

**Kathmandu University**  
**Department of Computer Science and Engineering**  
**Dhulikhel, Kavrepalanchowk**



**A Lab Report**  
**on**  
**“Control System”**  
**Lab No: I & II**

**[Code No: COEG304]**

**Submitted by:**

Royas Shakya

CE 2022

**Roll No: 55**

**Submitted to:**

Ms. Supriya Pandeya

Department of Electrical and Electronics. Engineering

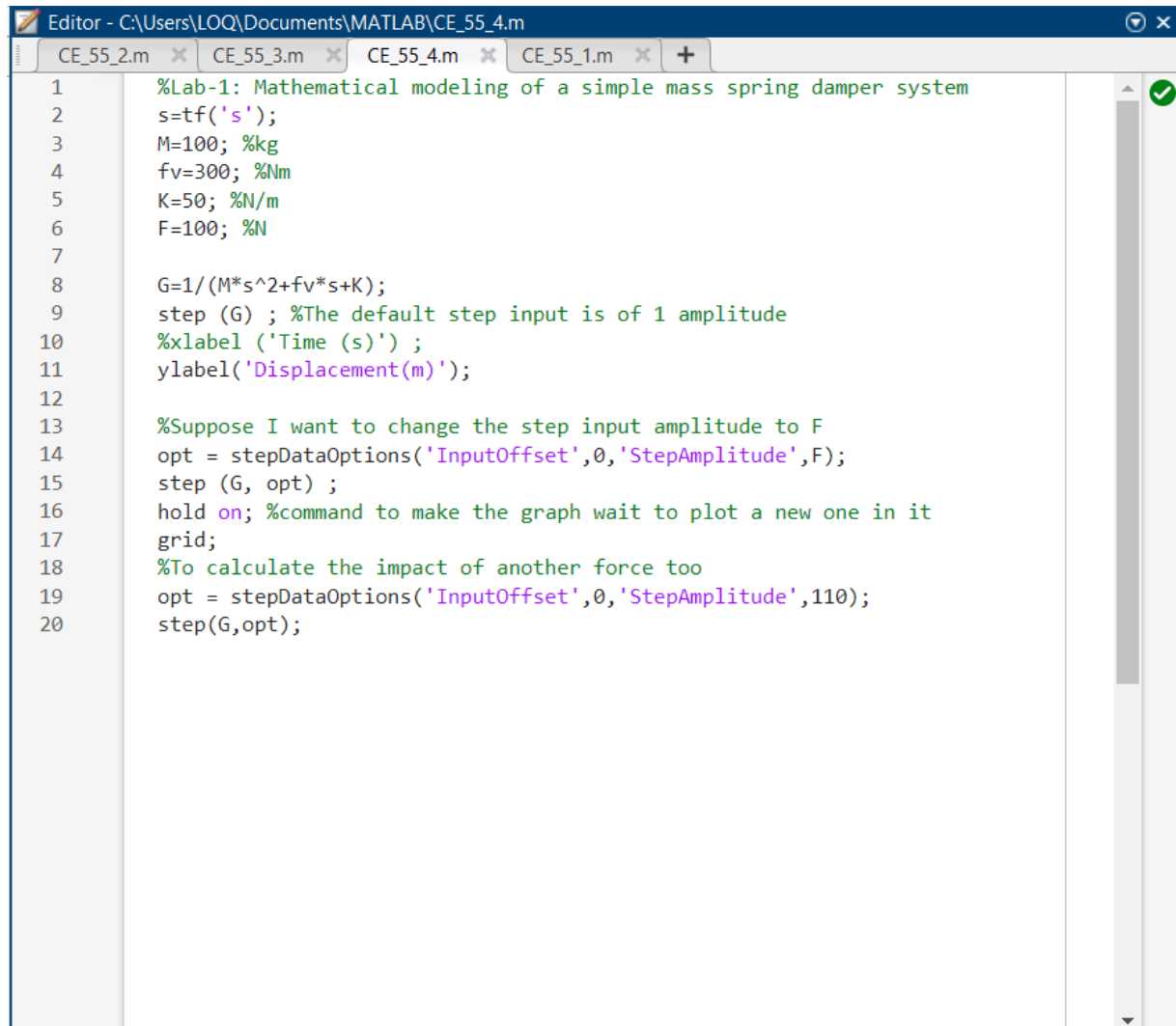
**Submission Date: 19/06/2025**

# Title: To learn to use MATLAB Models and SIMULINK

## Theory:

Mathematical models are extremely important to understand the behaviour of system to be controlled, MATLAB serves as an excellent software tool to develop and analyze these models.

## Task 1: Step Input Response

The image shows a MATLAB Editor window with the title bar 'Editor - C:\Users\LOQ\Documents\MATLAB\CE\_55\_4.m'. The window contains a script with 20 lines of MATLAB code. The code defines a transfer function 's' for a mass spring damper system with parameters M=100, fv=300, K=50, and F=100. It then uses the 'step' function to plot the response, with 'xlabel' and 'ylabel' labels. The code also includes comments and options for changing the step input amplitude and holding the plot. The script is saved, as indicated by the green checkmark in the top right corner of the editor window.

```
1 %Lab-1: Mathematical modeling of a simple mass spring damper system
2 s=tf('s');
3 M=100; %kg
4 fv=300; %Nm
5 K=50; %N/m
6 F=100; %N
7
8 G=1/(M*s^2+fv*s+K);
9 step (G) ; %The default step input is of 1 amplitude
10 xlabel ('Time (s)') ;
11 ylabel('Displacement(m)');
12
13 %Suppose I want to change the step input amplitude to F
14 opt = stepDataOptions('InputOffset',0,'StepAmplitude',F);
15 step (G, opt) ;
16 hold on; %command to make the graph wait to plot a new one in it
17 grid;
18 %To calculate the impact of another force too
19 opt = stepDataOptions('InputOffset',0,'StepAmplitude',110);
20 step(G,opt);
```

Figure 1: Step Input Response Code

This model represented a standard second-order system using a transfer function block. It was tested with a step input to observe time-domain characteristics such as rise time, overshoot, and settling behavior. This helped us understand how a second-order system responds to a sudden and sustained input.

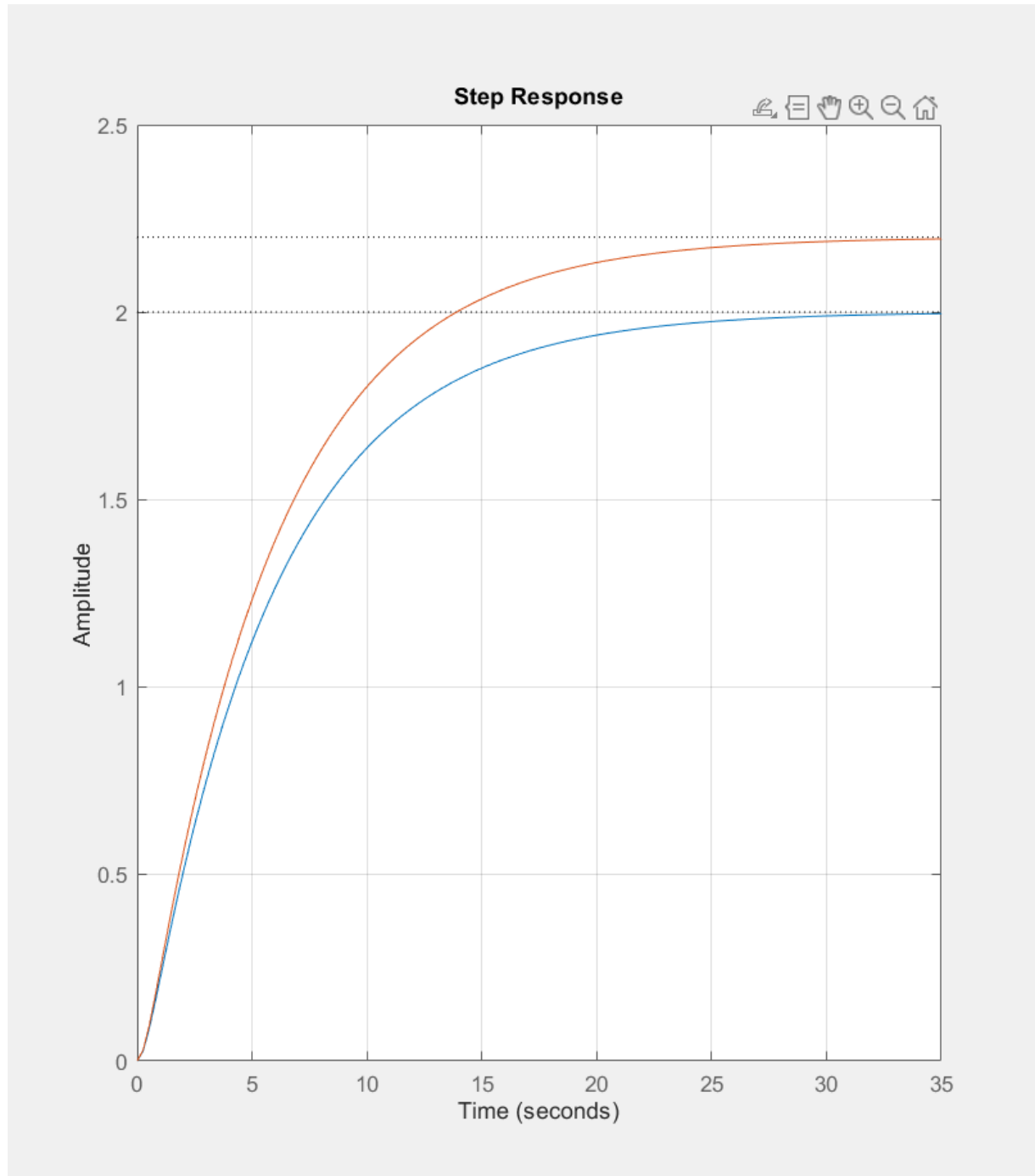
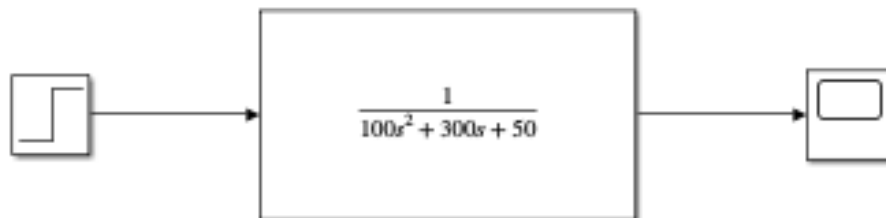
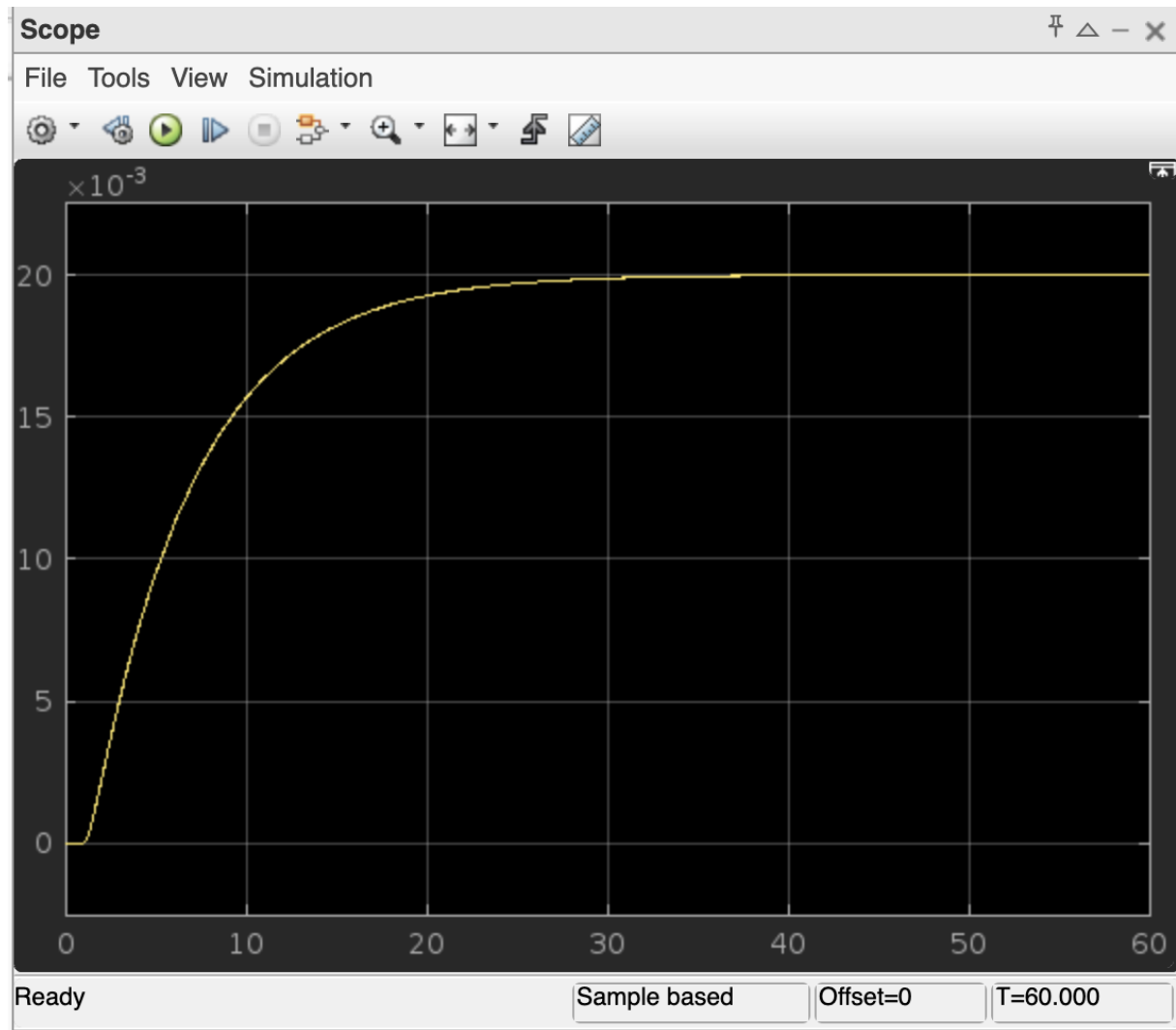


