Lab Report Glasgow College, UESTC Dynamic and Control (UESTC 3001)

1. UESTC ID: 2022190504031

2. GUID: 2839915H

3. Name: HeWenXiao

4. Introduction

Dynamic and control is the course that focuses on analyzing and designing dynamic system. A dynamic system is one which is in motion. Electronic circuits, motors and even airplanes are dynamic systems. These lab sessions take some typical dynamic systems that we have learnt on the lectures, aiming to improve our abilities to analyze and implement theses dynamic systems.

5. Lab 1

5.1. Baisc principles

This lab session's topic is the RLC circuit which are commonly used as bandpass and notch filters. The response of the RLC circuit can be classified as **overdamped, critically damped and underdamped**. The transition between initial and final conditions for component voltages and currents is fastest in a critically damped circuit which is difficult to achieve.

5.2. Principles

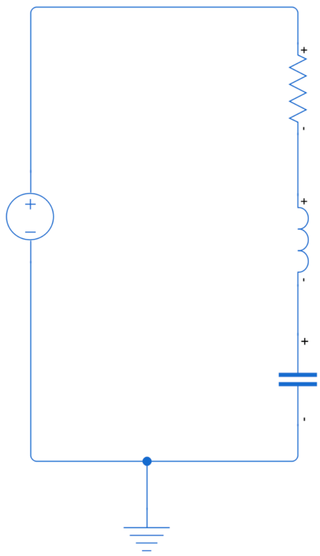

Figure 1: Lab1 setup

A basic RLC circuit is shown in Figure 1. An essential factor that can describe this factor is the damping ratio.

$$\zeta = \frac{R}{2}\sqrt{\frac{C}{L}}$$

- $\zeta > 1$, overdamped

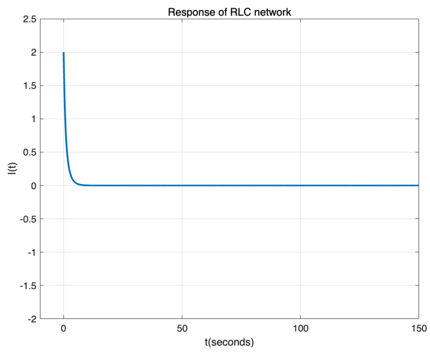

Figure 2: overdamped

- $\zeta = 1$, critically damped

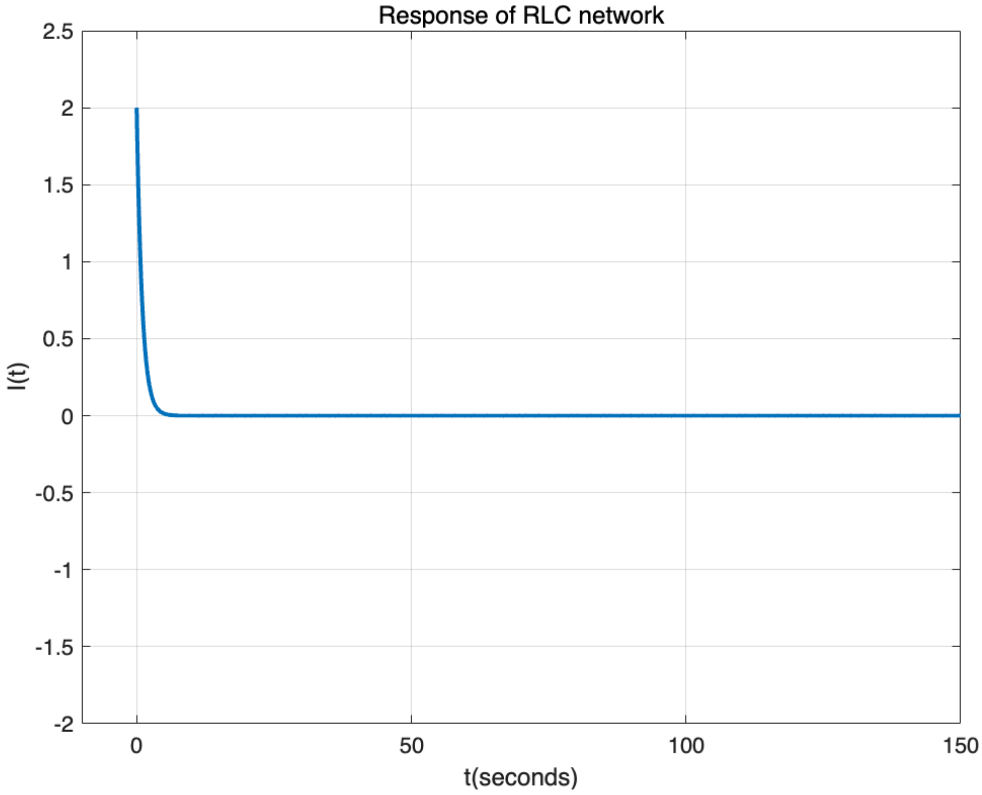

Figure 3: critically damped

- $\zeta < 1$, underdamped

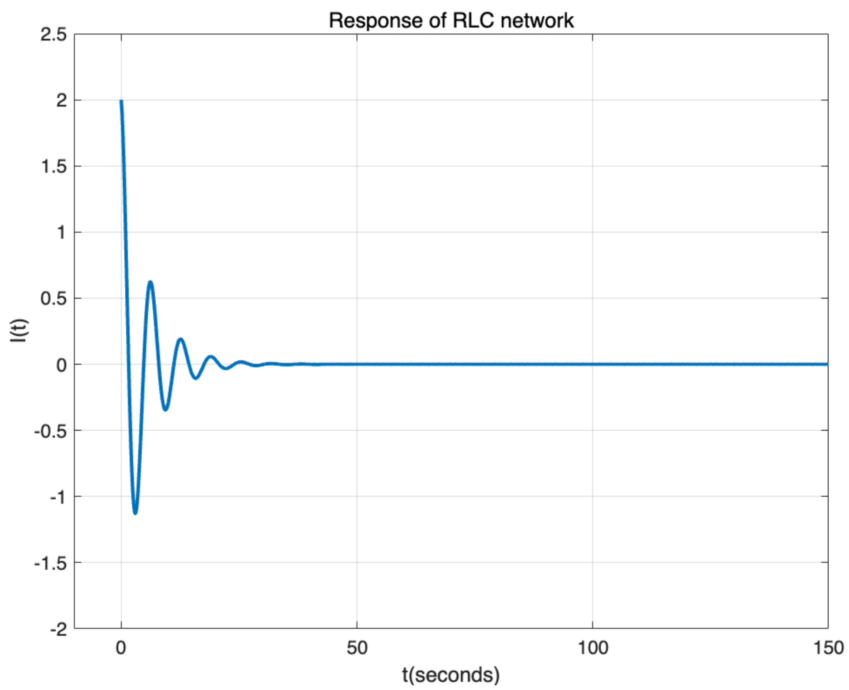

Figure 4: underdamped

5.3. Q2(G)

- Plot_response(ζ)

```
function plot_response(zeta)
    %custom function to plot response of the graph
    t=0:0.1:200;
    omega = 1;
    s1 = -1*omega*(zeta - sqrt(zeta^2-1)); %compute first
variable s1
    s2 = -1*omega*(zeta + sqrt(zeta^2-1)); %compute second
variable s2
    i = 1*exp(s1*t) + 1*exp(s2*t); %compute current taking
unity constants i.e. A1 = A2 = 1
    plot(t, real(i), 'linewidth', 2); %plot command
    xlabel('t (seconds)'), ylabel('I(t)'), title('Response of
RLC network'), grid on; %titles and labels
    if zeta>0 axis tight, else axis([-10 150 -2 2.5]);end %add
flexible axis
end
```

- system_characteristics

```
function [f_n, alpha, zeta, f_d] =
system_characteristics(R_th, R_2, R_ps, R_1)
    %system_characteristics(R_th, R_2, R_ps, R_1)
    %function to compute system characteristics which include
natural
    %frequency, f_n, attenuation, alpha, damping factor, zeta,
and damping frequency, f_d
    %the function takes in R_th, R_2, R_ps and R_1, and
outputs the system
    %characteristics in terms of the natural frequency,
attenuation, the damping factor
    %and the damping frequency

    % to be completed by student!!
    % write code here
    %b = L1*C1;
    %a = b^(1/2);
    %g = (L1*C1)^(1/2)

    C1 = 0.01;
```

```
    f_n = 1/((L1*C1)^(1/2));
    R = (R_th + R_2)/(R_th + R_2) + R_ps + R_1;
    alpha = R/(2*L1);
    zeta = alpha/f_n;
    f_d = f_n*((1-zeta^2)^(1/2));
    f1_n = f_n/(2*pi);
    f1_d = f_d/(2*pi);

    % Comment out the next five lines below after you have
completed your code
    %f_n = 0; zeta = 0; alpha = 0; f_d=0; %comment out this
line after completing your code
    disp(['the natural frequency is: ', num2str(f_n), ' rad/
s']); %code to display output
    disp(['the natural frequency is: ', num2str(f1_n), '
Hz']); %code to display output
    disp(['the attenuation is: ', num2str(alpha)]);
    disp(['the damping factor is: ', num2str(zeta)]);
    disp(['the damping frequency is: ', num2str(f_d), ' rad/
s']);
    disp(['the damping frequency is: ', num2str(f1_d), '
Hz']);
end
```

- function_A

```
function [A1, A2] = function_A(alpha, w_d, vc_inf,
vc_before_0) %#ok<INUSD>
    %functionA(alpha, w_f, vc_inf, vc_before_0)
    %This function is to compute the values of A1 and A2, it
takes in alpha,
    %damping frequency, Vc at t=infinity and Vc just before
t=0.
    % to be completed by student!!
    % write code here
    v_s = 2.91;
    A1 = vc_before_0 - v_s;
    A2 = (A1*alpha)/w_d;

    % Comment out the next four lines below after you have
completed your code
    disp(['the value of A1 is: ', num2str(A1)]); %code to
```

```
display outputs
    disp(['the value of A2 is: ', num2str(A2)]);
end
```

- plot_vc

```
function plot_vc(A1, A2, alpha, w_d)
    %plot_vc(A1, A2, alpha, w_d)
    %function takes in pre-computed values of A1,
    %A2, attenuation Walpha, across capacitor frequency w_d
and gives out a sketch
    %of the voltage across capacitor
    %Detailed explanation goes here
    t = 0:1e-6:0:0.25e-3; %time range for plot
    % to be completed by student!!
    % write code here
    vc = (A1*cos(w_d*t) + A2*sin(w_d*t)) .* exp(-alpha*t) +
2.91;

    % Comment out the next one line below after you have
completed your code
    grid minor;
    plot(t*1e3, vc); %plot command
    title('Voltage across C1') %title and labels
    xlabel('time (ms)')
    ylabel('VC1 (V)')
    axis tight
end
```

- general_characteristics

```
function general_characteristics(L_1, C_1, alphar, A1, A2)
    %general_characteristics(L_1, C_1, alpha, A1, A2)
    %function to compute general system characteristics
    %the function takes in the inductor value L_1, capacitor
C_1, attenuation alpha, A1, A2 (computed earlier)
    %plots the system characteristics

    % to be completed by student!!
    % write code here
    t = 0:1e-6:0.25e-3; %time range for plot
    w_n = 1/((L_1*C_1)^(1/2));
    zeta = alphar/w_n;
```

```
    w_d = w_n*((1-zeta^2)^(1/2));
    vc = (A1*cos(w_d*t) + A2*sin(w_d*t)) .* exp(-alphar*t) +
2.91;

    % Comment out the next one line below after you have
completed your code
    figure %to use new plot window
    plot(t*1e3, vc); %plot command
    title('Voltage across C1') %title and labels
    xlabel('time (ms)')
    ylabel('Vc1 (V)')
    grid minor
    axis tight
end
```

5.4. Q3

5.4.1. A

When adjusting only the parameter α, it solely affects the amplitude or intensity of the oscillation. An increase in α leads to a slight reduction in the oscillation's intensity.

