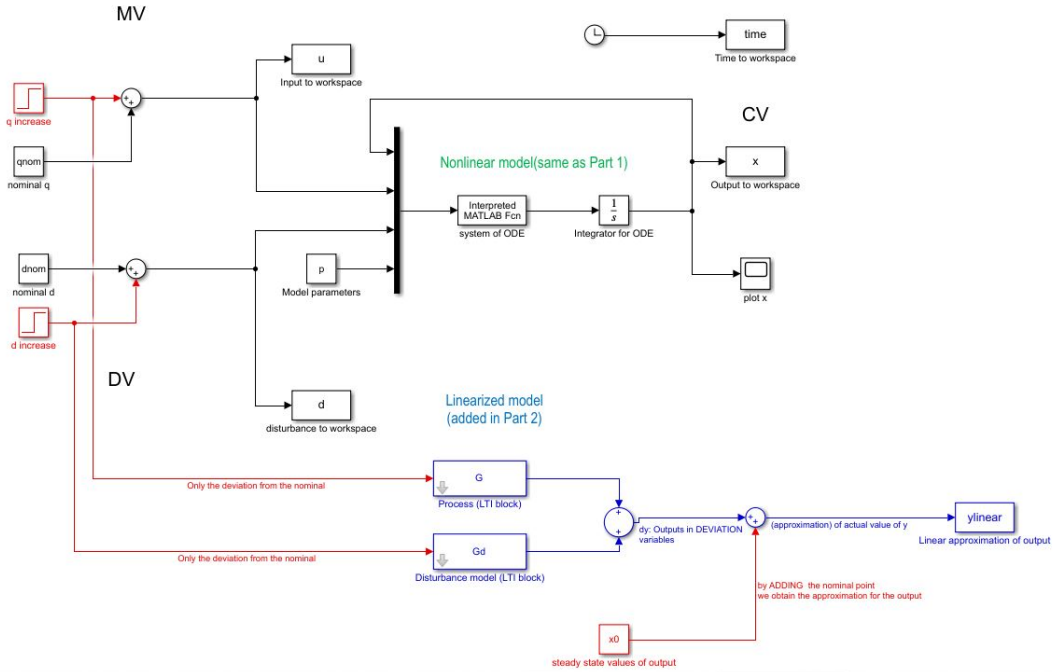

Figure 23: Screenshot of Simulink Block Diagram for part 2

B.3 Simulink Part 3

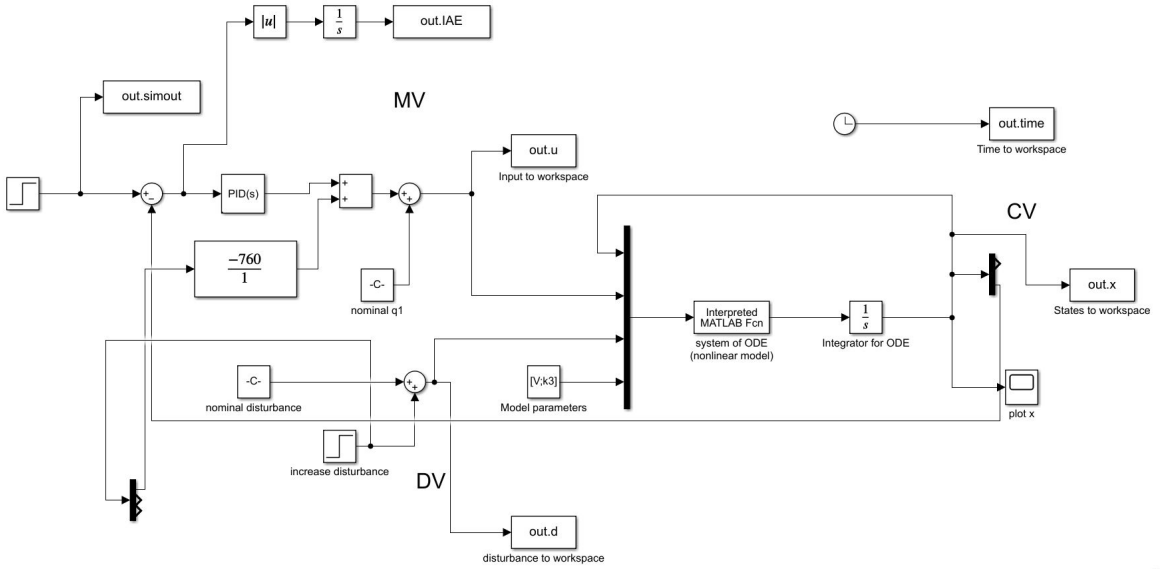

Figure 24: Screenshot of Simulink Block Diagram for part 3, Combined feedback and feed-forward control