Figure 2: Step Input Response Output

## Task 2: SIMULINK Execution



*Figure 3: SIMULINK model*



*Figure 4: SIMULINK response*

The second model used the same second-order system but with an **impulse input**, simulated using a short, high-amplitude pulse. This input represents a brief disturbance, allowing us to observe the system's **natural response** without continuous forcing. The output revealed key properties like damping and oscillation, helping us understand how the system reacts and stabilizes after a sudden shock.

## Conclusion

This experiment helped us understand the dynamic behavior of control systems by modeling and simulating their responses in Simulink. Through step and impulse tests, we observed how parameters like damping and natural frequency influence system performance. The use of Simulink helped us to understand theoretical concepts and highlighted the value of simulation in control system analysis.

**Kathmandu University**  
**Department of Computer Science and Engineering**  
**Dhulikhel, Kavrepalanchowk**



**A Lab Report**  
**on**  
**“Control System”**  
**Lab No: I & II**

**[Code No: COEG304]**

**Submitted by:**

Royas Shakya

CE 2022

**Roll No: 55**

**Submitted to:**

Ms. Supriya Pandeya

Department of Electrical and Electronics. Engineering

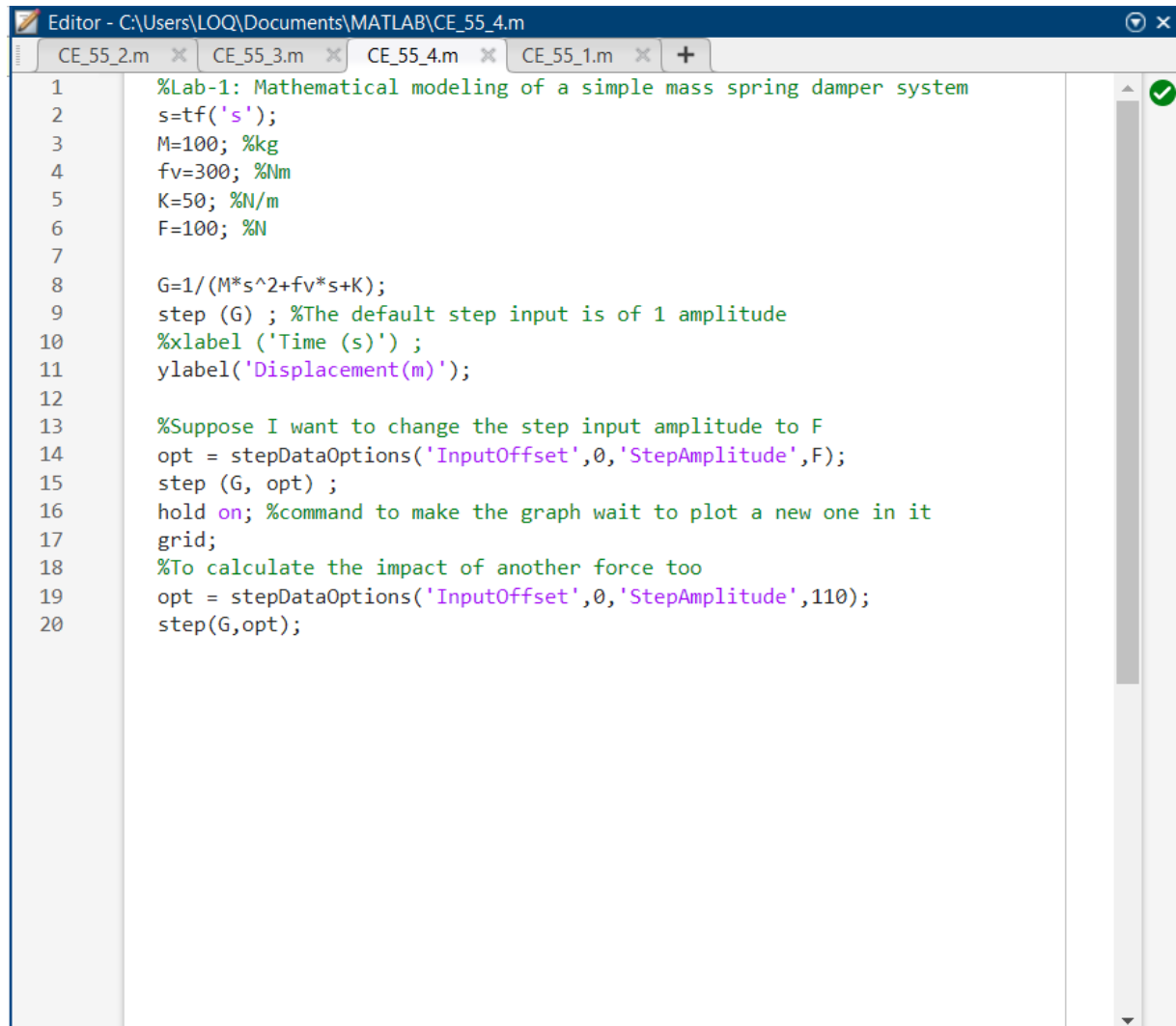
**Submission Date: 19/06/2025**

# Title: To learn to use MATLAB Models and SIMULINK

## Theory:

Mathematical models are extremely important to understand the behaviour of system to be controlled, MATLAB serves as an excellent software tool to develop and analyze these models.

## Task 1: Step Input Response

The image shows a MATLAB Editor window with the title bar 'Editor - C:\Users\LOQ\Documents\MATLAB\CE\_55\_4.m'. There are four tabs open: 'CE\_55\_2.m', 'CE\_55\_3.m', 'CE\_55\_4.m', and 'CE\_55\_1.m'. The active tab is 'CE\_55\_4.m', which contains the following MATLAB code:

```
1 %Lab-1: Mathematical modeling of a simple mass spring damper system
2 s=tf('s');
3 M=100; %kg
4 fv=300; %Nm
5 K=50; %N/m
6 F=100; %N
7
8 G=1/(M*s^2+fv*s+K);
9 step (G) ; %The default step input is of 1 amplitude
10 %xlabel ('Time (s)') ;
11 ylabel('Displacement(m)');
12
13 %Suppose I want to change the step input amplitude to F
14 opt = stepDataOptions('InputOffset',0,'StepAmplitude',F);
15 step (G, opt) ;
16 hold on; %command to make the graph wait to plot a new one in it
17 grid;
18 %To calculate the impact of another force too
19 opt = stepDataOptions('InputOffset',0,'StepAmplitude',110);
20 step(G,opt);
```

The code is color-coded: comments are green, function names like 'tf', 'step', and 'hold' are blue, and variable names and numerical values are black. The window has a scrollbar on the right and a green checkmark icon in the top right corner.

Figure 1: Step Input Response Code

This model represented a standard second-order system using a transfer function block. It was tested with a step input to observe time-domain characteristics such as rise time, overshoot, and settling behavior. This helped us understand how a second-order system responds to a sudden and sustained input.