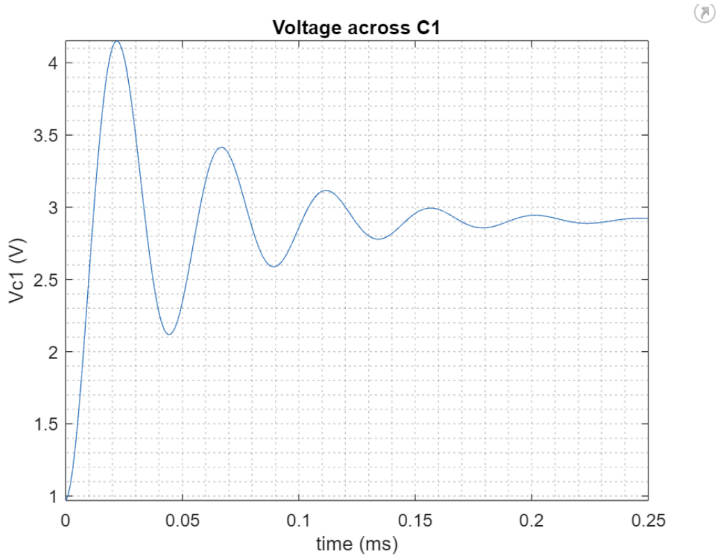

Figure 5: $L_1 = 0.004, C_1 = 1e-8, \text{alphar} = 2e4$

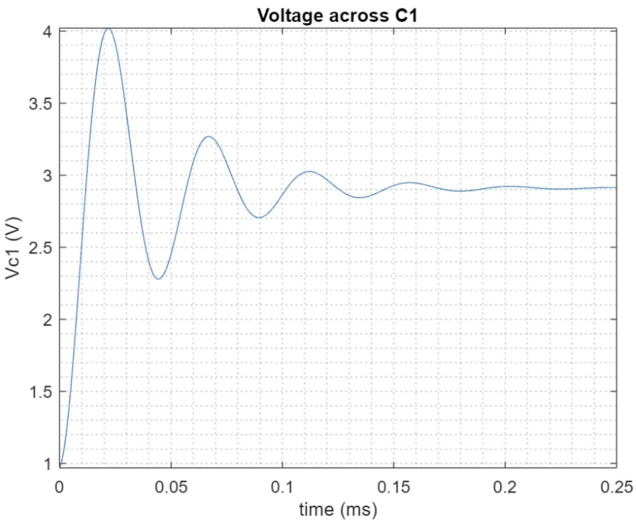

Figure 6: $L_1 = 0.004, C_1 = 1e - 8, \text{alphar} = 25000$

5.5. B

The magnitude of oscillations gradually decreases, eventually leading to a more stable response, as the inductance coefficient is altered. Therefore, increasing the inductance value, particularly L_1, is recommended as an effective strategy to minimize oscillations in the transient response.

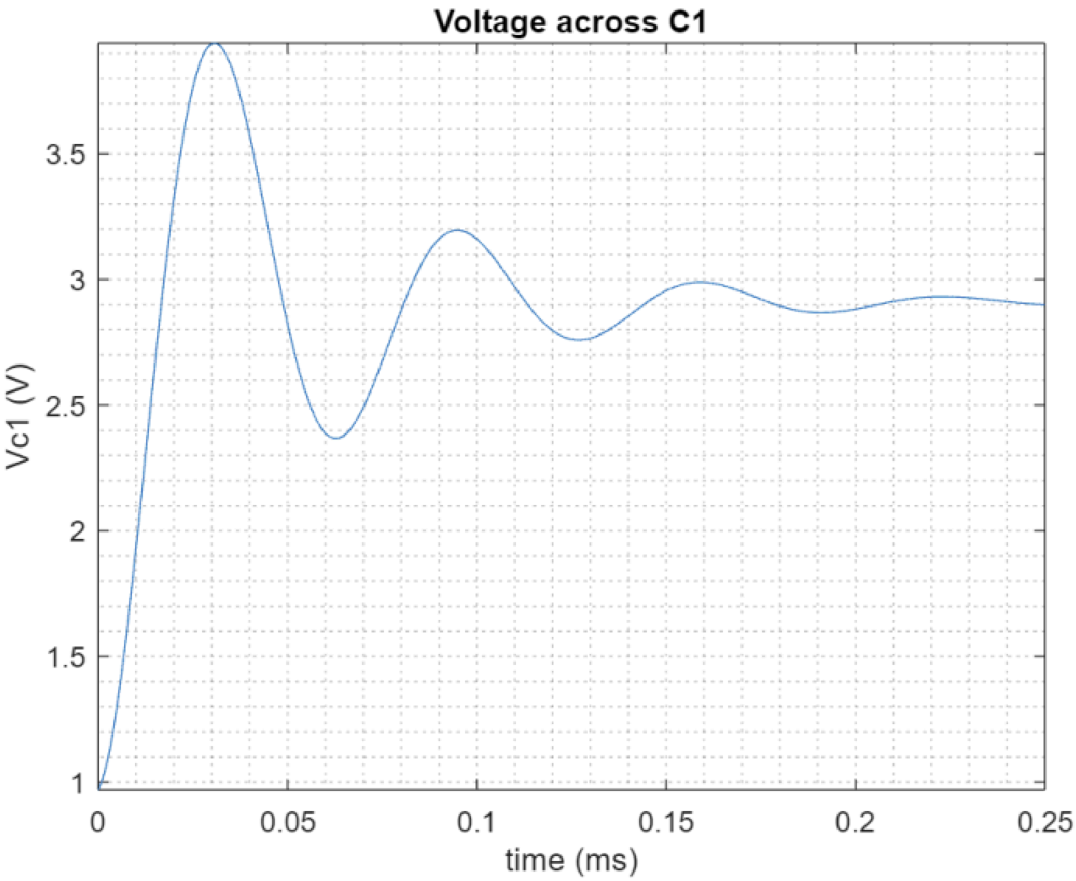

Figure 7: $L_1 = 1e-3, C_1 = 1e-7, \text{alphar} = 2e4$

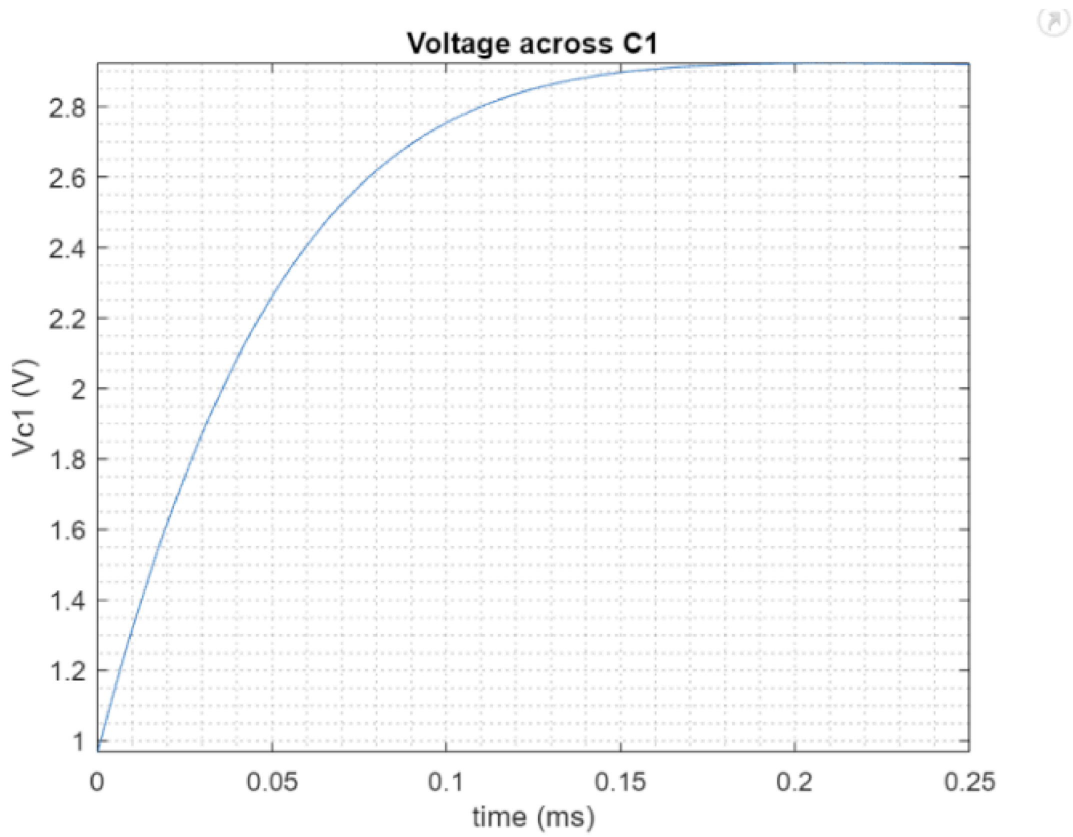

Figure 8: $L_1 = 0.02, C_1 = 1e - 7, \text{alphar} = 2e4$

5.6. C

Based on the examination of the following images, it is evident that keeping L and α constant while raising the capacitance leads to a decrease in oscillation frequency. Moreover, the amplitude of the oscillations undergoes a minor decline. Consequently, to mitigate oscillations in the transient response, it is advisable to enhance the value of C.

5.7. D

In RLC circuits, the rate at which voltage or energy in a capacitor dissipates is significantly affected by the equivalent resistance, regardless of whether it is configured in series or parallel. When damping occurs, the energy held in the capacitor is slowly released following an exponential decay pattern.

5.8. E

When the capacitance in a circuit is raised while keeping other components such as resistance and inductance unchanged, the amplitude of voltage overshoot in the transient response is likely to decrease. This happens because an increased capacitance extends the circuit's time constant, resulting in a more gradual voltage shift during the transient period. As a result, the voltage overshoot is reduced.

6. Lab 2

6.1. Problem IV

$$g = 9.81, m = 5, l = 0.4, c = 0.7$$

- moment of inertia: $I = m\left(\frac{l}{2}\right)^2 = 0.2$
- undamped natural frequency:$\omega_n = \sqrt{\frac{mgl}{2I}} = 7.0036$
- damping coefficient:$\xi = \frac{c}{2*I*\omega_n}$
- damped natural frequency:$\omega_d = \omega_n\sqrt{1-\xi^2} = 6.7814$
- rise time:10% to 90%:$t_r = \frac{1}{\omega_d} \arctan\left(\frac{\sqrt{1-\xi^2}}{x_i}\right) = 0.1944$
- peaking time from application of step:$t_p = \frac{\pi}{\omega_d} = 0.4633$
- settling time based on $\pm 1\%$ tolerance:$t_s = \frac{3}{\xi\omega_n} = 1.7143$
- overshoot in %:$M_p = 100 \times e^{-\xi\omega_n t_p} = 44.4540$
- CLTF(Closed-Loop Transfer Function):

$$\frac{500}{s^2 + 3.5s + 549}$$

- sys(transfer function):

$$\frac{5}{s^2 + 3.5s + 54.05}$$

- Difference: Different from the open-loop transfer function that only contains forward loops, the closed-loop transfer function also contains the feedback, the output has an impact on the input. The closed-loop transfer function has lower gain and is more stable than the open-loop transfer function.