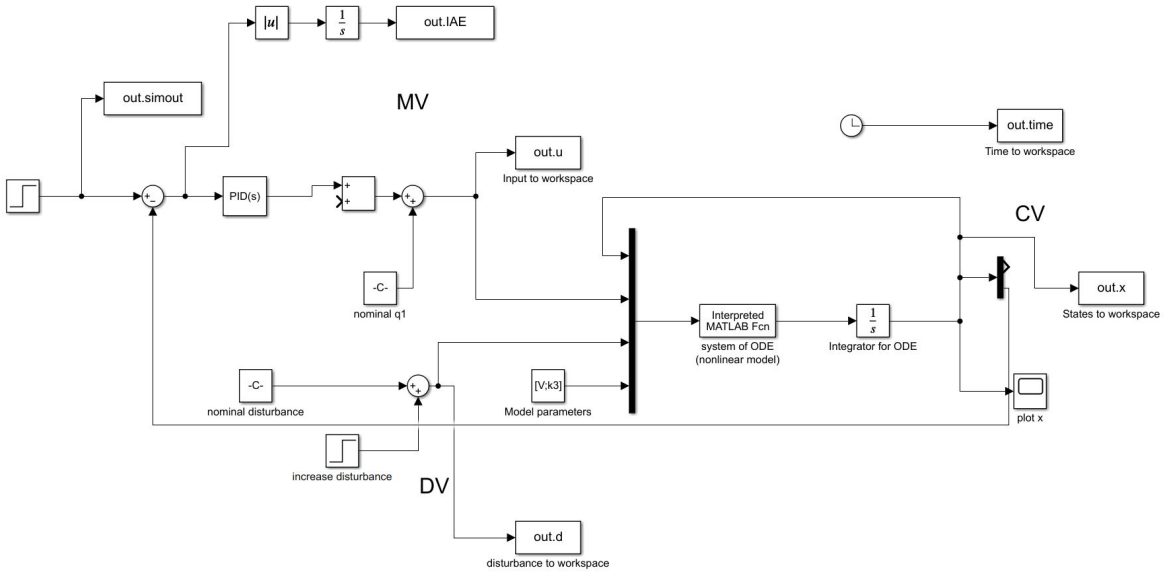

Figure 25: Screenshot of Simulink Block Diagram for part 3, feedback control

C Matlab code

C.1 Part 1

Matlab code for Part 1

Listing 1: Matlab code: script part 1

```
%preset
%warning off
clc
clear
close all

set(0,'DefaultTextFontName','Times',...
'DefaultTextFontSize',18,...
'DefaultAxesFontName','Times',...
'DefaultAxesFontSize',18,...
'DefaultLineLineWidth',2,...
'DefaultLineMarkerSize',7.75)

% Defining the number of variales and model parameters
Nx = 2; %%  Ca, Cb
Nu = 1; %%  q
Nd = 3; %%  Caf, k1, k2
Np = 2; %%  V, k3
Ny = Nx;


%% INITIALIZING
%Model parameters
% Volume, k3
V = 10000;     %Volume
k3 = 1 / 6;    %k3

p = [V; k3];  %Parameters

% Nominal values
q_nom = 5000;
%   u = [q_nom];

%Disturbances
Caf= 10;
k1 = 25/6;
k2 = 5/3;

d = [Caf; k1; k2];


%initializing states
Ca0 = 1.0332964;
Cb0 = k1*V*Ca0/(q_nom+k2*V);
Cc0=k2*V*Cb0/q_nom;
Cd0=k3*V*Ca0^2/q_nom/2;

x0 = [Ca0; Cb0]; % Initial states

%% Simulate a step increase on input (q)
q_step = 0.1*q_nom;
Caf_step = 0;
k1_step = 0;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
```

```
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%% Running Simulink!!
tsim = 10;
sim('projectSimulink')


%% PLOT
%We are using subplots to plot the result.
% We want 3 plots (2 columns and 2 rows)

figure(1)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
ylim([0.9 1.15]);
xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
ylim([1.8 2.2]);
xlim([0 6]);

subplot(223)
plot(ans.time,ans.u)
title('Step on q');
ylabel('q [L/min]');
xlabel('Time [min]');
xlim([0 6]);

%% Simulate a step increase on feed concentration (Caf)
q_step = 0;
Caf_step = 0.1*Caf;
k1_step = 0;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%% Running Simulink!!
tsim = 10;
sim('projectSimulink')


%% PLOT
figure(2)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
ylim([0.9 1.15]);
xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
```

```
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
ylim([1.8 2.2]);
xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,1))
title('Step on Caf');
ylabel('Caf [mol/L]');
xlabel('Time [min]');
xlim([0 6]);

%% Simulate a step increase on k1
q_step = 0;
Caf_step = 0;
k1_step = 0.1*k1;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%% Running Simulink!!
tsim = 10;
sim('projectSimulink')


%% PLOT
figure(3)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
ylim([0.9 1.15]);
xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
ylim([1.8 2.2]);
xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,2))
title('Step on k1');
ylabel('k1 [1/min]');
xlabel('Time [min]');
xlim([0 6]);

%% Simulate a step increase on k2
q_step = 0;
Caf_step = 0;
k1_step = 0;
k2_step = 0.1*k2;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;
```

```
%% Running Simulink!!
tsim = 10;
sim('projectSimulink')


%% PLOT
figure(4)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
ylim([0.9 1.15]);
xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
ylim([1.8 2.2]);
xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,3))
title('Step on k2');
ylabel('k2 [1/min]');
xlabel('Time [min]');
xlim([0 6]);
```