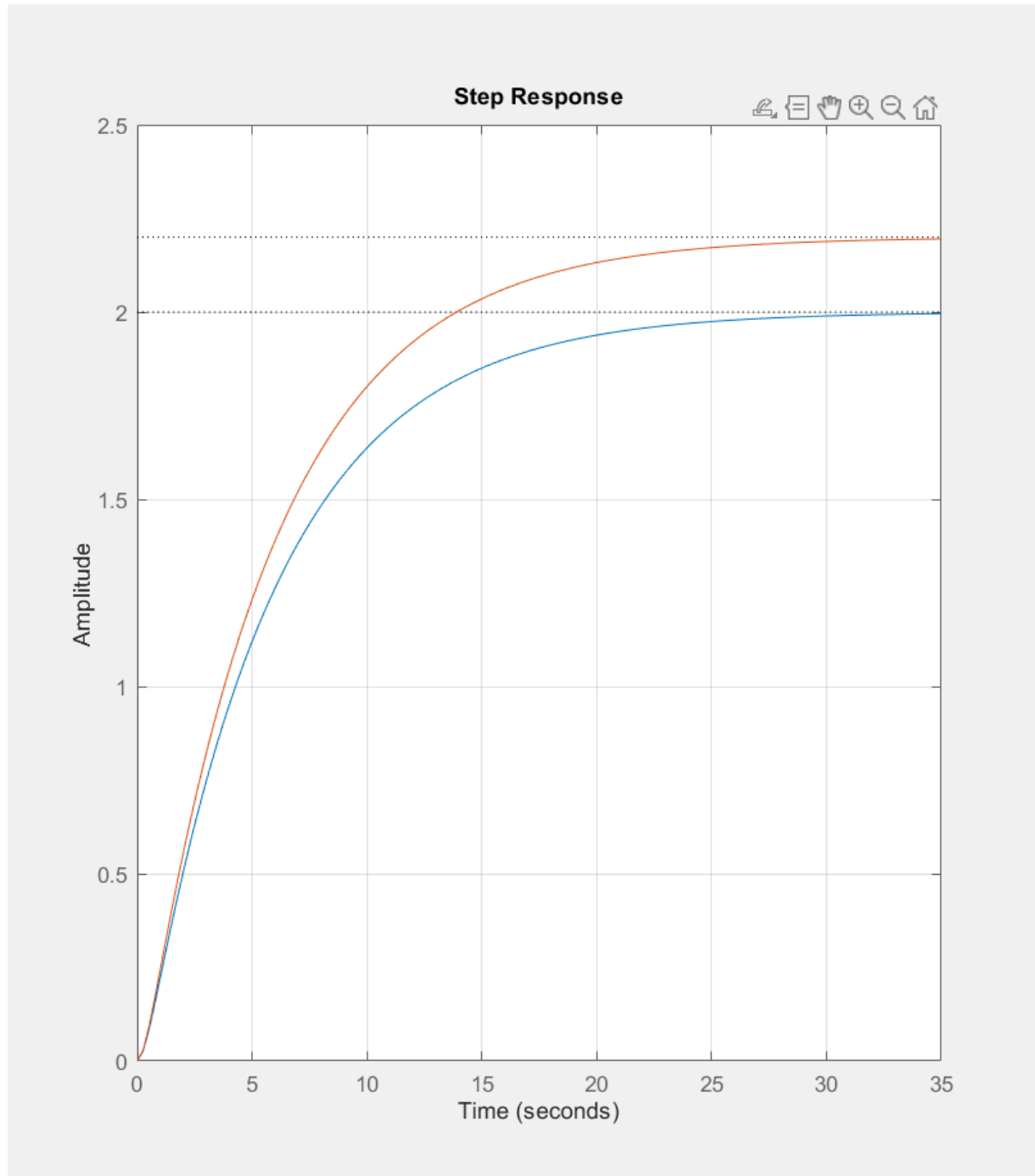
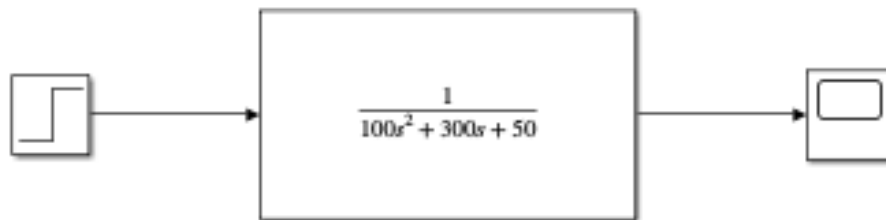


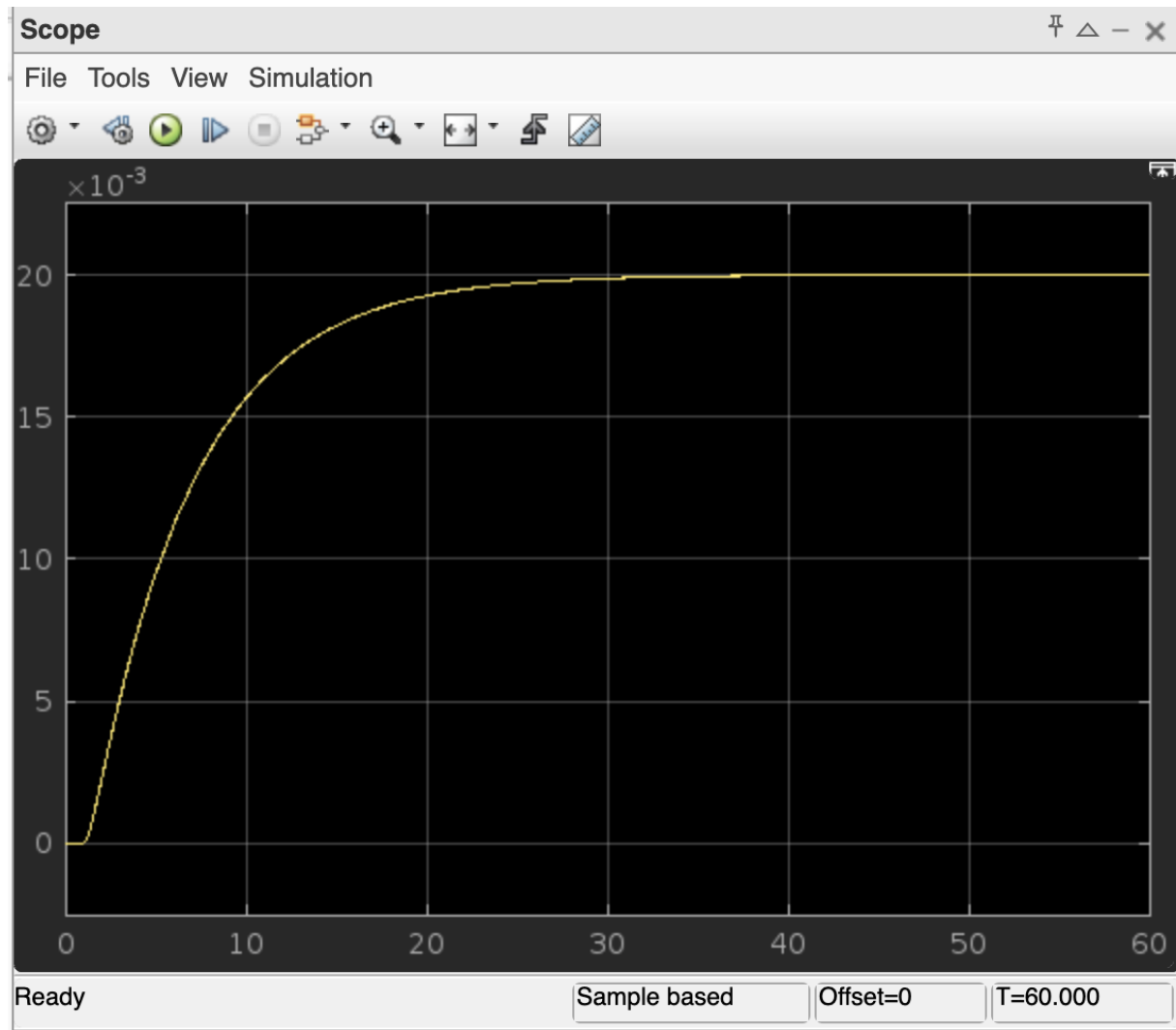
Figure 2: Step Input Response Output



## Task 2: SIMULINK Execution



*Figure 3: SIMULINK model*



*Figure 4: SIMULINK response*

The second model used the same second-order system but with an **impulse input**, simulated using a short, high-amplitude pulse. This input represents a brief disturbance, allowing us to observe the system's **natural response** without continuous forcing. The output revealed key properties like damping and oscillation, helping us understand how the system reacts and stabilizes after a sudden shock.

## Conclusion

This experiment helped us understand the dynamic behavior of control systems by modeling and simulating their responses in Simulink. Through step and impulse tests, we observed how parameters like damping and natural frequency influence system performance. The use of Simulink helped us to understand theoretical concepts and highlighted the value of simulation in control system analysis.

**Kathmandu University**  
**Department of Computer Science and Engineering**  
**Dhulikhel, Kavrepalanchowk**



**A Lab Report**  
**on**  
**“Control System”**  
**Lab No: III**

**[Code No: COEG304]**

**Submitted by:**

Royas Shakya

CE 2022

**Roll No: 55**

**Submitted to:**

Ms. Supriya Pandeya

Department of Electrical and Electronics. Engineering

**Submission Date: 19/06/2025**

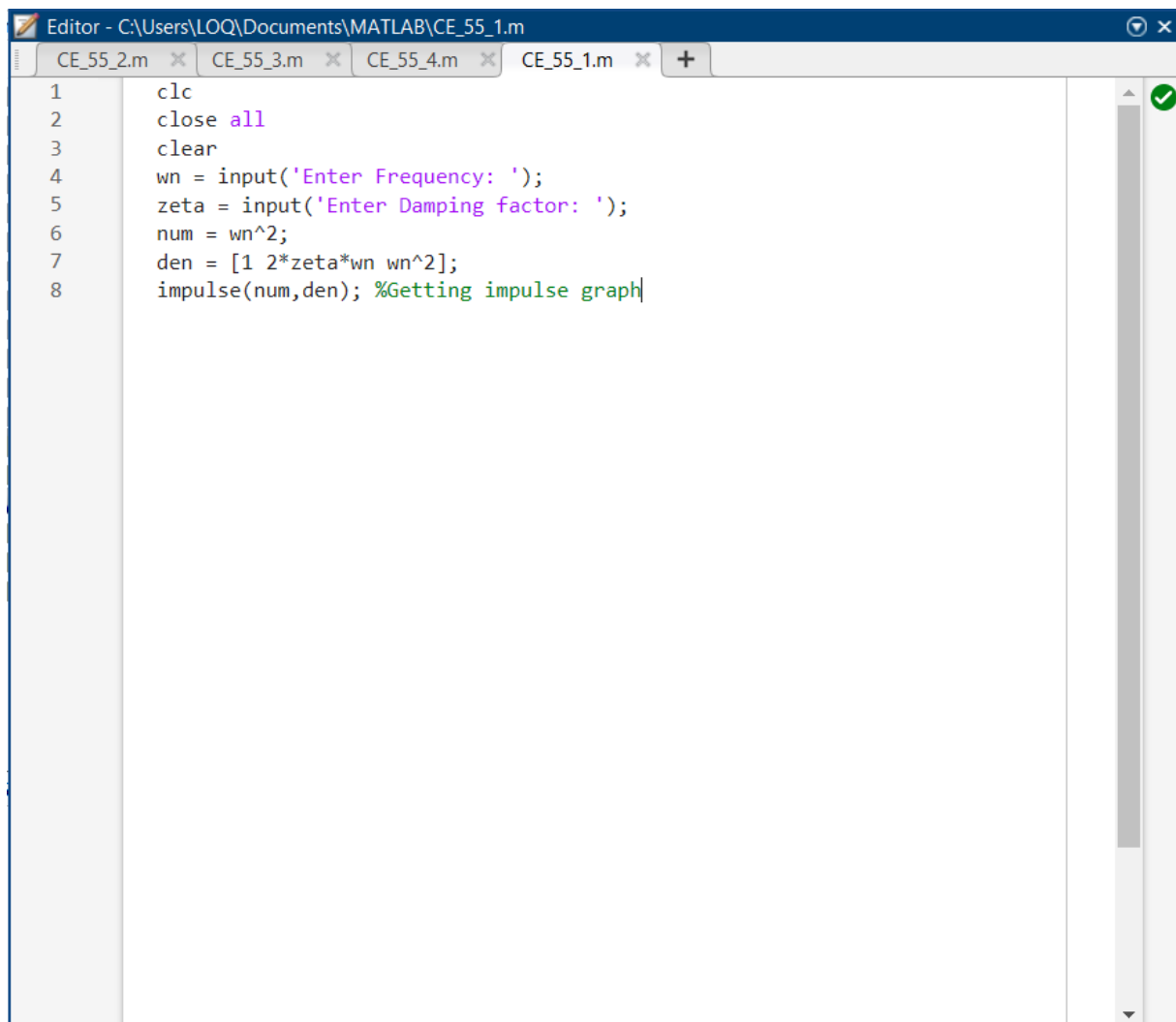
## Title: To visualize the response of a Second Order System