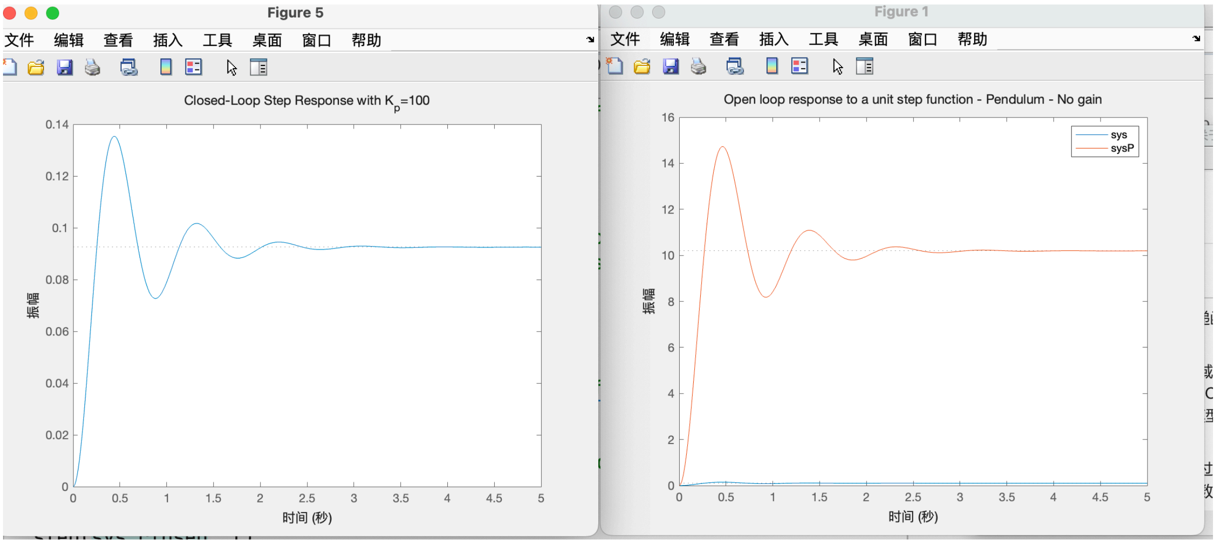

Figure 9: The open-loop and closed-loop transfer function

6.2. Calculation of the open-loop transfer function

$$\text{sys} = \text{tf}\left(\frac{1}{I}, 2\xi\omega_n\omega_n^2\right)$$

$$I = 0.2$$

$$2\xi\omega_n = 2 \cdot 0.25 \cdot 7.004 \approx 3.502$$

$$\omega_n^2 = (7.004)^2 \approx 49.056$$

$$\text{sys} = \frac{5}{s^2 + 3.502s + 49.056}$$

By transforming these frequency-domain equation to the time-domain equation:

$$\text{sys(t)} = \frac{5}{\omega_n^2}\left[1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \cdot \left(\cos(\omega_d t) + \frac{\xi}{\sqrt{1-\xi^2}}\sin(\omega_d t)\right)\right]$$

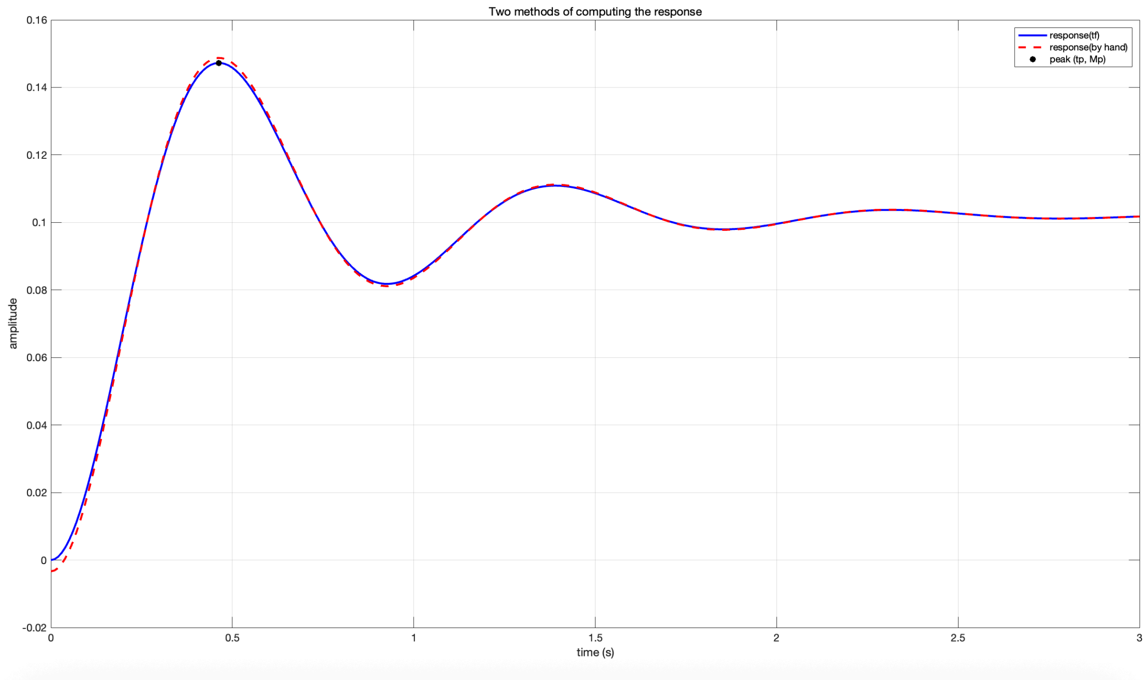

Figure 10: Response derived by two methods

The matlab’s result has lower peak value.

6.3. Poles and zeros

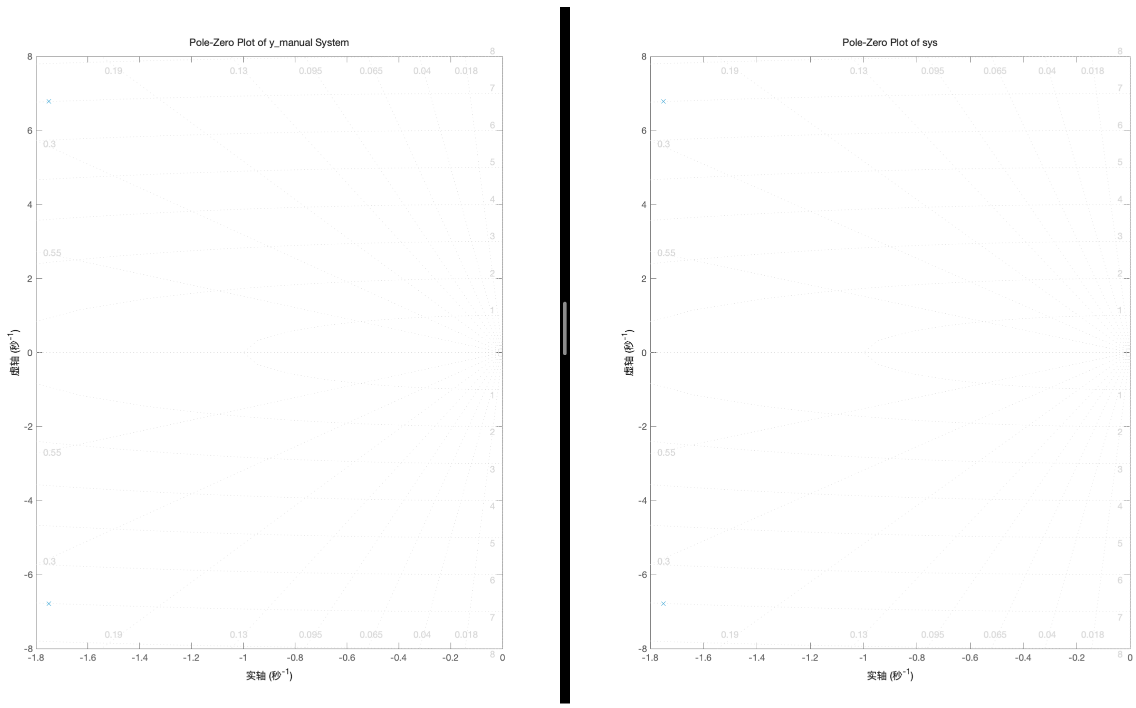

Figure 11:

6.4. Analysis of polse and zeros of two methods’ results

The poles and zeros from the two methods are the same

6.5. 12

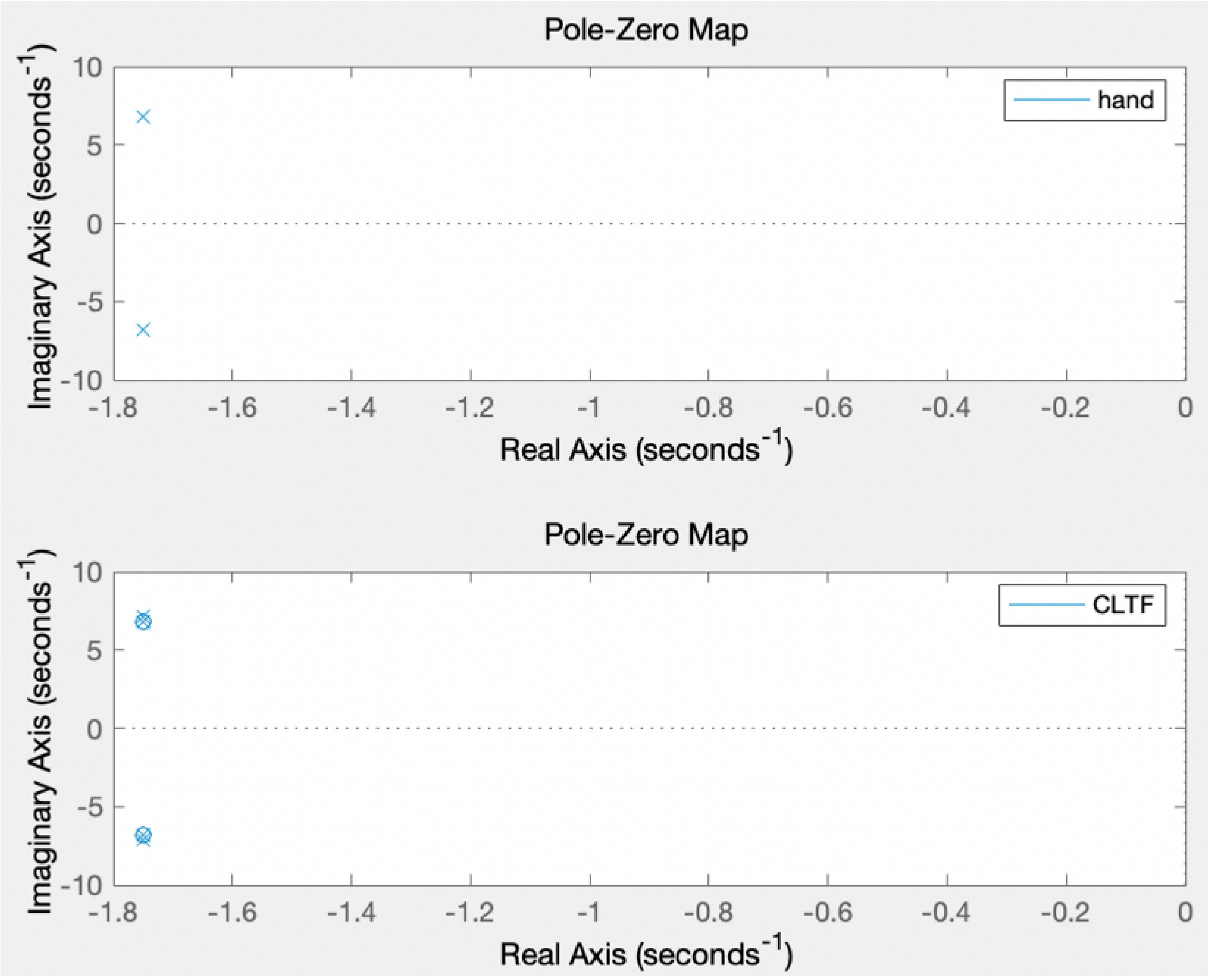

Figure 12: PZ plot

Because of the zero-pole cancellation, two zeros and two poles are missing.