Listing 2: Matlab code: SysODE part 1

```
% This function defines the system of ordinary differential equations (ODE)
function  dxdt = SysODEKira(x,u,d,p)

% ==========MODEL PARAMETERS=========================

%states
Ca = x(1)
Cb = x(2)

%input
q1 = q1nom;

%disturbances
Caf = d(1);
k1  = d(2);
k2  = d(3);

%parameters
V = p(1);
k3 = p(2);

%Differential equations
dxdt(1) = (Caf*q1)-(Ca*q1)-V*Ca*(k1+Ca*k3);    % This corresponds to the first state.
dxdt(2)= -Cb*q1+k1*V*Ca-k2*Cb*V;

dxdt = dxdt(:); % system of differential equations as vertical vector
```

C.2 Part 2

Matlab code for Part 2

Listing 3: Matlab code: script part 2

```
% TKP4140 Process Control Project - Group E2

warning off
clc
clear
close all

%% Set default options for plotting. You can change this to your preferences
set(0,'DefaultTextFontName','Times',...
'DefaultTextFontSize',18,...
'DefaultAxesFontName','Times',...
'DefaultAxesFontSize',18,...
'DefaultLineLineWidth',2,...
'DefaultLineMarkerSize',5)

%% Initializing

%Compute the transfer functions
[G,Gd]=computeTransferFunctions; %Get G and Gd from computeTransferFunctions

% Hint:
% If you have more than one MV, CV, or DV, be careful with the order of the
% states, inputs, and disturbances when connecting the Simulink diagram.
% The order must correspond to the order you use in the
% computeTransferFunctions function. It is also very recommendable to be
% consistent with the order you used in the nonlinear model in part 1. In
% other words, if h was x1 and T was x2, you should always keep it so, and
% don't use T as x1 h as x2.



%Model parameters
V = 10000;
k3 = 1/6;
p = [V, k3];

% Nominal  values for  q1 and q0.
qnom=5000;
dnom=[10, 25/6, 5/3];

% Set steps to 0 - Operating at nominal point

q_step=0;
d_step=[0,0,0];

% Define the number of variales and model parameters
% These are used in the Interpreted Matlab Fcn block in Simulink
Nx = 2; %%  Ca, Cb          %--CHANGE HERE-- number of states--NUMBER OF DIFFERENTIAL EQUATIONS
Nu = 1; %%  q1              %--CHANGE HERE-- number of inputs--NUMBER OF MVS
Nd = 3; %%  Caf, k1, k2     %--CHANGE HERE-- number of disturbances--NUMBER OF DVS
Np = 2; %%  V, k3           %--CHANGE HERE-- number of model parameters--CONSTANT VALUES
Ny = Nx;%% Cb

%INITIAL CONDITIONS FOR THE STATES
Ca0 = 1.03329638;
Cb0 = 1.987108423;

x0 = [Ca0; Cb0];

%% Check what happens when there is a disturbance

%step on Caf
d_step=[0.1*10,0,0];
q_step = 0;
```

```
% Set the time for giving the Step on q0
d_t_step = [1,1,1];
q_t_step = 0;

%run Simulink model to test the step in q0
tsim = 10; % set simulation time. same units as your model
sim('SimulinkPart2')

%We are using subplots to plot the result.
% We want 9 plots (3 columns and 3 rows)
figure(1)
subplot(223)
plot(time,d(:,1),'red')   %corresponds to Caf
title('Step on C_{Af}')
ylabel('C_{Af} [mol/L]')
xlabel('Time [min]')

subplot(221)
plot(time,x(:,1)) %corresponds to Ca
hold on
plot(time,ylinear(:,1),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_A')
ylabel('C_A [mol/L]')
xlabel('Time [min]')
ylim([1.02 1.15])

subplot(222)
plot(time,x(:,2)) %corresponds to Cb