### Theory:

A second order system is given by the following transfer function:

$$T(s) = \frac{kw^2}{s^2 + 2\xi\omega s + \omega^2}$$

### Task 1: Impulse Response of System

The image shows a MATLAB Editor window with the title bar 'Editor - C:\Users\LOQ\Documents\MATLAB\CE\_55\_1.m'. There are four tabs open: 'CE\_55\_2.m', 'CE\_55\_3.m', 'CE\_55\_4.m', and 'CE\_55\_1.m'. The code in the editor is as follows:

```
1 clc
2 close all
3 clear
4 wn = input('Enter Frequency: ');
5 zeta = input('Enter Damping factor: ');
6 num = wn^2;
7 den = [1 2*zeta*wn wn^2];
8 impulse(num,den); %Getting impulse graph
```

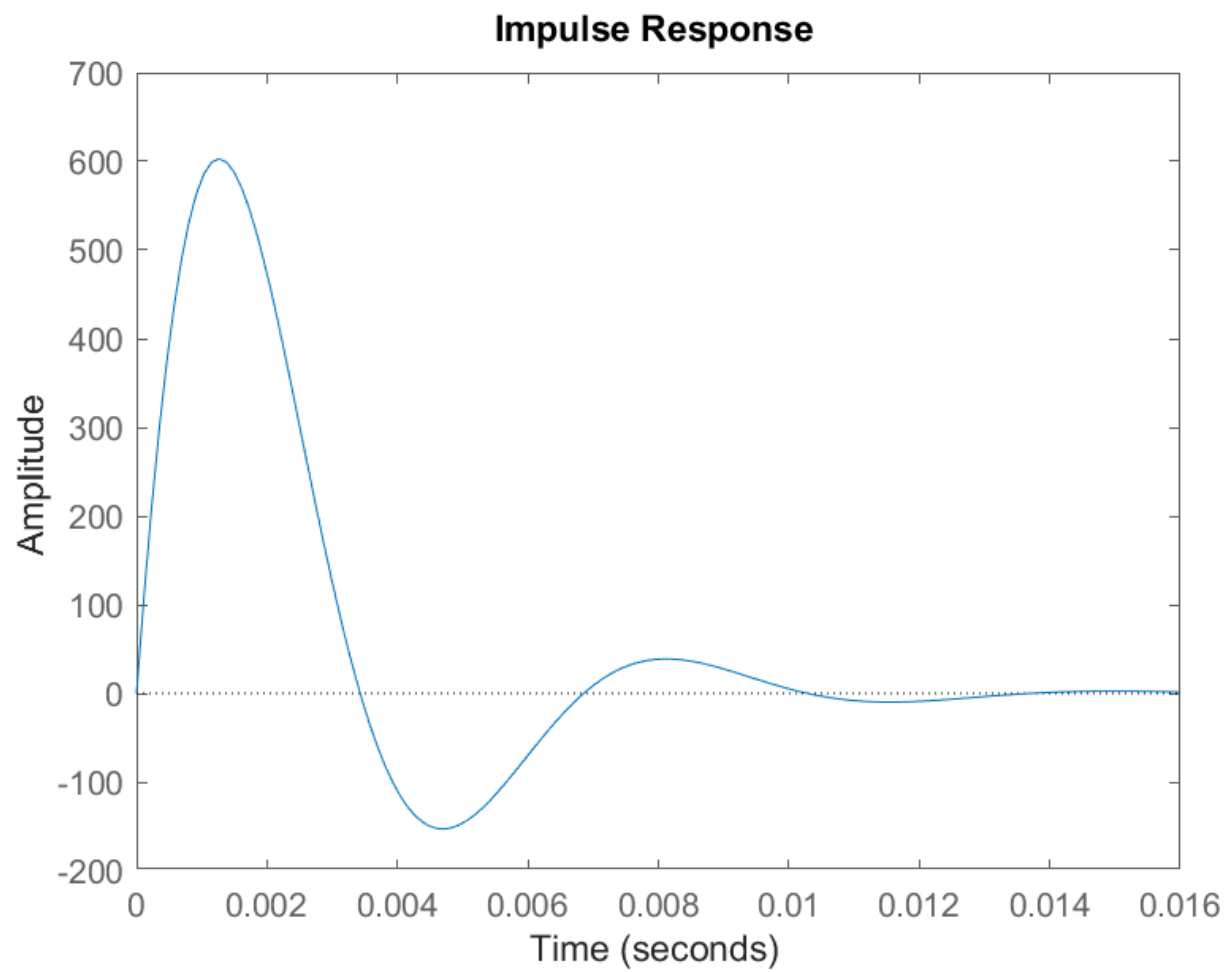
The code is color-coded: 'clc', 'close all', and 'clear' are in black; 'wn = input('Enter Frequency: ');' and 'zeta = input('Enter Damping factor: ');' are in purple; 'num = wn^2;' and 'den = [1 2\*zeta\*wn wn^2];' are in black; and 'impulse(num,den); %Getting impulse graph' is in green. A green checkmark icon is visible in the top right corner of the editor window.

Figure 1: Impulse Response of System Code

The impulse response of a second order system shows how the system reacts to a very short input applied at time 0. It typically exhibits oscillations whose amplitude and decay depend on the damping factor and natural frequency. Underdamped condition means system gradually decays whereas overdamped means system returns to equilibrium without oscillations.

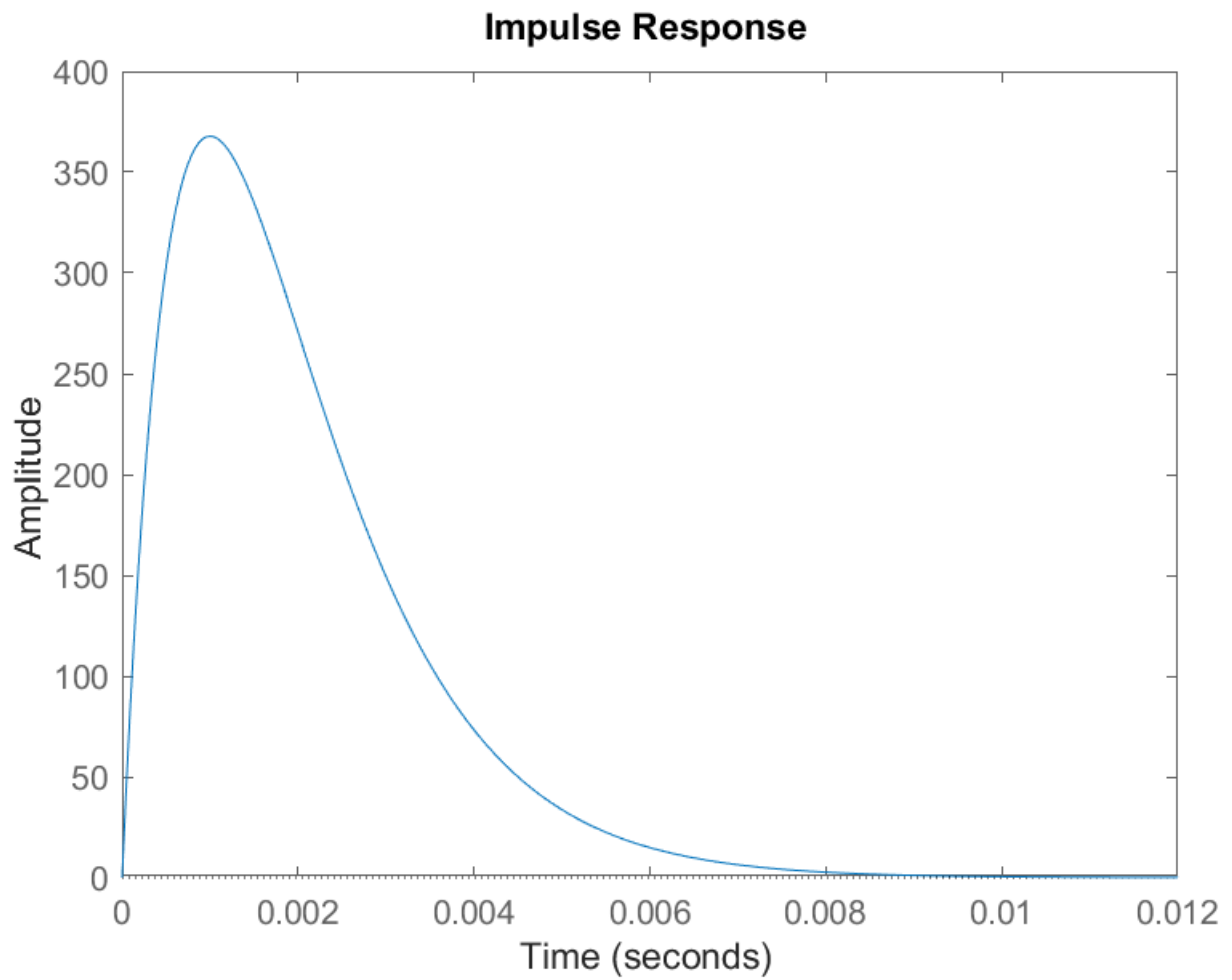
## Results:

For frequency = 1000 Hz, Damping Factor = 0.5



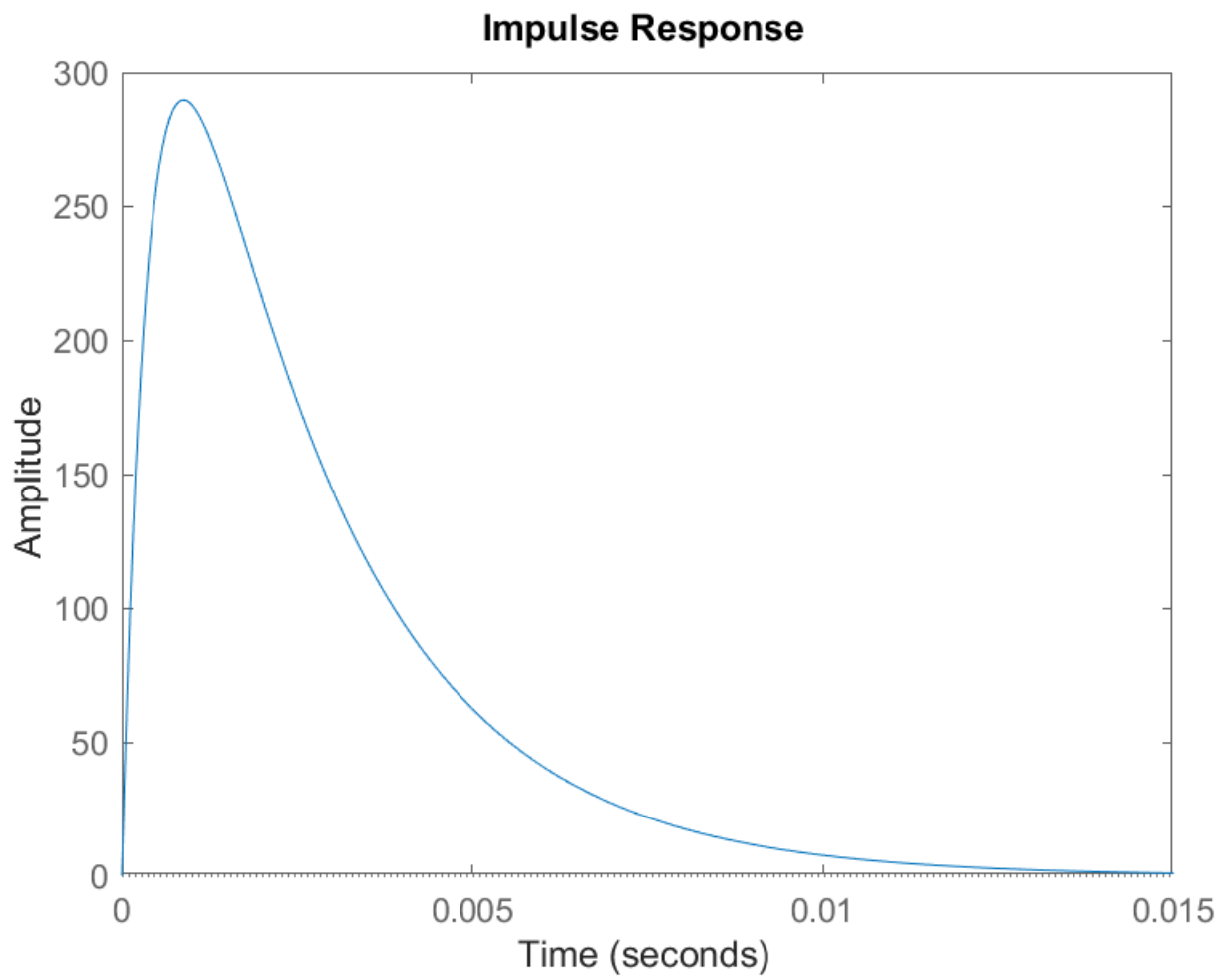
*Figure 2: Underdamped Impulse Response*

For frequency = 1000 Hz, Damping Factor = 1



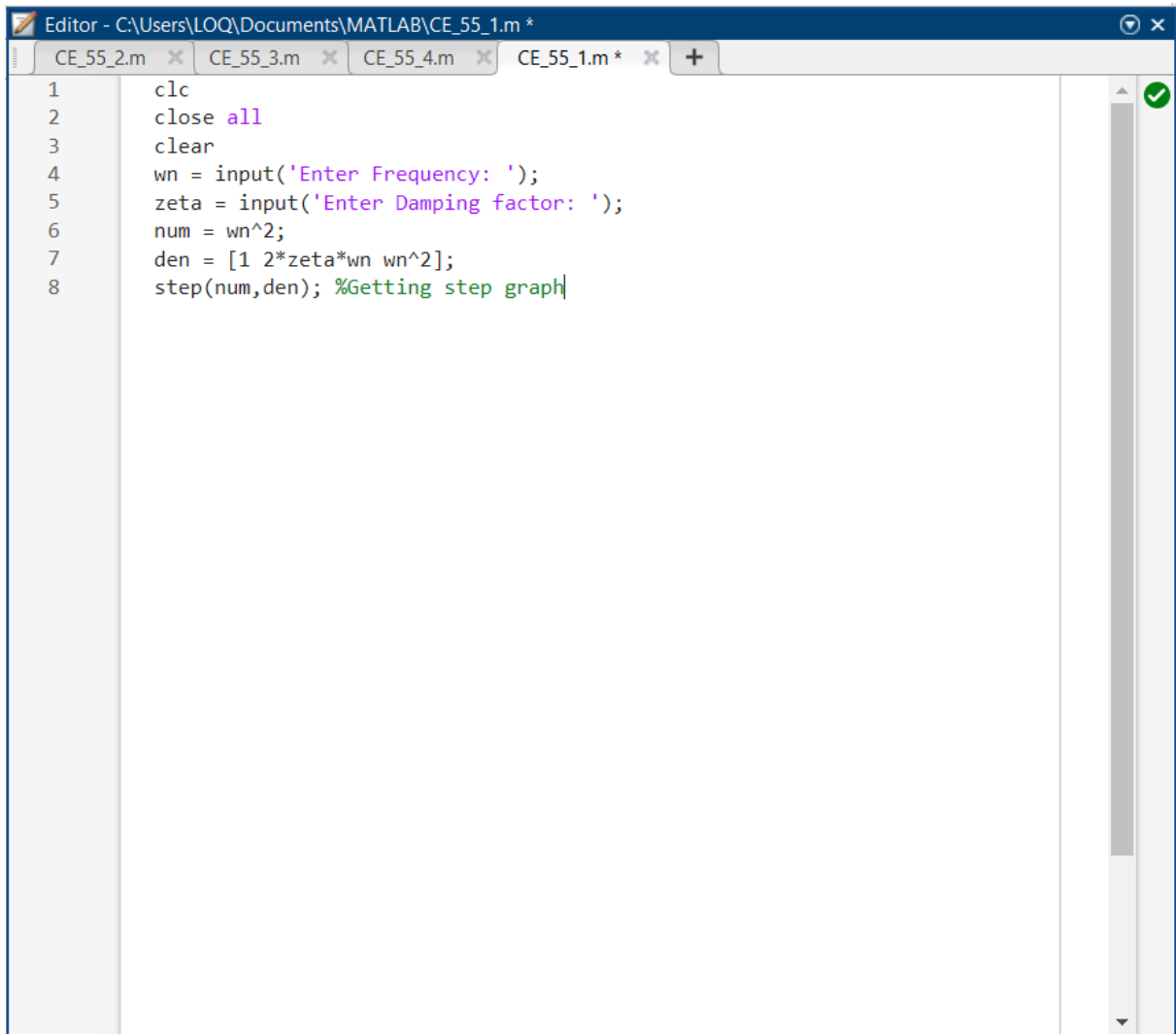
*Figure 3: Critically Damped Response*

For Frequency = 1000 Hz, Damping Factor = 1.4



*Figure 4: Overdamped Impulse Response*

## Task 2: Step Response of System

The image shows a MATLAB Editor window with the title bar "Editor - C:\Users\LOQ\Documents\MATLAB\CE\_55\_1.m \*". The window contains a script with the following code:

```
1  clc
2  close all
3  clear
4  wn = input('Enter Frequency: ');
5  zeta = input('Enter Damping factor: ');
6  num = wn^2;
7  den = [1 2*zeta*wn wn^2];
8  step(num,den); %Getting step graph
```

The script is saved in a file named CE\_55\_1.m. The window also shows other tabs for CE\_55\_2.m, CE\_55\_3.m, and CE\_55\_4.m. A green checkmark icon is visible in the top right corner of the editor window.

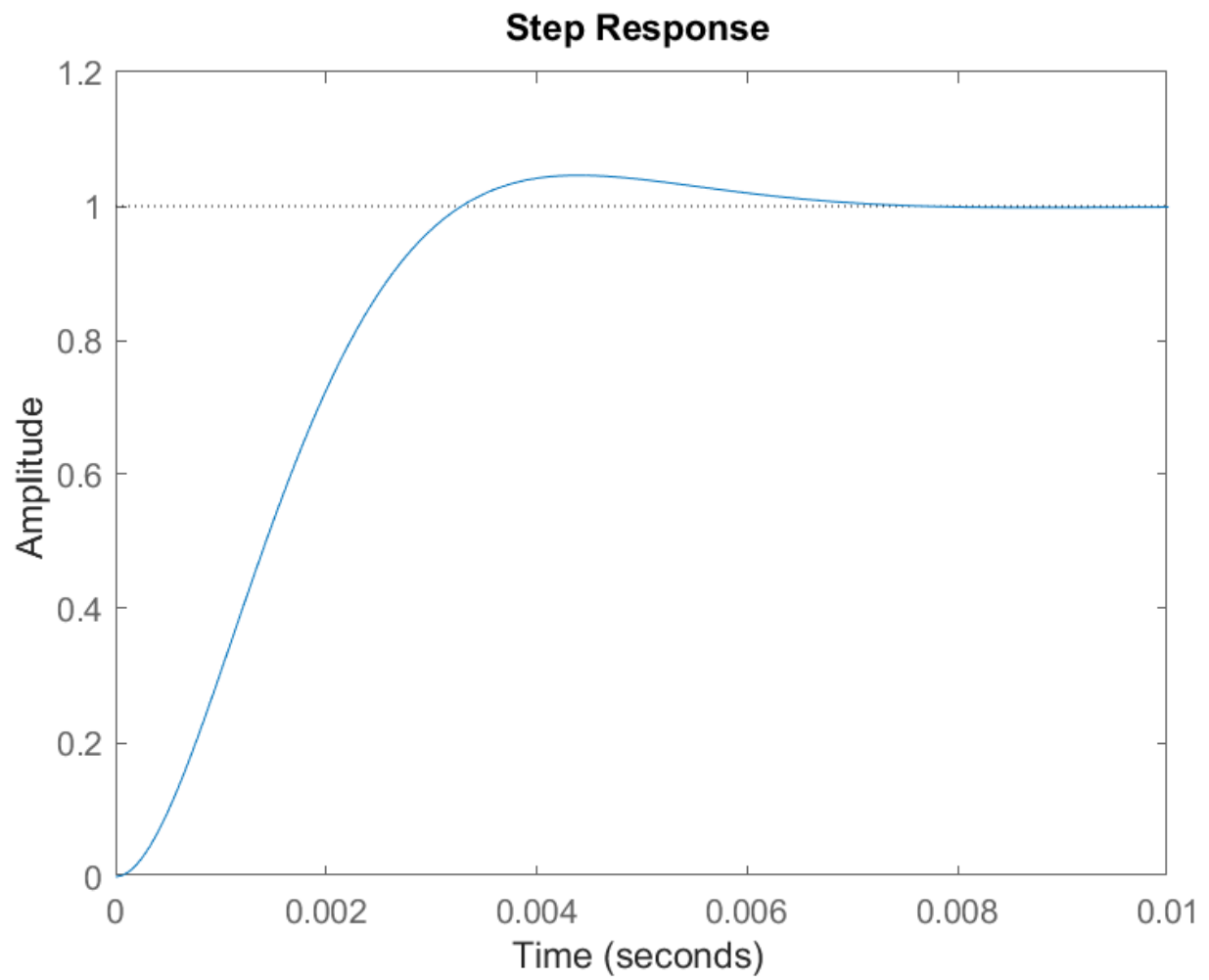
*Figure 5: Step Response of System Code*