6.6. 13

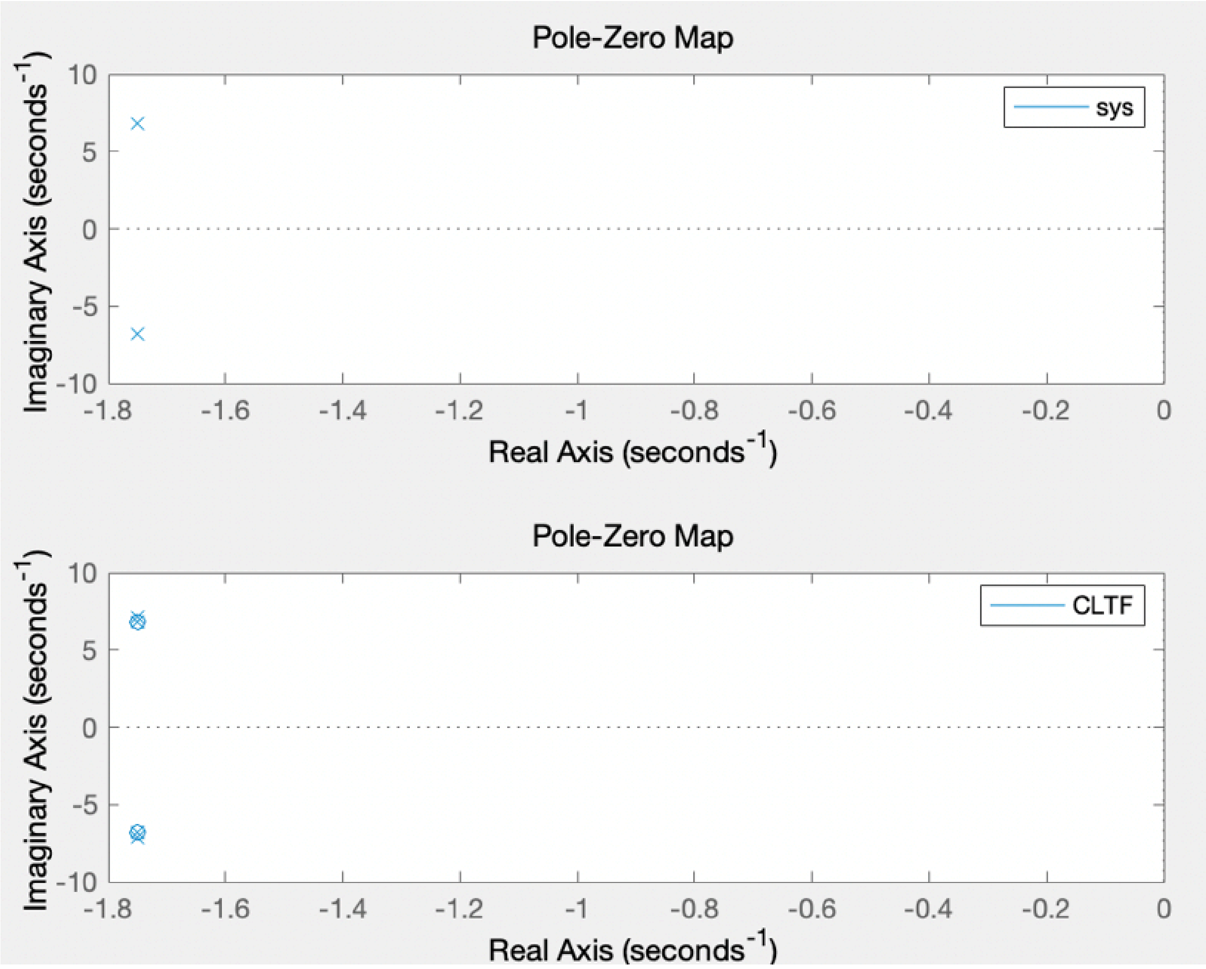

Figure 13: PZ plot

The closed-loop system has two additional poles with the same real part but larger imaginary part than the open-loop system. This occurs because the denominator of the CLTF incorporates both the denominator and numerator of the system.

6.7. 14

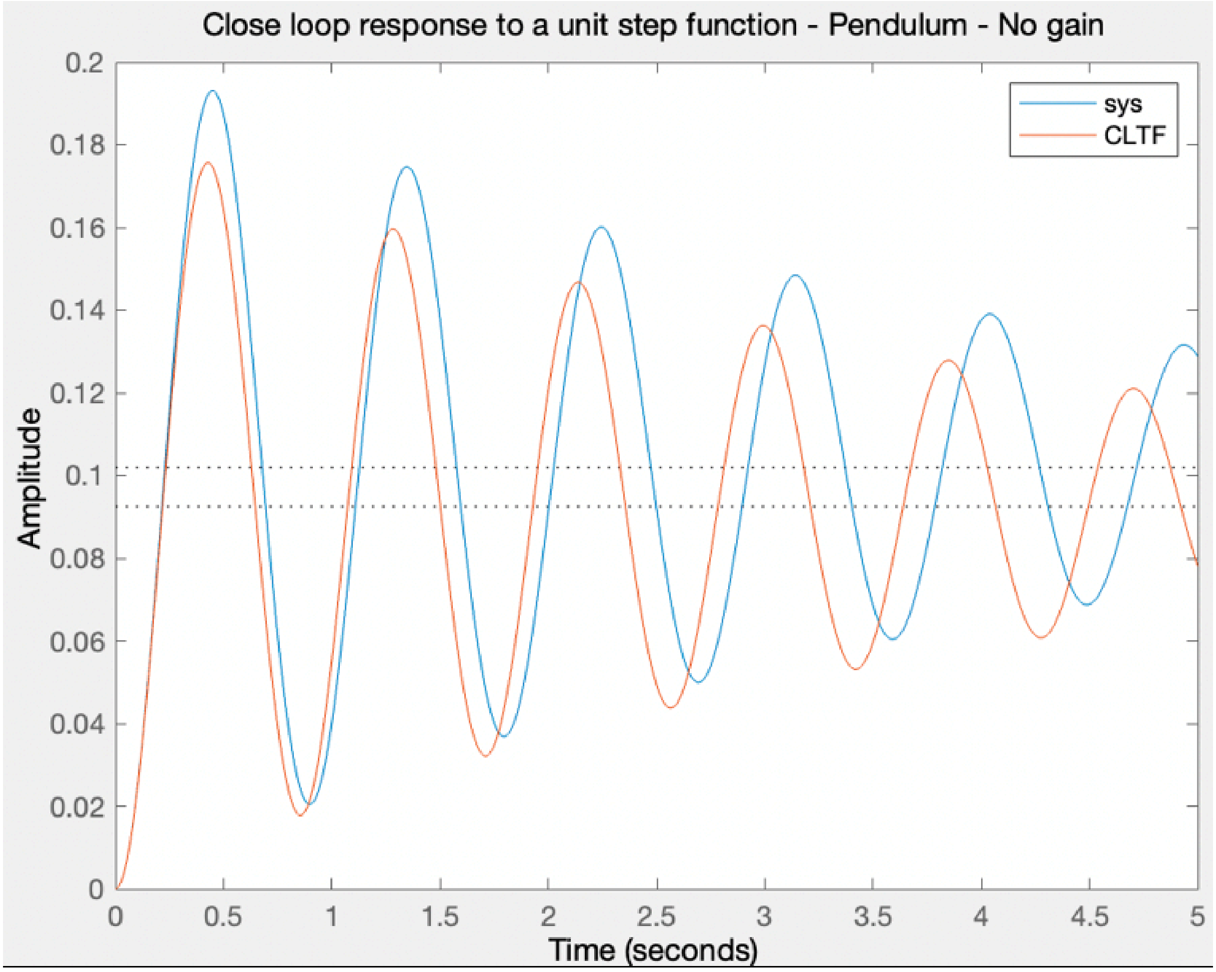

Figure 14: Closed-Loop unit step Response, c=0.08

If the poles are all pure-real, ther system is critically damped or overdamped. Thus, poles should be none-real values. $b^2 - 4ac < 0$,

$$c^2 < \frac{2mgl + ml^2 + 4}{8}$$

7. Lab 3

7.1. Problem X

- Question 4

As a first experiment, assume the controller transfer function above. By setting Kd and Ki to zero and influencing Kp, note what happens to the output response. Why does the output behave like this?