hold on
plot(time,ylinear(:,2),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_B')
ylabel('C_B [mol/L]')
xlabel('Time [min]')
ylim([1.95 2.2])

%step on k1

d_step=[0,0.1*(25/6),0];
%q_step = 0;

%run Simulink model to test the step in q0
tsim = 10; % set simulation time. same units as your model
sim('SimulinkPart2')

%plotting in same plot as before
figure(2)
subplot(223)
plot(time,d(:,2),'green')   %Corresponds to k1
title('Step on k_1')
ylabel('k_1 [1/min]')
xlabel('Time [min]')


subplot(221)
plot(time,x(:,1)) %Corresponds to Ca
hold on
plot(time,ylinear(:,1),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_A')
ylabel('C_A [mol/L]')
xlabel('Time [min]')
ylim([0.92 1.04])

subplot(222)
```

```
plot(time,x(:,2))   %corresponds to Cb
hold on
plot(time,ylinear(:,2),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_B')
ylabel('C_B [mol/L]')
xlabel('Time [min]')
ylim([1.98 2.05])

%step k2
d_step=[0,0,0.1*(5/3)];
%q_step = 0;

% Set the time for giving the Step on q0
d_t_step = [1,1,1];


%run Simulink model to test the step in q0
tsim = 10; % set simulation time. same units as your model
sim('SimulinkPart2')

%plotting the results
figure(3)
subplot(223)
plot(time,d(:,3),'blue')   %Corresponds to k2
title('Step on k_2')
ylabel('k_2 [1/min]')
xlabel('Time [min]')


subplot(221)
plot(time,x(:,1)) %Corresponds to Ca
hold on
plot(time,ylinear(:,1),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_A')
ylabel('C_A [mol/L]')
xlabel('Time [min]')
ylim([1 1.05])

subplot(222)
plot(time,x(:,2))  %Corresponds to Cb
hold on
plot(time,ylinear(:,2),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_B')
ylabel('C_B [mol/L]')
xlabel('Time [min]')
ylim([1.82 2])

%Doing a step on q
%step k2
d_step=[0,0,0];
q_step = 0.1*qnom;

% Set the time for giving the Step on q0
q_t_step = 1;

%run Simulink model to test the step in q0
tsim = 10; % set simulation time. same units as your model
sim('SimulinkPart2')

%plotting the results
figure(4)
subplot(223)
plot(time,u,'red') %Corresponds to q
```

```
title('Step on q')
ylabel('q [L/min]')
xlabel('Time [min]')
%ylim([4999 5002])

subplot(221) % in a 2 line 2 column array plot, 3rd position (lower left)
plot(time,x(:,1)) %Corresponds to Ca
hold on
plot(time,ylinear(:,1),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_A')
ylabel('C_A [mol/L]')
xlabel('Time [min]')
ylim([1.03 1.13])

subplot(222)
plot(time,x(:,2))   %corresponds to Cb
hold on
plot(time,ylinear(:,2),'*')
legend('non-linear','linearlized','Location','northwest')
title('C_B')
ylabel('C_B [mol/L]')
xlabel('Time [min]')
ylim([1.97 2.13])
```

Listing 4: Matlab code: SysODE part 2

```
% This function defines the system of ordinary differential equations (ODE)
function  dxdt = SysODE(x,u,d,p)

% ==========MODEL PARAMETERS==========================

%states
Ca = x(1)
Cb = x(2)

%input
q = u(1);

%disturbances
Caf = d(1);
k1  = d(2);
k2  = d(3);

%parameters
V = p(1);
k3 = p(2);

%Differential equations
dxdt(1)=(Caf*q)/V-(Ca*q)/V-Ca*(k1+Ca*k3);
dxdt(2)=-Cb*q/V+k1*Ca-k2*Cb;

dxdt = dxdt(:); % system of differential equations as vertical vector
```