The step response of a second order system shows how the system reacts to a sudden, sustained input. Depending on the damping factor, the response may overshoot (underdamped), smoothly rise to final value (overdamped) or quickly reaches the final value without overshoot (critically damped)



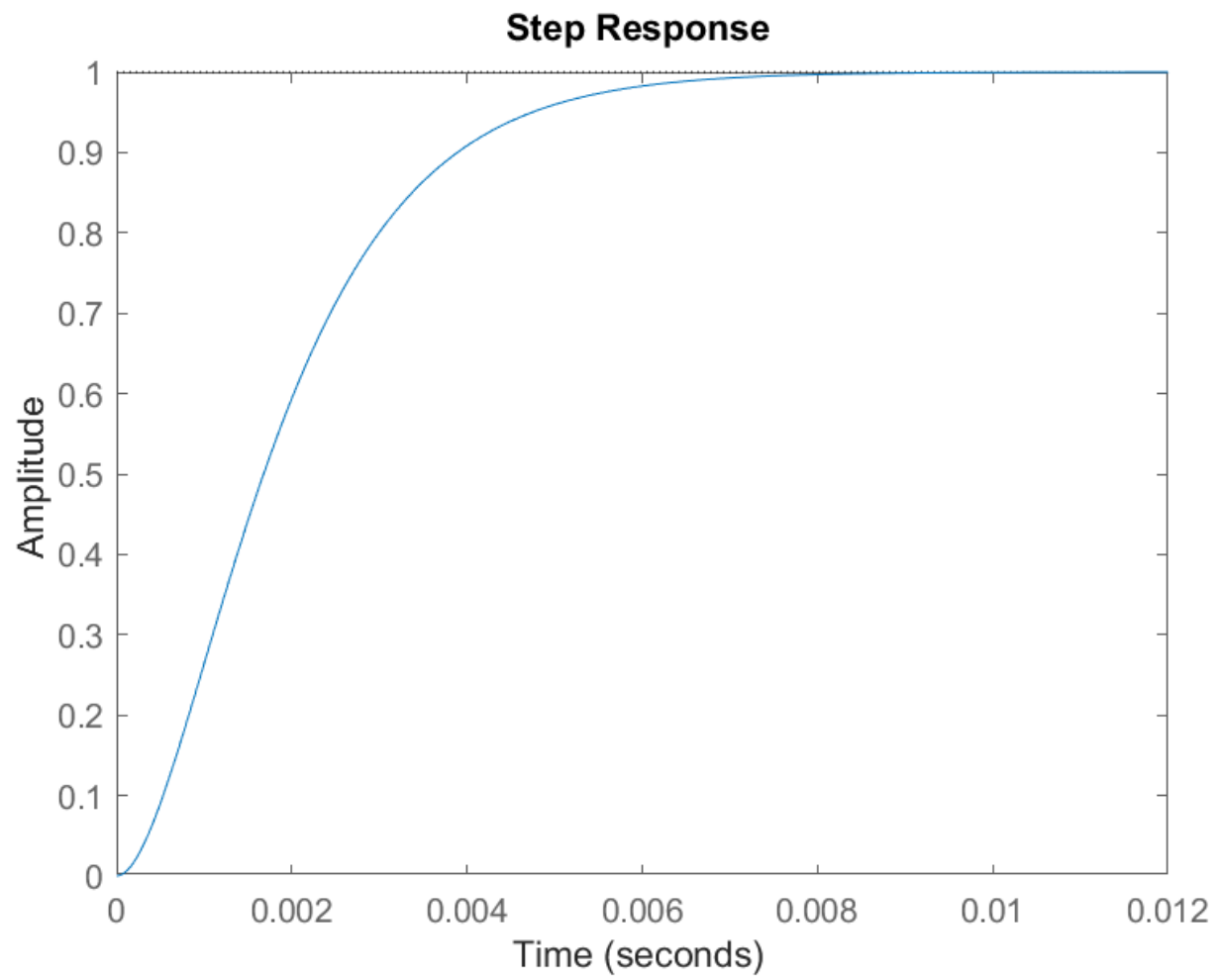
## Results:

For Frequency = 1000 Hz, Damping Factor = 0.7



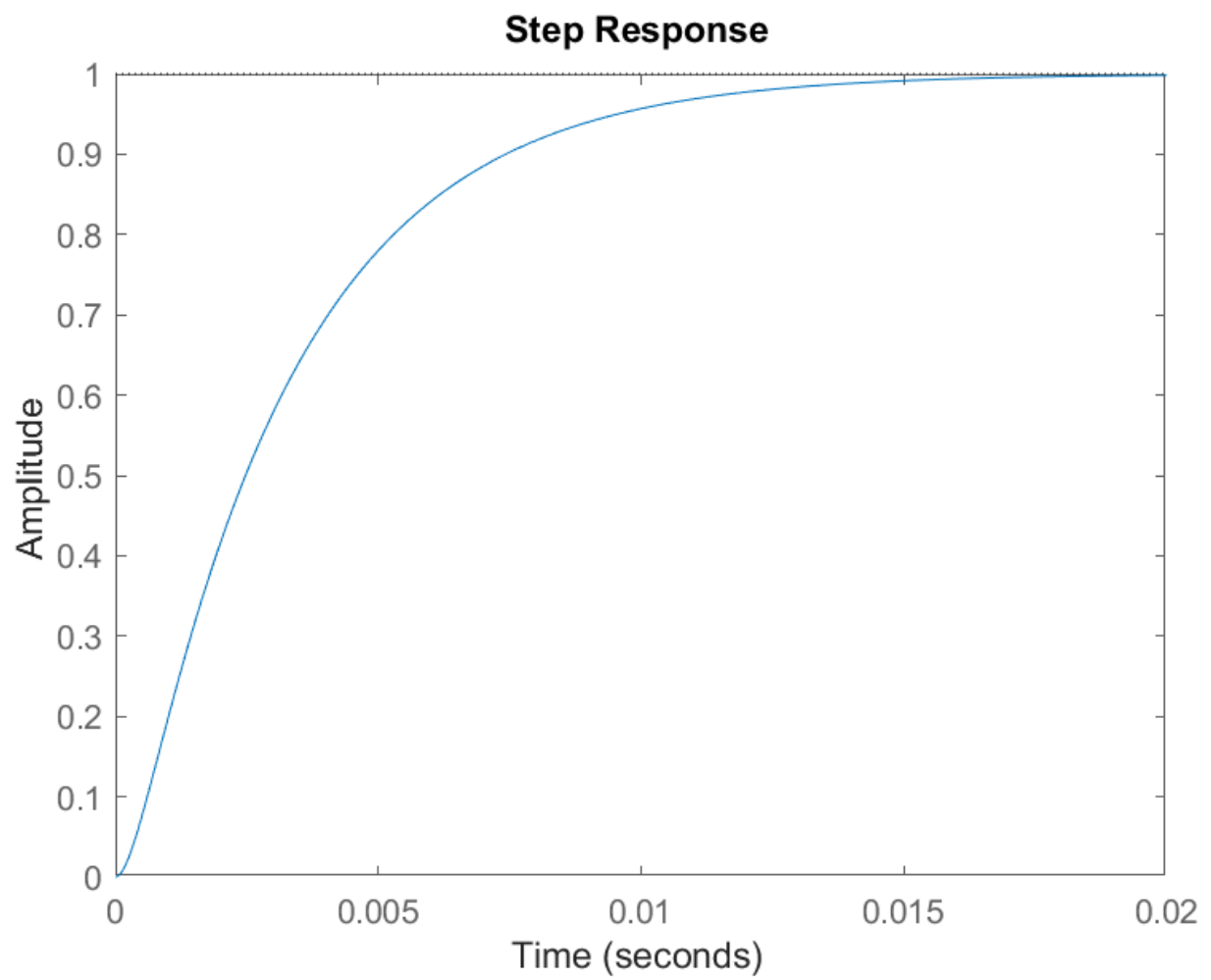
*Figure 6: Underdamped Step Response*

For Frequency = 1000 Hz, Damping Factor = 1



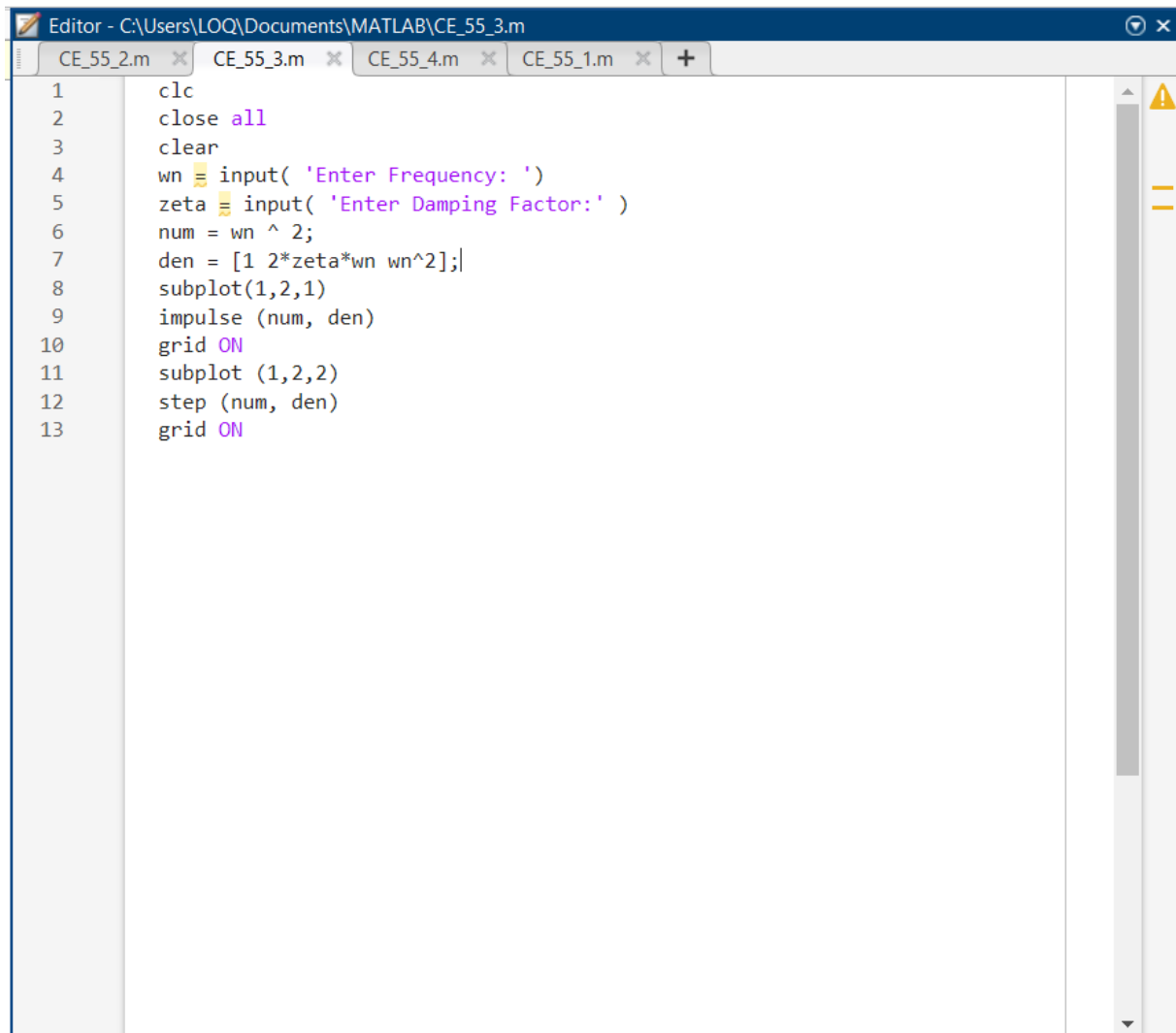
*Figure 7: Critically Damped Step Response*

For Frequency = 1000 Hz, Damping Factor = 1.7



*Figure 8: Overdamped Step Response*

### Task 3: Impulse/Step Response of System

The image shows a MATLAB Editor window with the title bar 'Editor - C:\Users\LOQ\Documents\MATLAB\CE\_55\_3.m'. The window contains four tabs: 'CE\_55\_2.m', 'CE\_55\_3.m', 'CE\_55\_4.m', and 'CE\_55\_1.m'. The 'CE\_55\_3.m' tab is active, displaying the following MATLAB code:

```
1  clc
2  close all
3  clear
4  wn = input( 'Enter Frequency: ' )
5  zeta = input( 'Enter Damping Factor:' )
6  num = wn ^ 2;
7  den = [1 2*zeta*wn wn^2];|
8  subplot(1,2,1)
9  impulse (num, den)
10 grid ON
11 subplot (1,2,2)
12 step (num, den)
13 grid ON
```

The code is line-numbered from 1 to 13. The window includes a vertical scrollbar on the right and a status bar at the bottom.

Figure 9: Impulse/Step System Code

For Frequency = 60 Hz, Damping Factor = 0.3

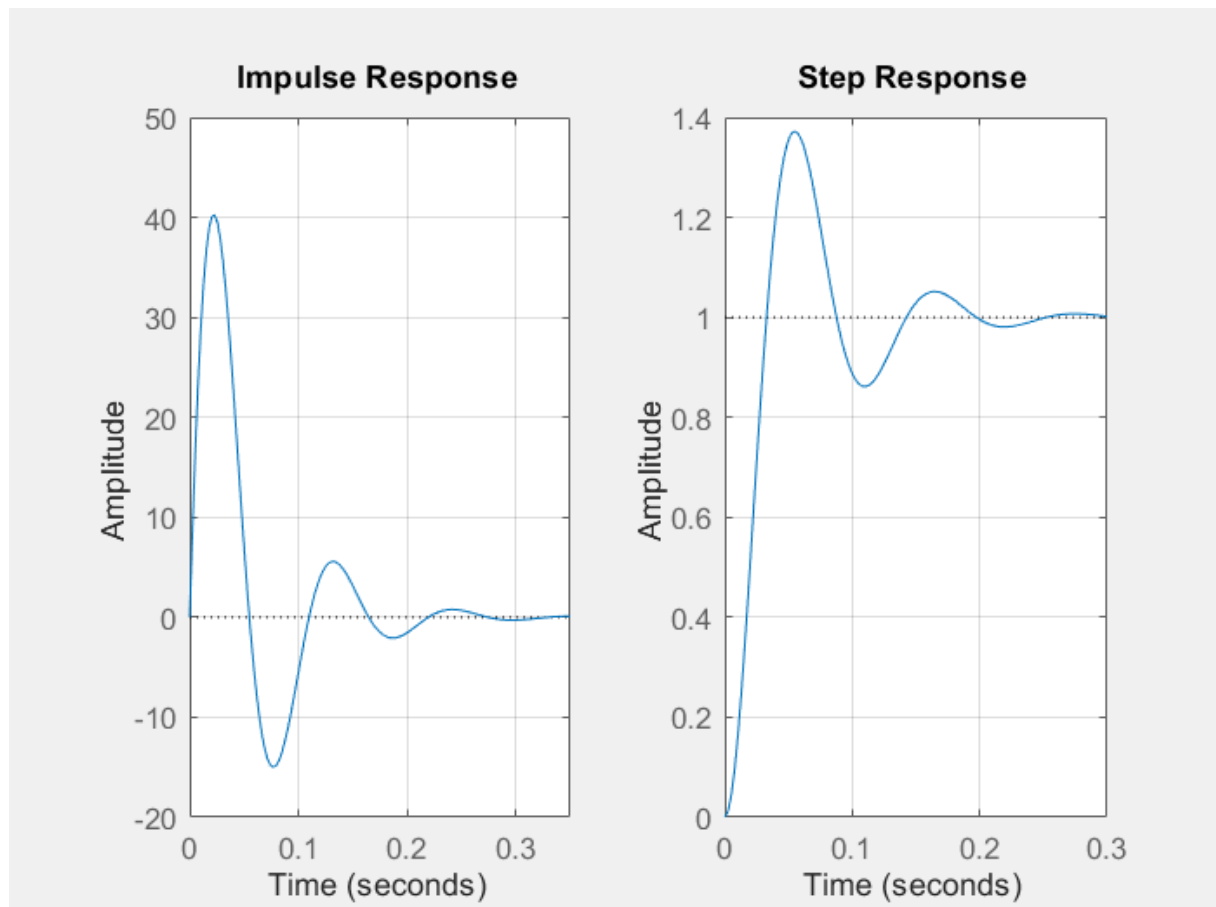


Figure 10: Underdamped Response

For Frequency = 60 Hz, Damping Factor = 1

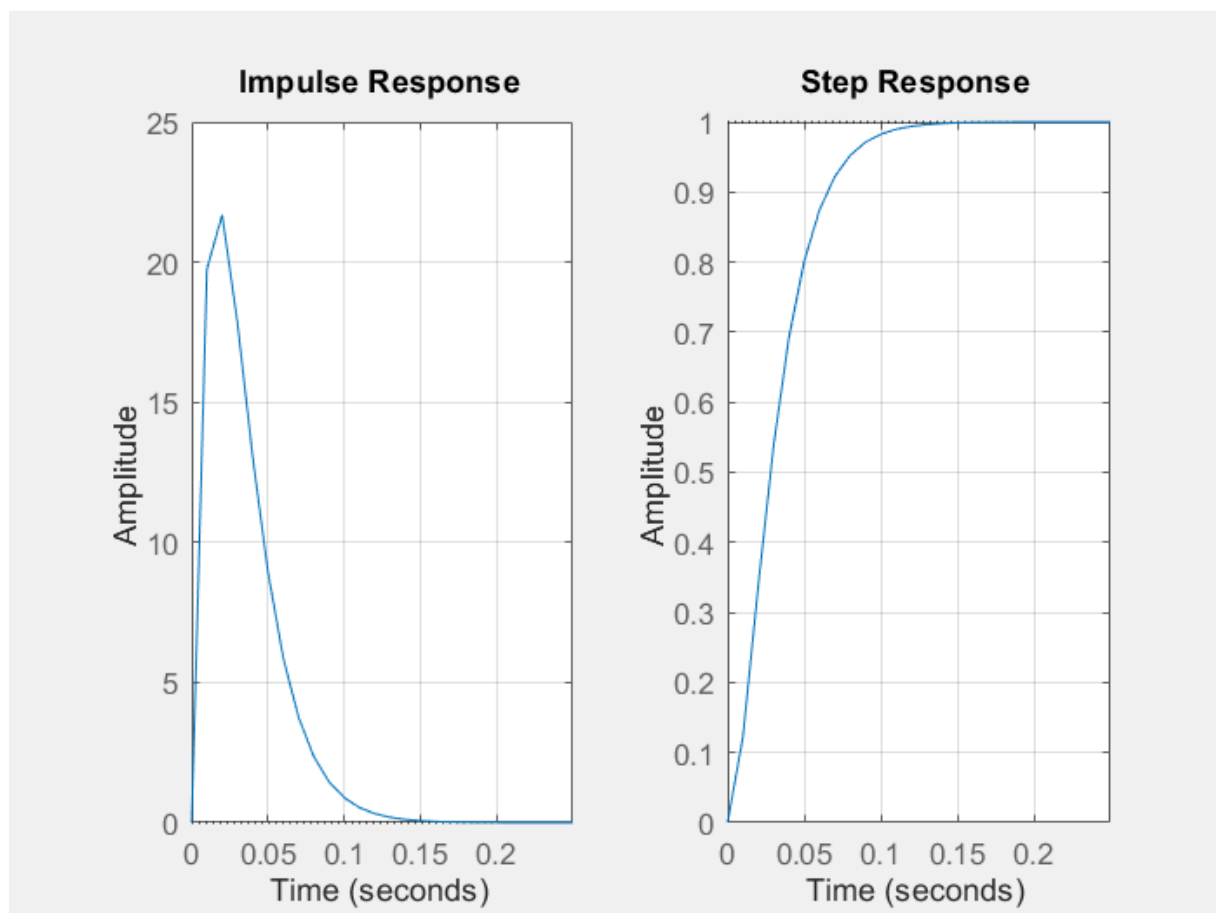
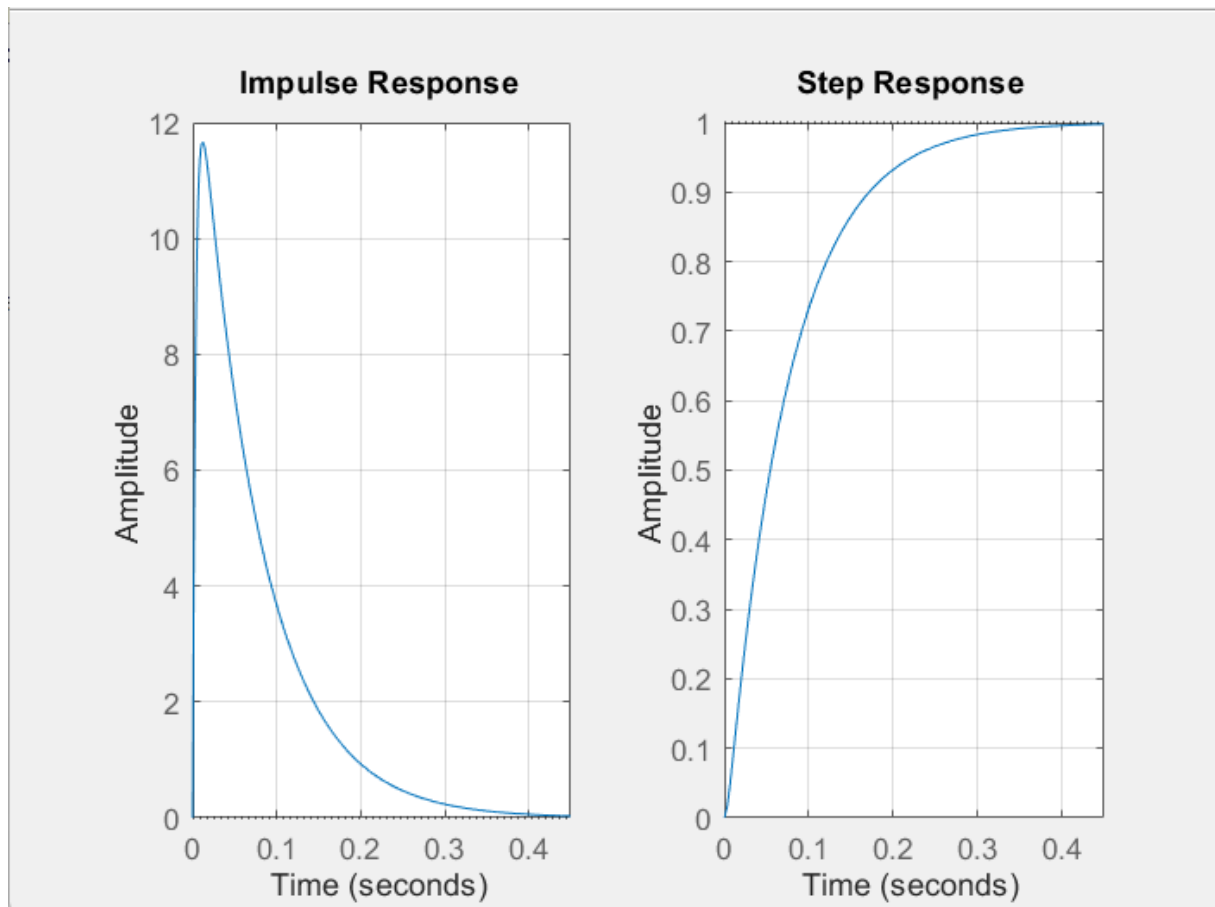