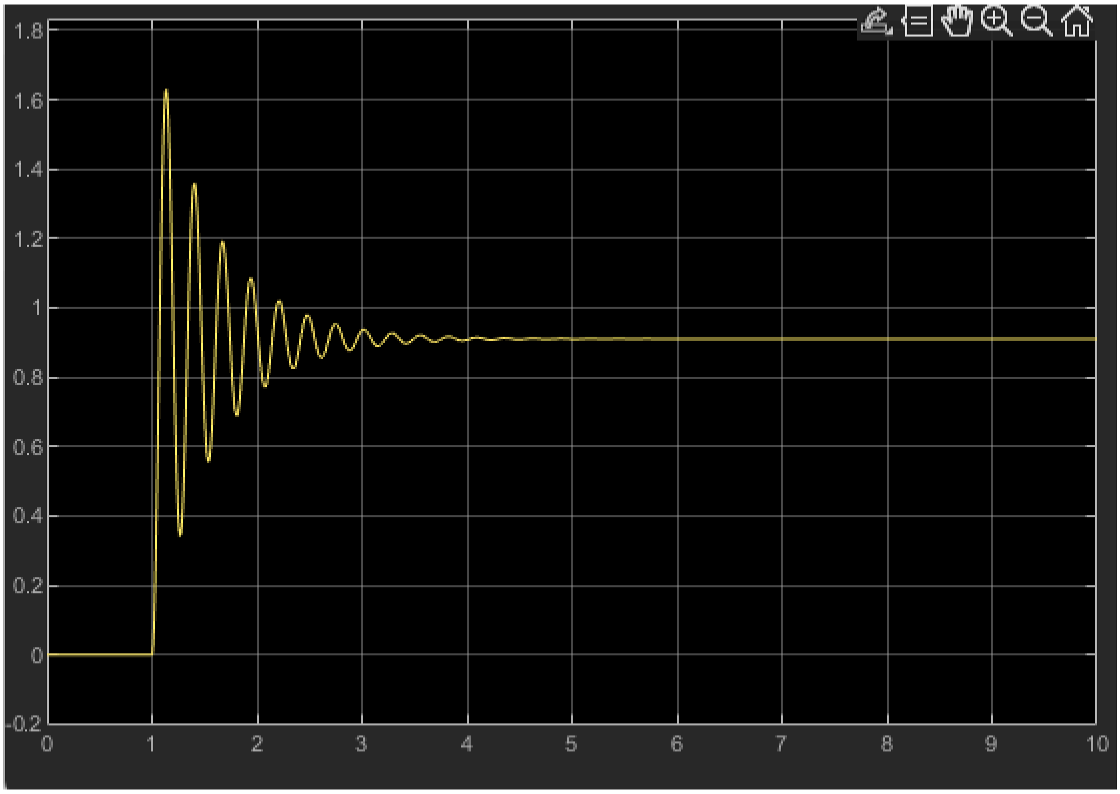

Figure 15: K_p=100

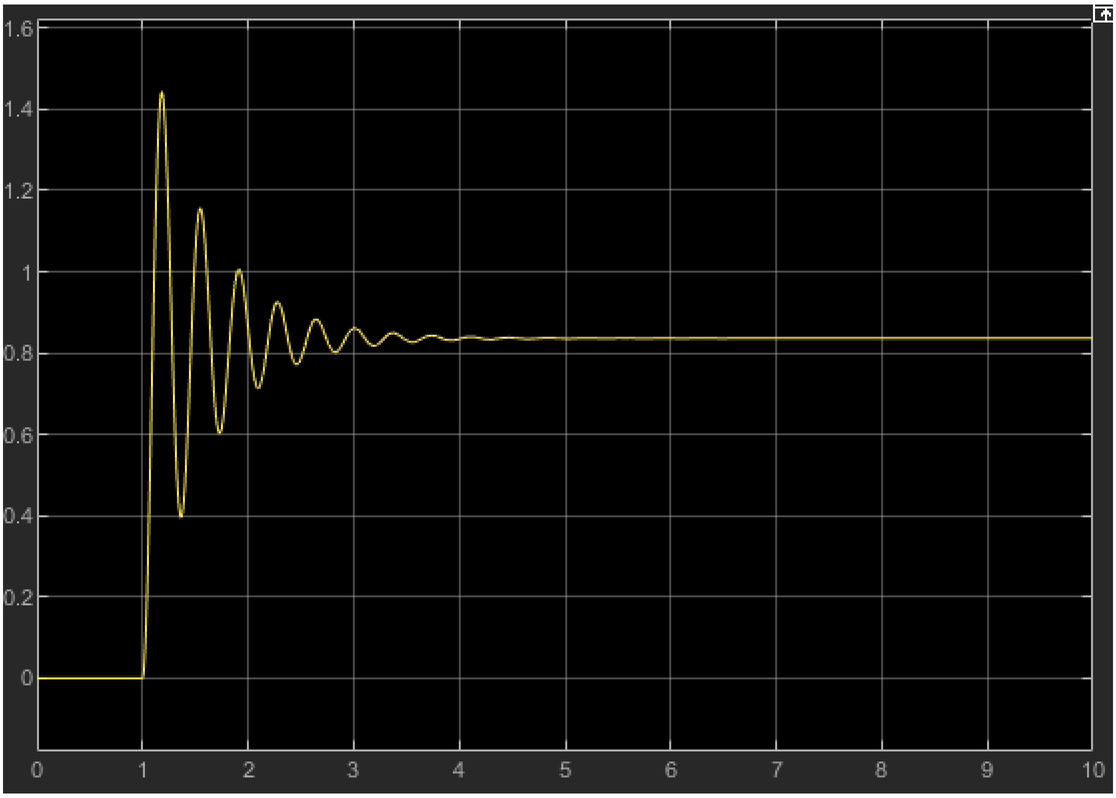

Figure 16: K_p=50

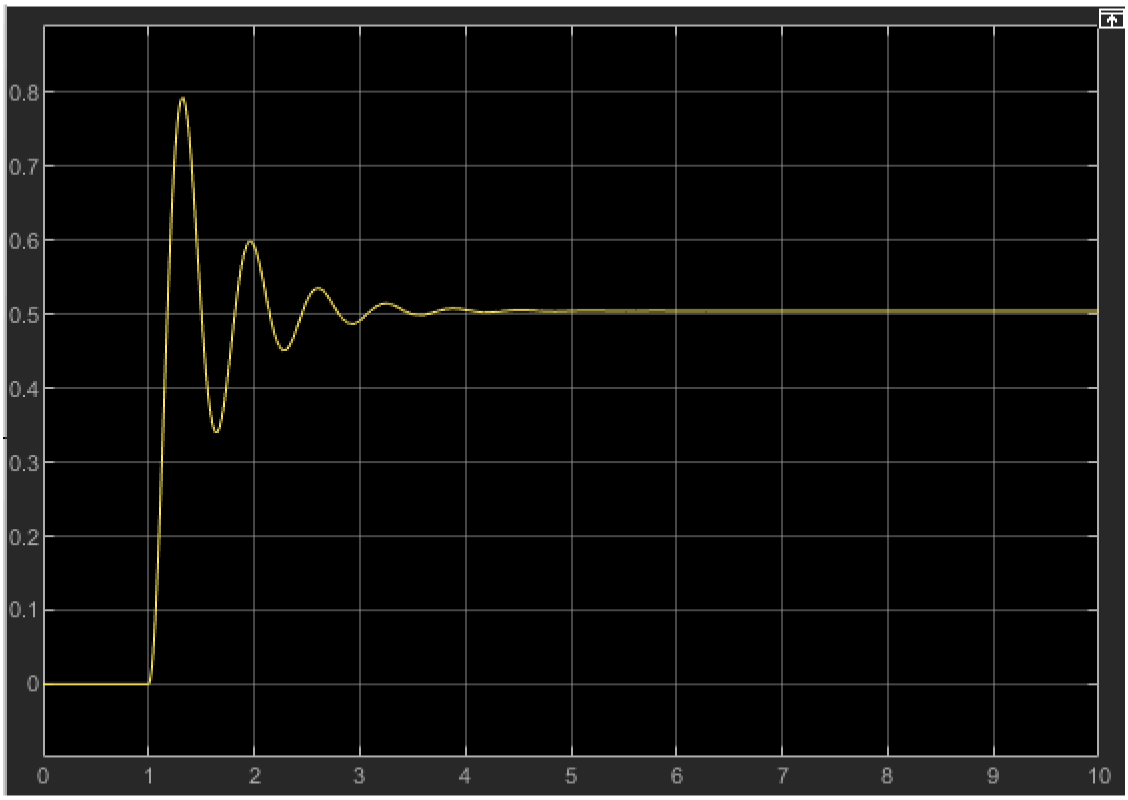

Figure 17: K_p=10

As K_p increases, the setting time decreases, the steady state error increases, and it may oscillate.

- Question 5

You can create a script that will plot the poles to allow you to visualise easily where the poles move to depending on the system gain configuration. So from this what do you conclude is the reason for the output behaving as it does with Kp=1, Ki and Kd=0?

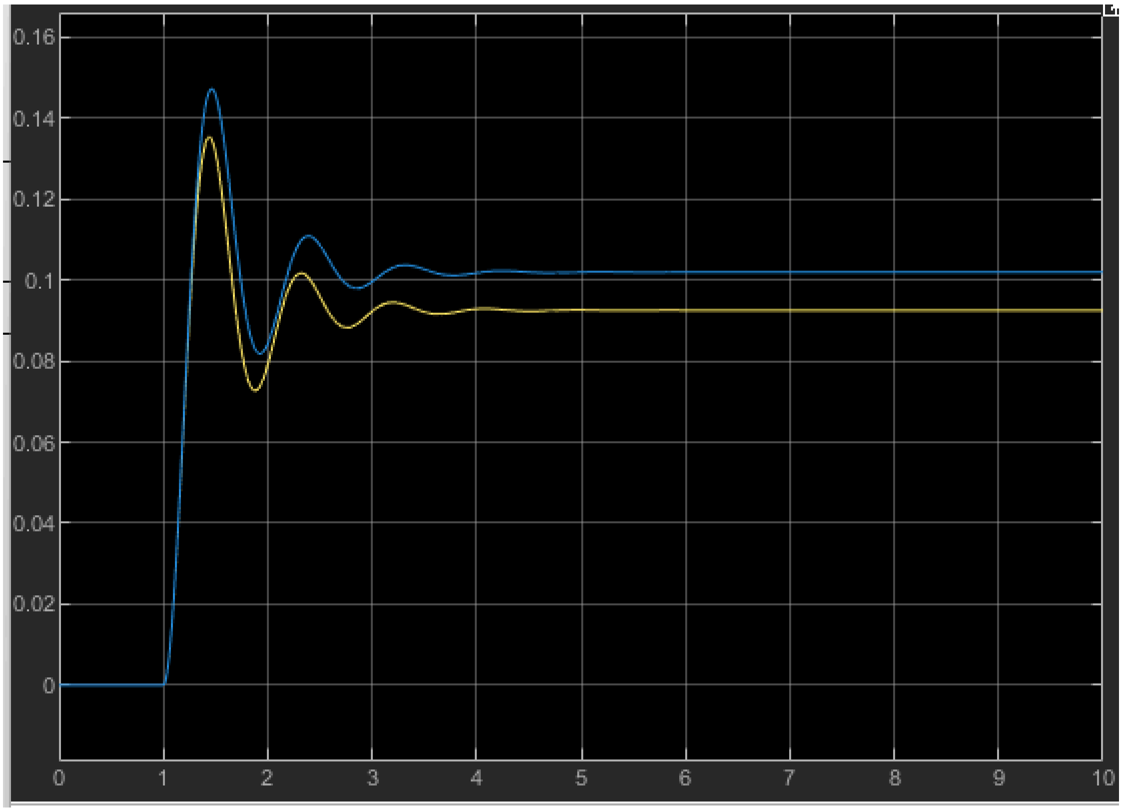

Figure 18: $K_p = 1, K_i = K_d = 0$

$$e_{\text{ss}} = \frac{1}{1+K_p}$$

$K_p = 1$, the steady error is 0.5.