Listing 5: Matlab code: computeTransferFunctions part 2

```
%% In this function you:
% - Define the differential equations for your model
% - Linearize and get your model in the the state-space form.
% - Obtain the transfer function matrices G(s) and Gd(s)
%
% To obtain the transfer functions G and Gd from the state-space form:
% Suppose you have the linearized system x_dot=Ax+Bu+Ed, y=C*x; Eq.(1)
```

```
% where x=delta states, u= delta MV, d= delta DV and y= delta CV
% From this we want to compute the transfer function matrices G(s) and Gd(s)
%
% We apply Laplace transform to Eq (1) to obtain:
% G(s)=C*(I*s-A)^-1*B and Gd(s)=C*(I*s-A)^-1*E
% You should know how to derive this by hand! possible exam question
%
% You can also consider that u inclues MVs and DVs. In this case the model
% would be x_dot=Ax+Bd*u, where Bd is a nx-by-(nMV+nDV) matrix.
% In this case you would get only one transfer function matrix G(s).
% Both forms are equivalent, as long you know what your are doing.
%
%%
function [G,Gd]=computeTransferFunctions


%%Definition of differential equations

% Define/create symbolic variables for your MV, DV, CV
syms Caf k1 k2 q Ca Cb

% Tip: if your equations have additional variables that depend on
% your MVs, DVs or CVs and you want to define these additional variables,
% so that you can write your differential equations in a simpler way,
% you can declare these "aid" variables as symbolic variables too. For
% examples: valve_flow, enthalpy_in, etc.

%Define vectors of DV(d), MV(u), states(x) and CV(y)
d = [Caf, k1, k2];           %deviation variables; vertical vector (if more than 1 DV)
u = q;                        %inputs; vertical vector (if more than 1 MV)
x = [Ca, Cb];                  %states; vertical vector (if more than one states)
y = x;                      % outputs; vertical vector (if more than one CV)
% y can be different than the states x, but we recommend to select y=x as
% in part 3 you may decide to control all your states


%Parameters that will be used in the differential equations
V = 10000;
k3 = 1/6;


%Differential equations
f(1,1)=(Caf*q)/V-(Ca*q)/V-Ca*(k1+Ca*k3); f(1,2)=(-Cb*q)/V+(k1*Ca)-k2*Cb;

%% Linearization
% Define the Jacobian - for the state space form x_dot=Ax+Bu+Ed
A=jacobian(f,x);
B=jacobian(f,u);
E=jacobian(f,d);

% Define the Jacobian from states to output: y=Cx+Du
C=jacobian(y,x);                        % Select the outputs from the states
D=jacobian(y,u);                % Matrix from input to output. Usually  D=0

%Define the nominal point for x,y, u, d
Ca = 1.03329638;
Cb = 1.987108423;
q=5000;
Caf=10;
k1=25/6;
k2=5/3;


%Replace the symbolic variables by their corresponding nominal value
A=double(subs(A))
B=double(subs(B))
```

```
E=double(subs(E))

C=double(subs(C))
D=double(subs(D))

%Laplace variable
s=tf('s');

%Identity matrix
n=length(A); I=eye(n);

%Obtaining transfer function matrix - y = G*u+Gd*d
% you must know how to do this by hand! possible exam question.
G=C*inv(s*I-A)*B+D                  % tranfer function from input to output
Gd=C*inv(s*I-A)*E           % tranfer function from disturbance to output

%Simplifying equation - minimum realization (cancels common roots in numerator and denominator)
G=minreal(G);
Gd=minreal(Gd);

%You may change the display format.
G=set(zpk(G), 'DisplayFormat','time constant')
Gd=set(zpk(Gd), 'DisplayFormat','time constant')

%
```

C.3 Part 3

Matlab code for Part 3

Listing 6: Matlab code: script part 3

```
%preset
%warning off
clc
clear
close all

set(0,'DefaultTextFontName','Times',...
'DefaultTextFontSize',18,...
'DefaultAxesFontName','Times',...
'DefaultAxesFontSize',18,...
'DefaultLineLineWidth',2,...
'DefaultLineMarkerSize',7.75)

% Define the number of variales and model parameters
Nx = 2; %%  Ca, Cb          %--CHANGE HERE-- number of states--NUMBER OF DIFFERENTIAL EQUATIONS
Nu = 1; %%  q              %--CHANGE HERE-- number of inputs--NUMBER OF MVS
Nd = 3; %%  Caf, k1, k2     %--CHANGE HERE-- number of disturbances--NUMBER OF DVS
Np = 2; %%  V, k3           %--CHANGE HERE-- number of model parameters--CONSTANT VALUES
Ny = Nx;% Do not change--number of outputs---equal to number of states


%% INITIALIZING
%Model parameters
% Volume, k3
V = 10000;    %Volume
k3 = 1 / 6;   %k3

p = [V; k3];  %Parameters

% Nominal values
q_nom = 5000;
%   u = [q_nom];
```

```
%Disturbances
Caf= 10;
k1 = 25/6;
k2 = 5/3;

d = [Caf; k1; k2];


%initializing states
Ca0 = 1.0332964;
Cb0 = k1*V*Ca0/(q_nom+k2*V);
Cc0=k2*V*Cb0/q_nom;
Cd0=k3*V*Ca0^2/q_nom/2;

x0 = [Ca0; Cb0]; % Initial states

%% Simulate a step increase on input (q)
q_step = 1.1;
Caf_step = 0;
k1_step = 0;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%Defining setpoint value
Cb_set = 1.1*Cb0
%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3FFFeedback')


%% PLOT
%We are using subplots to plot the result.
% We want 6 plots (3 columns and 3 rows)

figure(1)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.u)
title('Step on q');
ylabel('q [L/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
```

```
title("IAE")
ylabel("IAE")
xlabel("Time")

%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3Feedback')


%% PLOT
%We are using subplots to plot the result.
% We want 6 plots (3 columns and 3 rows)

figure(2)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.u)
title('Step on q');
ylabel('q [L/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")

%% Simulate a step increase on feed concentration (Caf)
q_step = 0;
Caf_step = 1.1;
k1_step = 0;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;


%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3FFFeedback')


%% PLOT
figure(3)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
```

```
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,1))
title('Step on Caf');
ylabel('Caf [mol/L]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")

%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3Feedback')


%% PLOT
figure(4)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,1))
title('Step on Caf');
ylabel('Caf [mol/L]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")


%% Simulate a step increase on k1
q_step = 0;
Caf_step = 0;
```

```
k1_step = 1.1;
k2_step = 0;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3FFFeedback')


%% PLOT
figure(5)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,2))
title('Step on k1');
ylabel('k1 [1/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")

%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3Feedback')


%% PLOT
figure(6)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
```

```
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,2))
title('Step on k1');
ylabel('k1 [1/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")

%% Simulate a step increase on k2
q_step = 0;
Caf_step = 0;
k1_step = 0;
k2_step = 1.1;

% Set the time for giving the step on q
q_t_step = 1;
Caf_t_step = 1;
k1_t_step = 1;
k2_t_step = 1;

%% Running Simulink!!
tsim = 10;
sim('projectSimulinkpart3FFFeedback')


%% PLOT
figure(7)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,3))
title('Step on k2');
ylabel('k2 [1/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")


%% Running Simulink!!
tsim = 10;
```

```
sim('projectSimulinkpart3Feedback')


%% PLOT
figure(7)
subplot(221)
plot(ans.time,ans.x(:,1),'red')
title('CA');
ylabel('CA [mol/L]');
xlabel('Time [min]');
%ylim([0.9 1.15]);
%xlim([0 6]);

subplot(222)
plot(ans.time,ans.x(:,2),'green')
title('CB');
ylabel('CB [mol/L]');
xlabel('Time [min]');
%ylim([1.8 2.2]);
%xlim([0 6]);

subplot(223)
plot(ans.time,ans.d(:,3))
title('Step on k2');
ylabel('k2 [1/min]');
xlabel('Time [min]');
%xlim([0 6]);

subplot(224)
plot(ans.time,ans.IAE(:,1))
title("IAE")
ylabel("IAE")
xlabel("Time")
```

Listing 7: Matlab code: SysODE part 3

```
% This function defines the system of ordinary differential equations (ODE)
function  dxdt = SysODE(x,u,d,p)

% ==========MODEL PARAMETERS=========================

%states
Ca = x(1)
Cb = x(2)

%input
q = u(1);

%disturbances
Caf = d(1);
k1  = d(2);
k2  = d(3);

%parameters
V = p(1);
k3 = p(2);

%Differential equations
dxdt(1)=(Caf*q)/V-(Ca*q)/V-Ca*(k1+Ca*k3);
dxdt(2)=-Cb*q/V+k1*Ca-k2*Cb;

dxdt = dxdt(:); % system of differential equations as vertical vector
```