- Question 6

Now in addition to Kp=1, add derivative gain, Kd=1 and note the change in behaviour. Explain this by again looking at the poles and zeros in the system

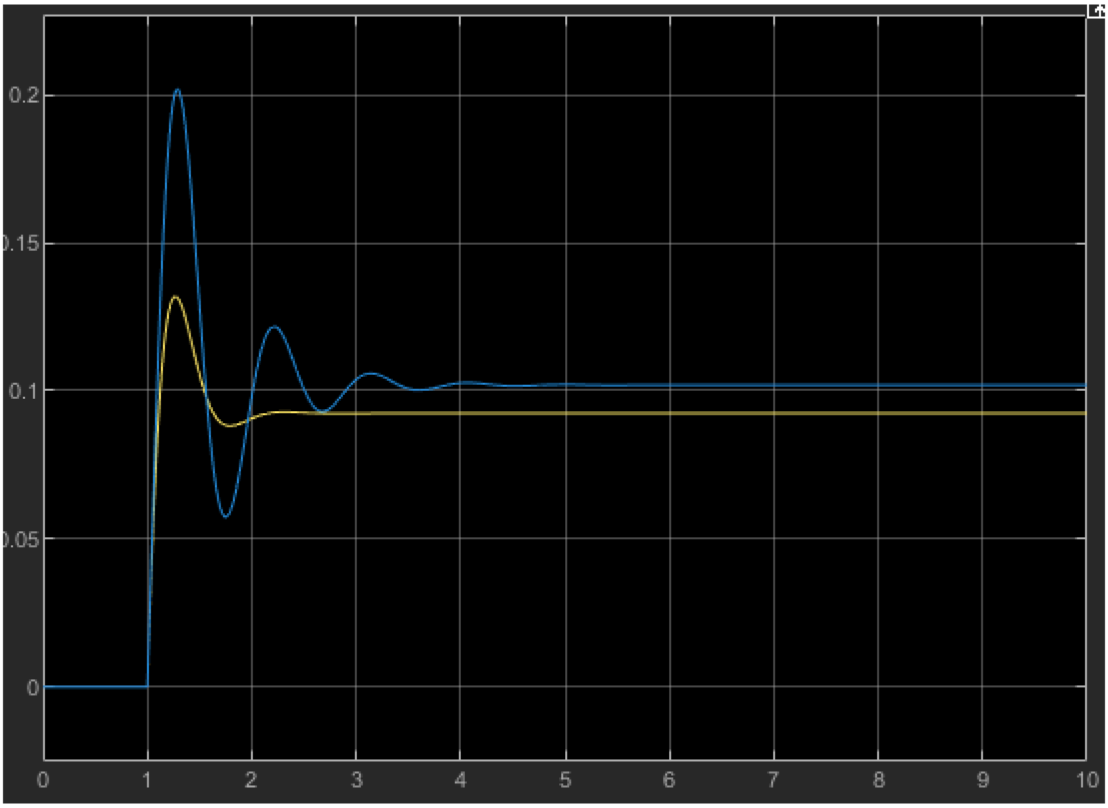

Figure 19: $K_d = 1$

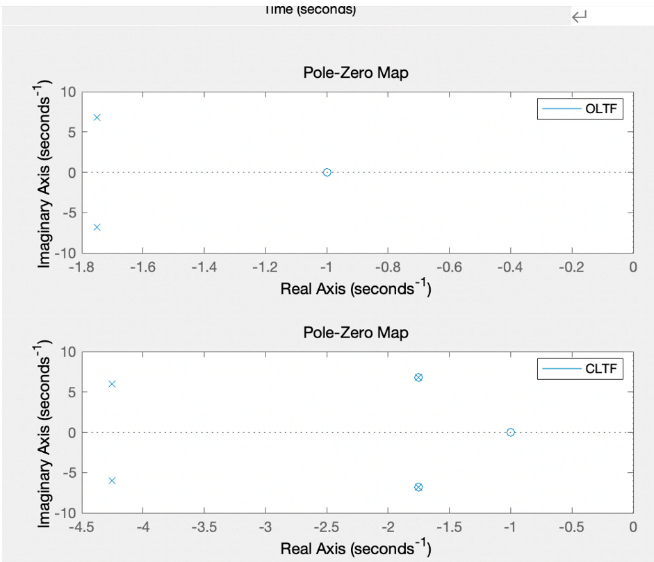

Figure 20: Zeors-poles graph

$$\text{CLTF} = \frac{5s^3 + 22.5s^2 + 262.7s + 245.2}{s^4 + 12s^3 + 132.8s^2 + 606.1s + 2651}$$

- Problem 7

$$\text{CLTF} = \frac{5s^5 + 22.5s^4 + 267.7s^3 + 262.7s^2 + 245.2s}{s^6 + 12s^5 + 132.8s^4 + 611.1s^3 + 2669s^2 + 245.2s}$$

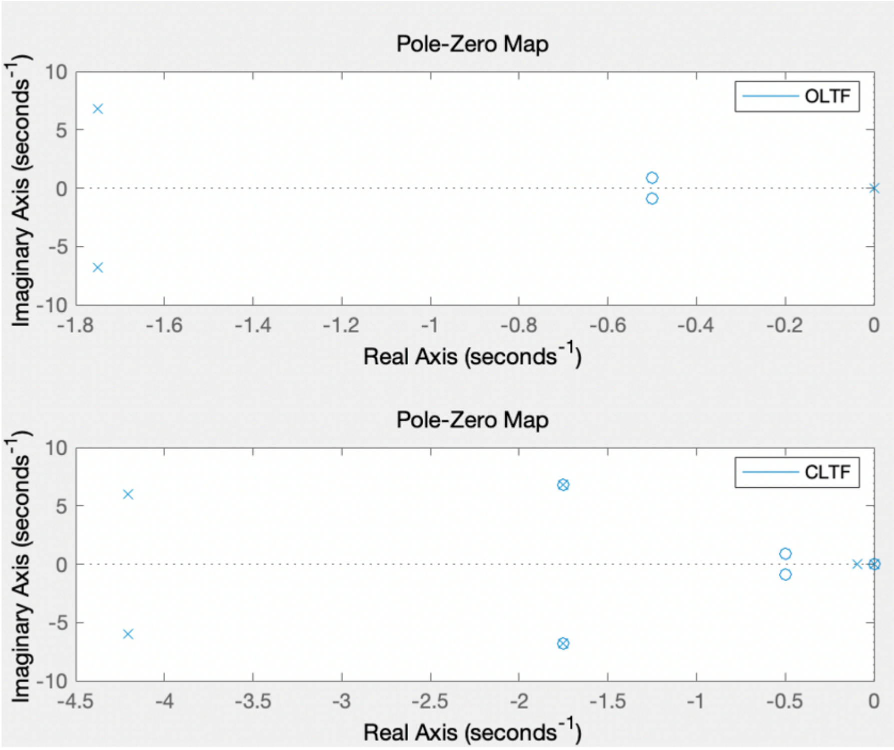

Figure 21: ZP graph of Problem 7

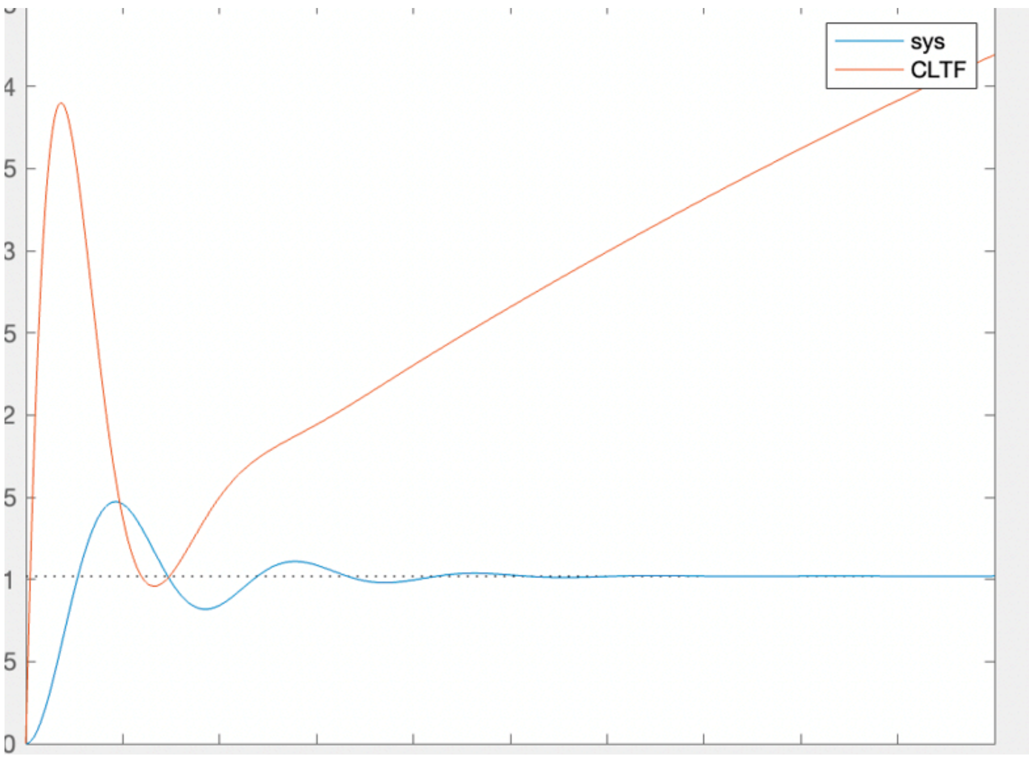

Figure 22: Response of Problem 7

- Problem 8

The proportional gain, Kp, is always set to a non-zero value and influences both the rise time and steady-state error. However, it cannot eliminate steady-state error entirely on its own.

The derivative gain, Kd, responds to input signal changes, helping the system reach steady state faster.

The integral gain, Ki, removes steady-state error by introducing a pole at DC.

- Problem 9 and Problem 10

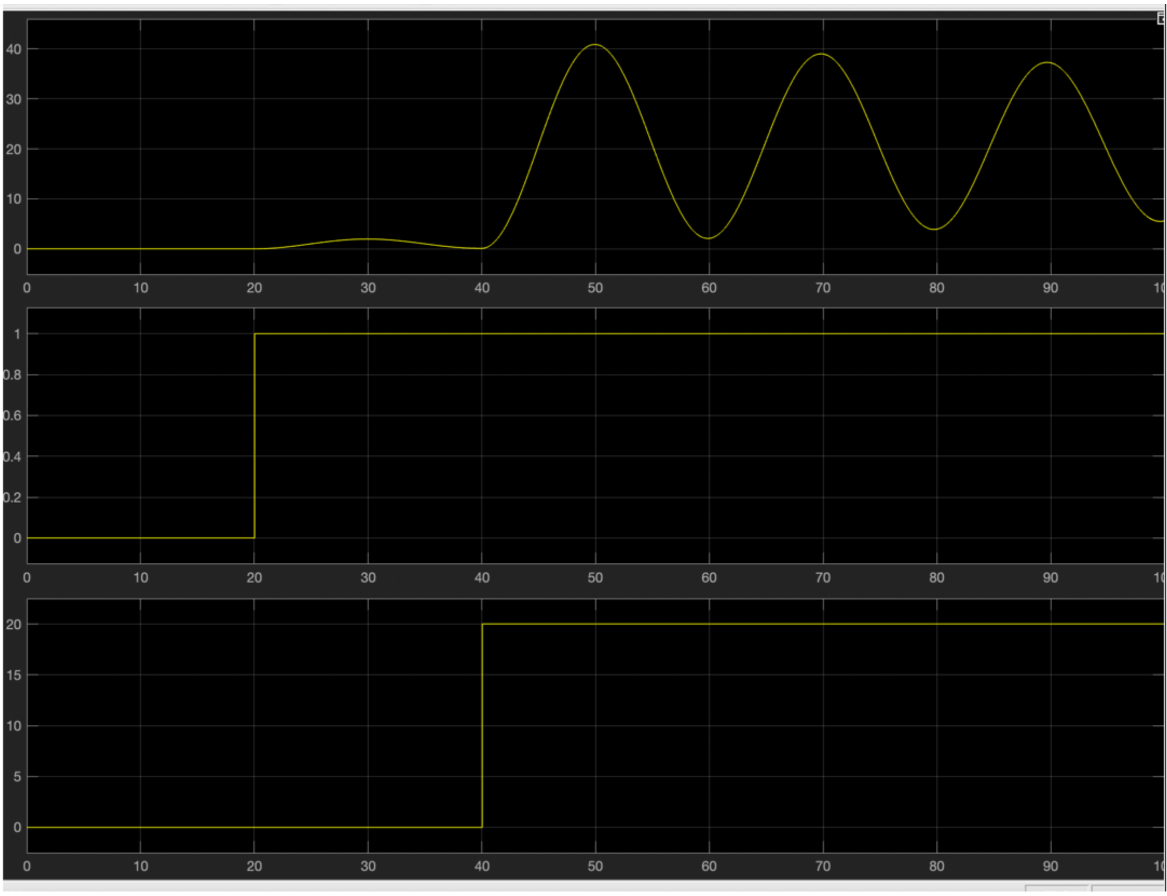

Figure 23: $K_p = 1, K_d = K_i = 0$

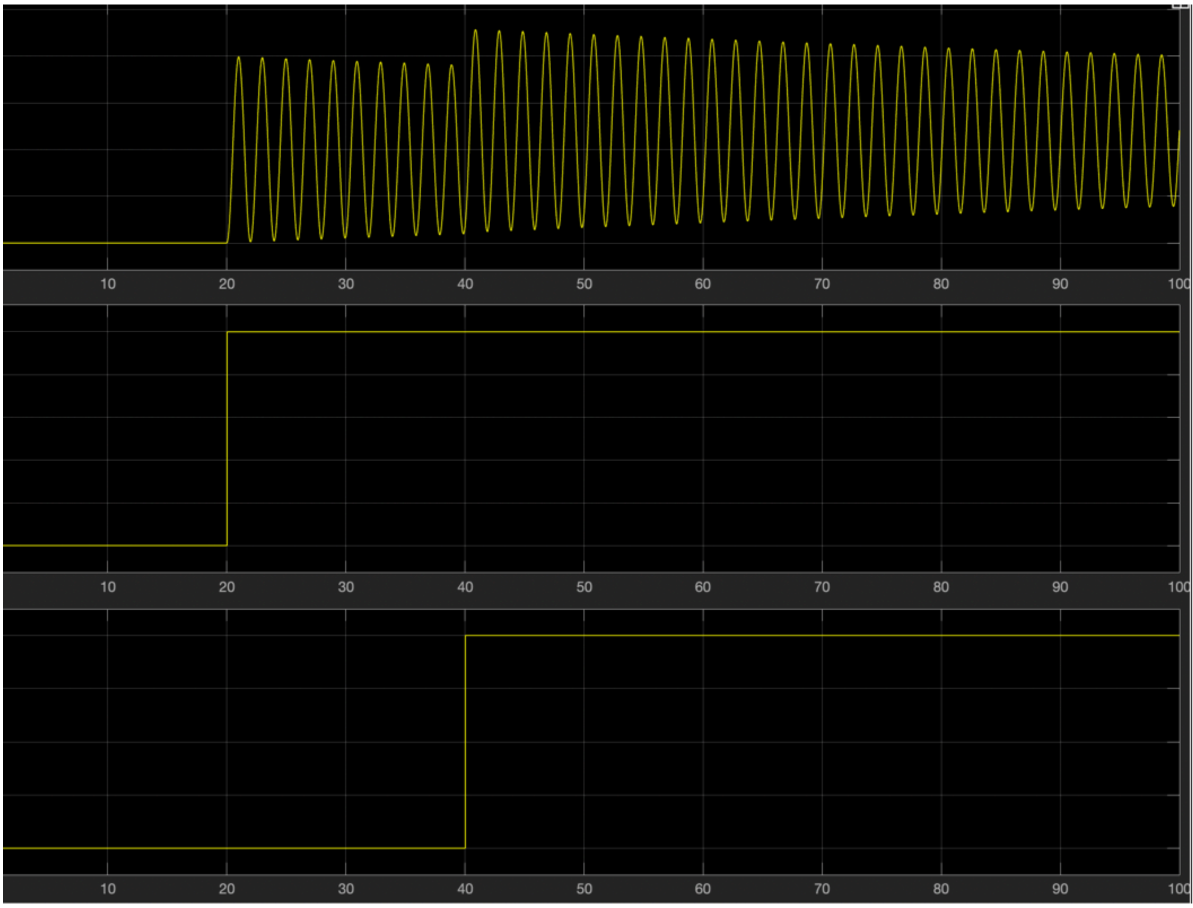

Figure 24: $K_p = 100$

Plant transfer function:

$$\frac{K_p}{10s^2 + 0.1s + K_p}$$

- Problem 11
  - Part I

$$\frac{C(s)}{R(s)} = \frac{K_p + sK_d}{10s^2 + (0.1 + K_d)s + K_p}$$

$$\omega_n = \sqrt{\frac{K_p}{10}}, \xi = \frac{0.1 + K_d}{20} \times \sqrt{\frac{10}{K_p}}$$

$$c(\infty) = \lim_{s \to 0} s \times \frac{1}{s} \times \frac{K_p + sK_d}{10s^2 + (0.1 + K_d)s + K_p} = 1$$

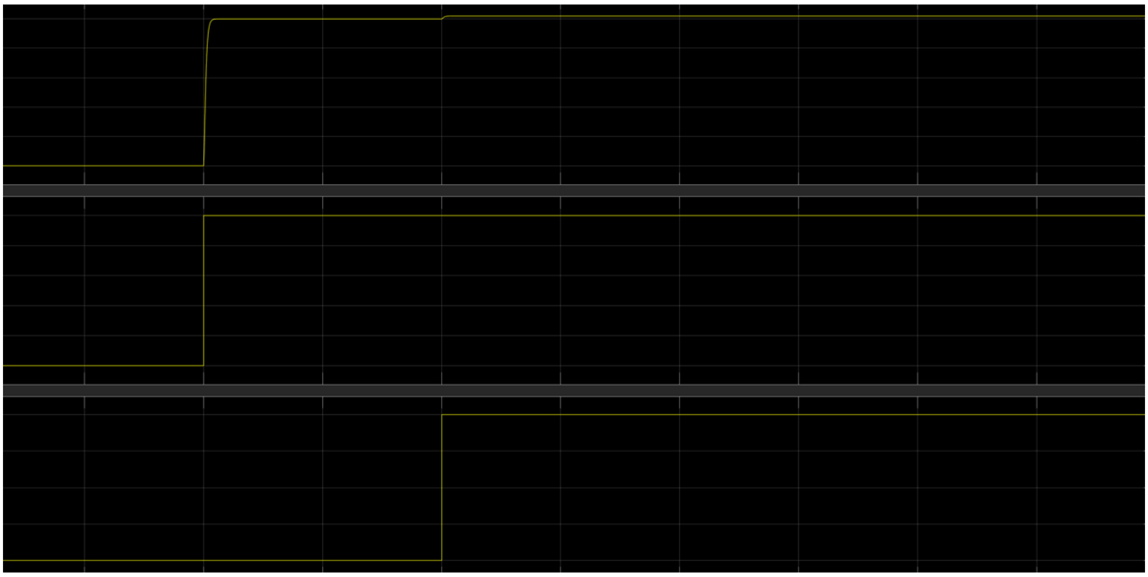

Figure 25: Part I

- Part II
  Characteristic equation:$s^3 + 20s^2 + 100s + \frac{K_i}{10} = 0$, by using Routh's stability criterion:$K_i < 20000$, For $K_i = 19050$:

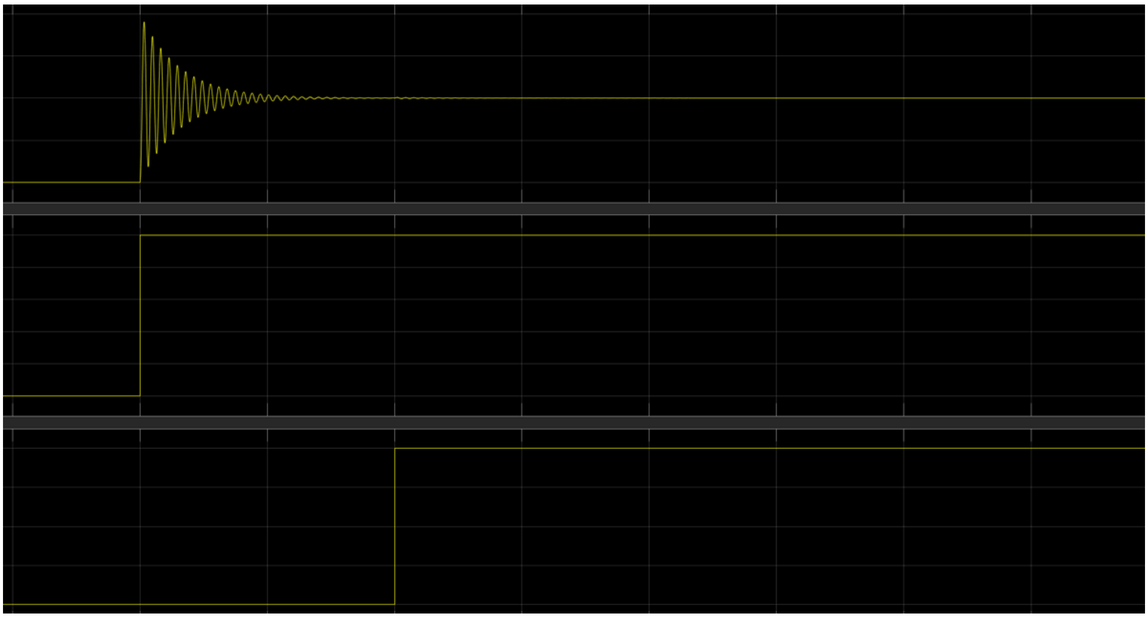

Figure 26: Part II

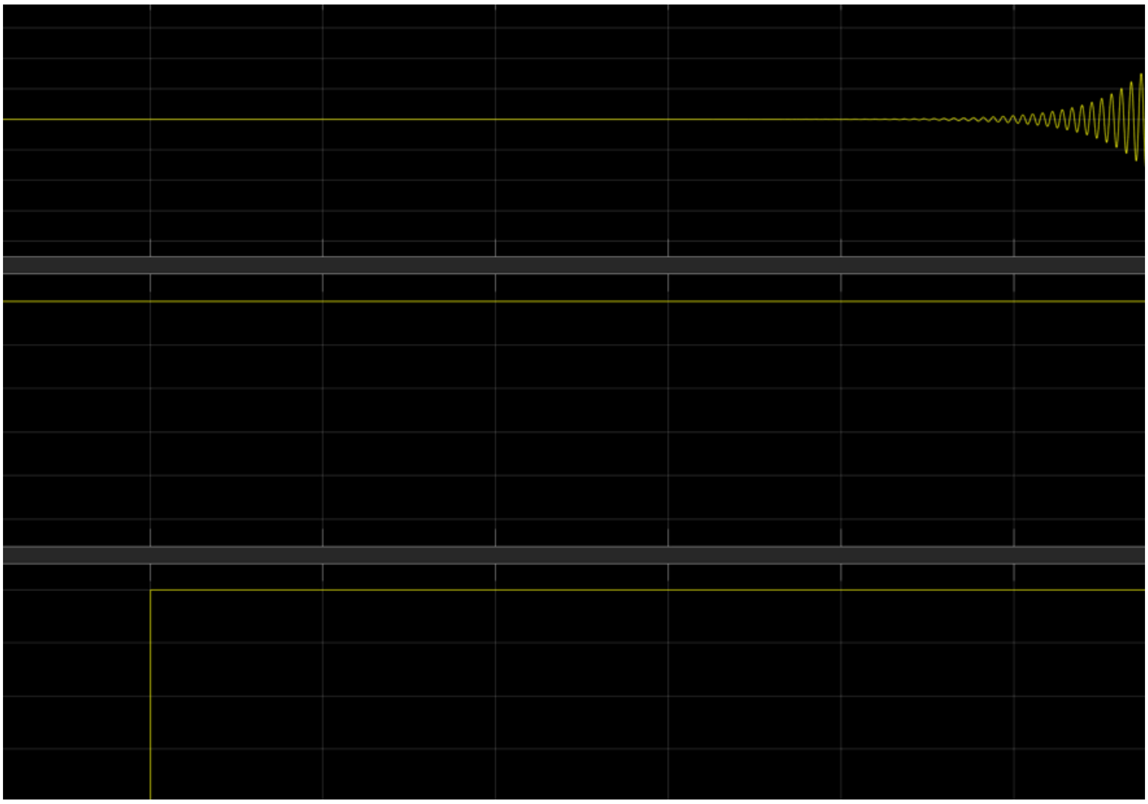

Figure 27: Part II, $K_i = 26000$

- Does the output respond faster or slower as you increase ω_n? K_p increases $\Rightarrow \omega_n$ increases $\Rightarrow$ faster response.

8. Lab 4

A PID controller (Proportional-Integral-Derivative controller) is a widely used feedback control mechanism that continuously calculates an error value as the difference between a desired setpoint and a measured process variable. It applies a correction based on three terms:

1. Proportional (P)-Responds to the current error
2. Integral (I)-Eliminate steady-state offset by addressing accumulated past error
3. Derivate (D)-Improve stability by prediciting the future error

8.1. Exercise I

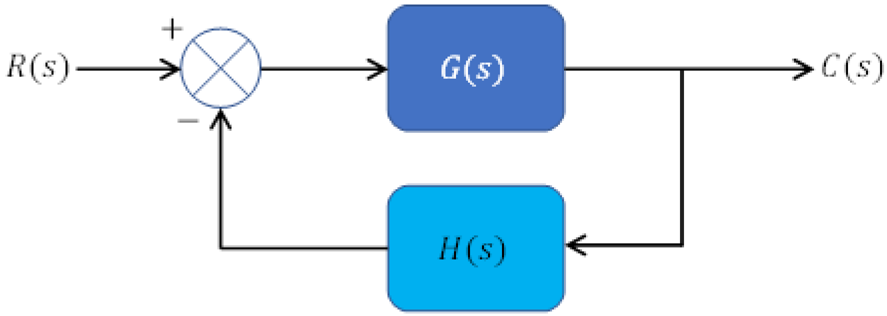

Figure 28: System for exercise I

$$G(s) = \frac{(s+1)(s+2)}{s^2(s+3)(s+4)(s+10)}$$

$$H(s) = 1$$

- OLTF

$$\text{OLTF} = G(s)H(s) = \frac{(s+1)(s+2)}{s^2(s+3)(s+4)(s+10)}$$

- CLTF

$$\text{CLTF} = \frac{G(s)}{1+G(s)H(s)}$$

$$\text{CLTF} = \frac{(s+1)(s+2)}{s^2(s+3)(s+4)(s+10)+(s+1)(s+2)}$$

• Root locus

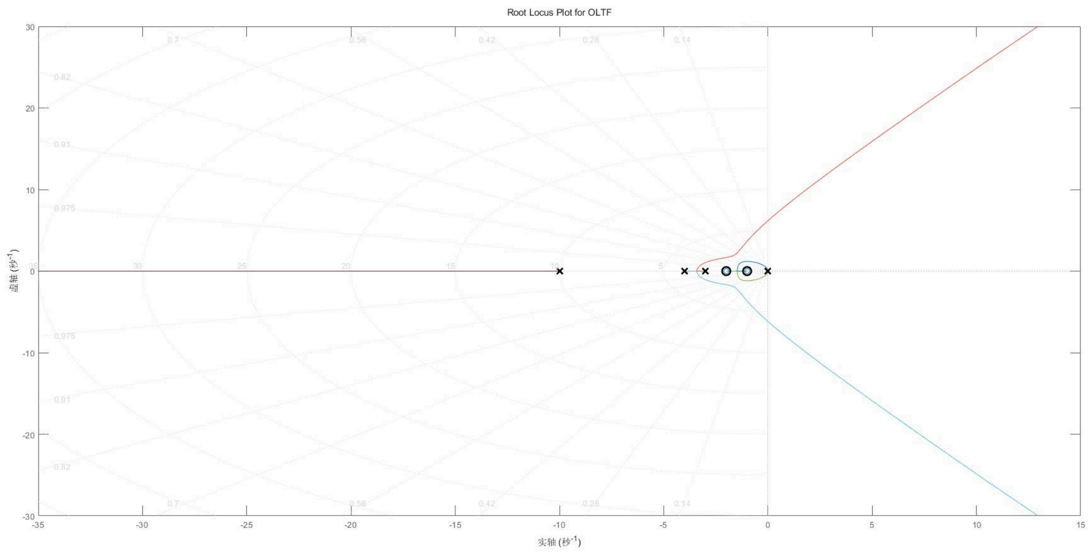

Figure 29: Root locus

• Bode magnitude and phase plots

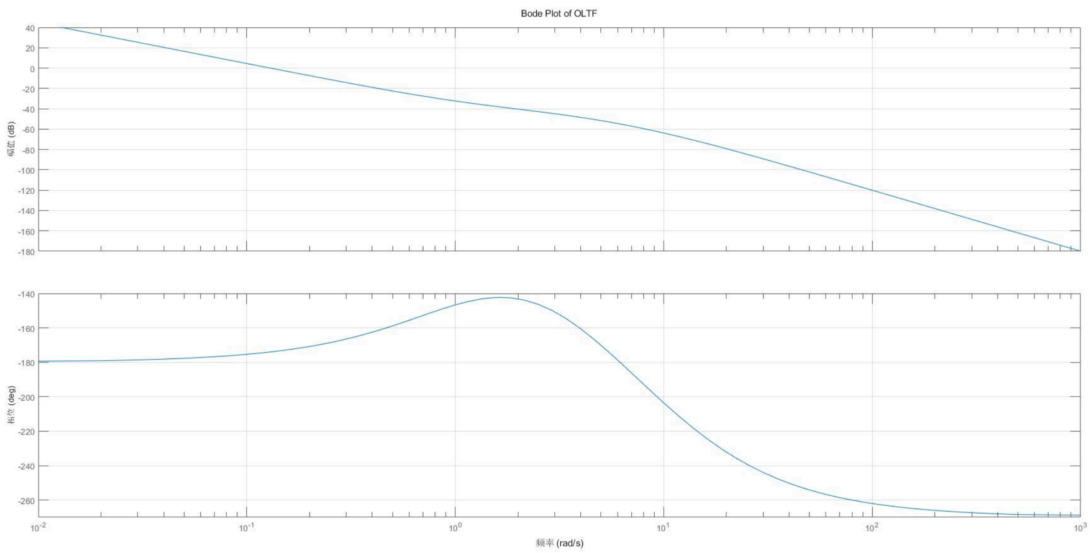

Figure 30: Bode Plot of OLTF

- Nyquist plot of OLTF

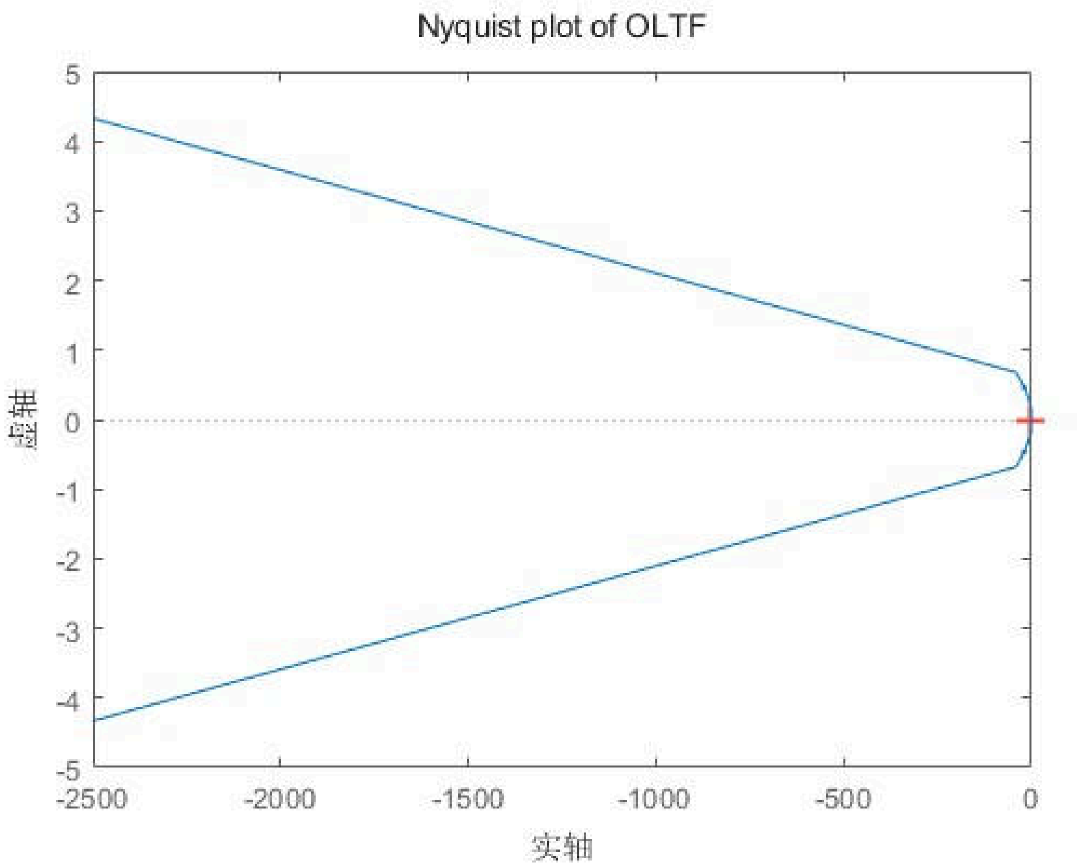

Figure 31: Nyquist plot of OLTF

- Some parameters

```
num = [1, 3, 2];
den = poly([0, 0, -3, -4, -10]);
G = tf(num, den);
[GM, PM, GMFrequency, PMFrequency] = margin(G);
DM = PM / (180 * pi * PMFrequency);
fprintf('Gain Margin (GM): %.4f (linear), %.4f dB\n', GM,
20*log10(GM));
fprintf('GM Frequency: %.4f rad/s\n', GMFrequency);
fprintf('Phase Margin (PM): %.4f degrees\n', PM);
fprintf('PM Frequency: %.4f rad/s\n', PMFrequency);
fprintf('Delay Margin (DM): %.4f seconds\n', DM);
fprintf('DM Frequency: %.4f rad/s\n', PMFrequency);
if PM > 0 && GM > 1
    fprintf('The system is stable.\n');
else
```

```
    fprintf('The system is unstable or marginally stable.\n');
end
```

Gain Margin (GM): 557.5962 (linear), 54.9264 dB

GM Frequency: 6.1756 rad/s

Phase Margin (PM): 6.0247 degrees

PM Frequency: 0.1297 rad/s

Delay Margin (DM): 0.0821 seconds

DM Frequency: 0.1297 rad/s

The system is stable.

9. Conclusion

In the course of dynamic and control, we learnt a lot about how to analyze and design a dynamic system based on examples on typical dynamic systems.

In the lab sessions, I analyzed some systems with MATLAB based on mathematical principles such as z ans s transform and differential equations. Analysis of a dynamic system is the process that convert the physical model into the mathematical model. With computer and mathematical algorithms, these system can be analyzed and simulated making them easier to be designed and optimized.