Figure 11: Critically Damped Response

For Frequency = 60 Hz, Damping Factor = 2.3



*Figure 12: Overdamped Response*

Here, with constant frequency, increasing the damping reduces oscillations and overshoot, leading to a slower but smoother response. Lower damping results in more pronounced oscillations and overshoot before settling.

For Frequency = 30 Hz, Damping Factor = 0.3

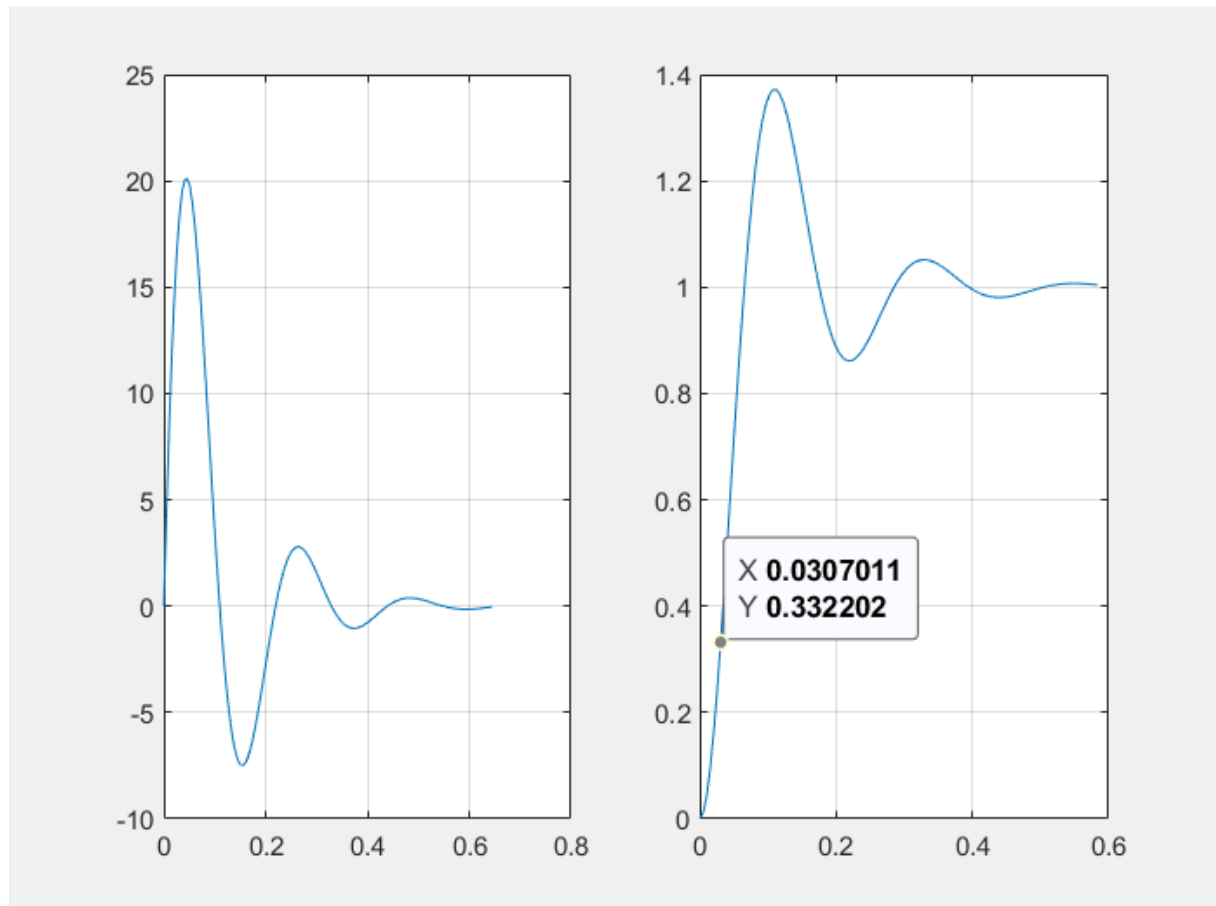


Figure 13: Underdamped Response at 30Hz



For Frequency = 60 Hz, Damping Factor = 0.3

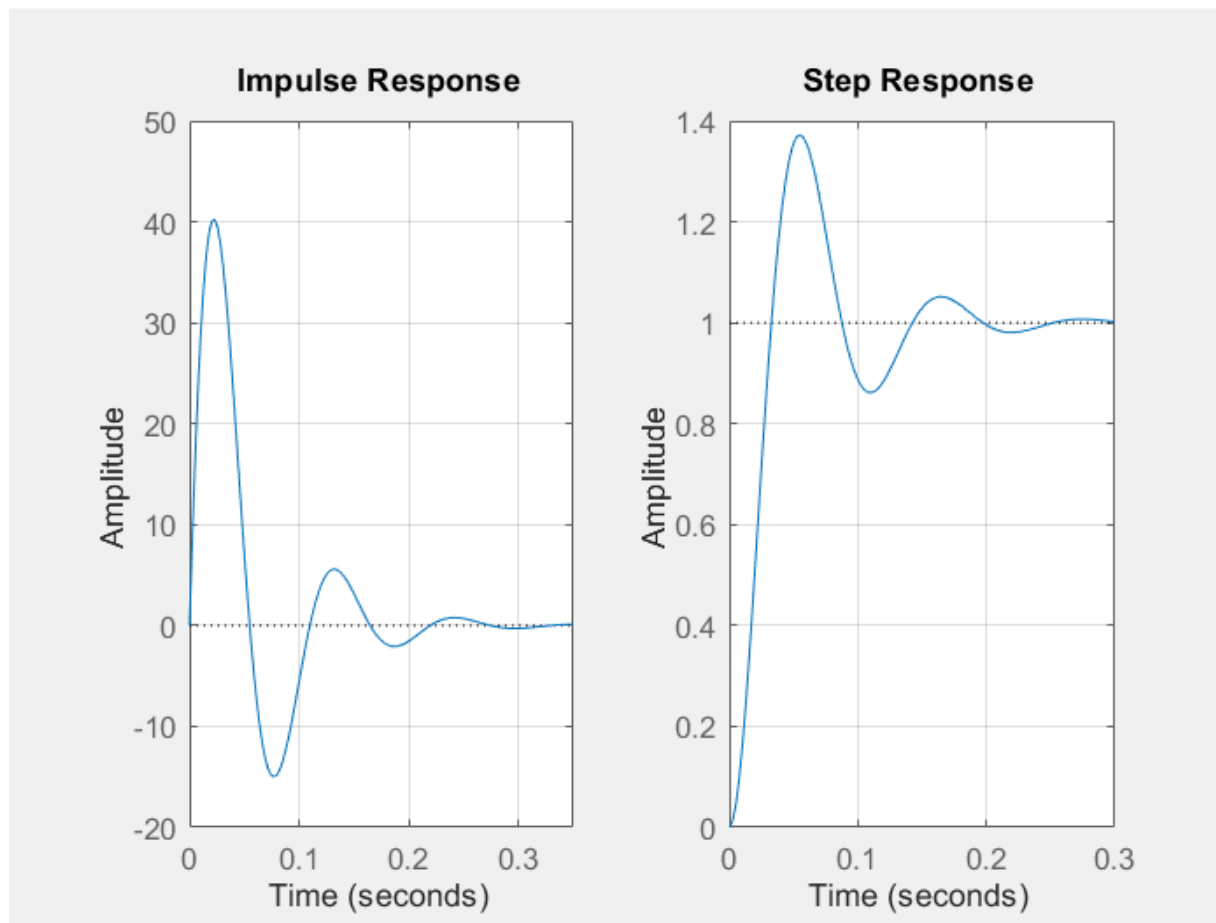
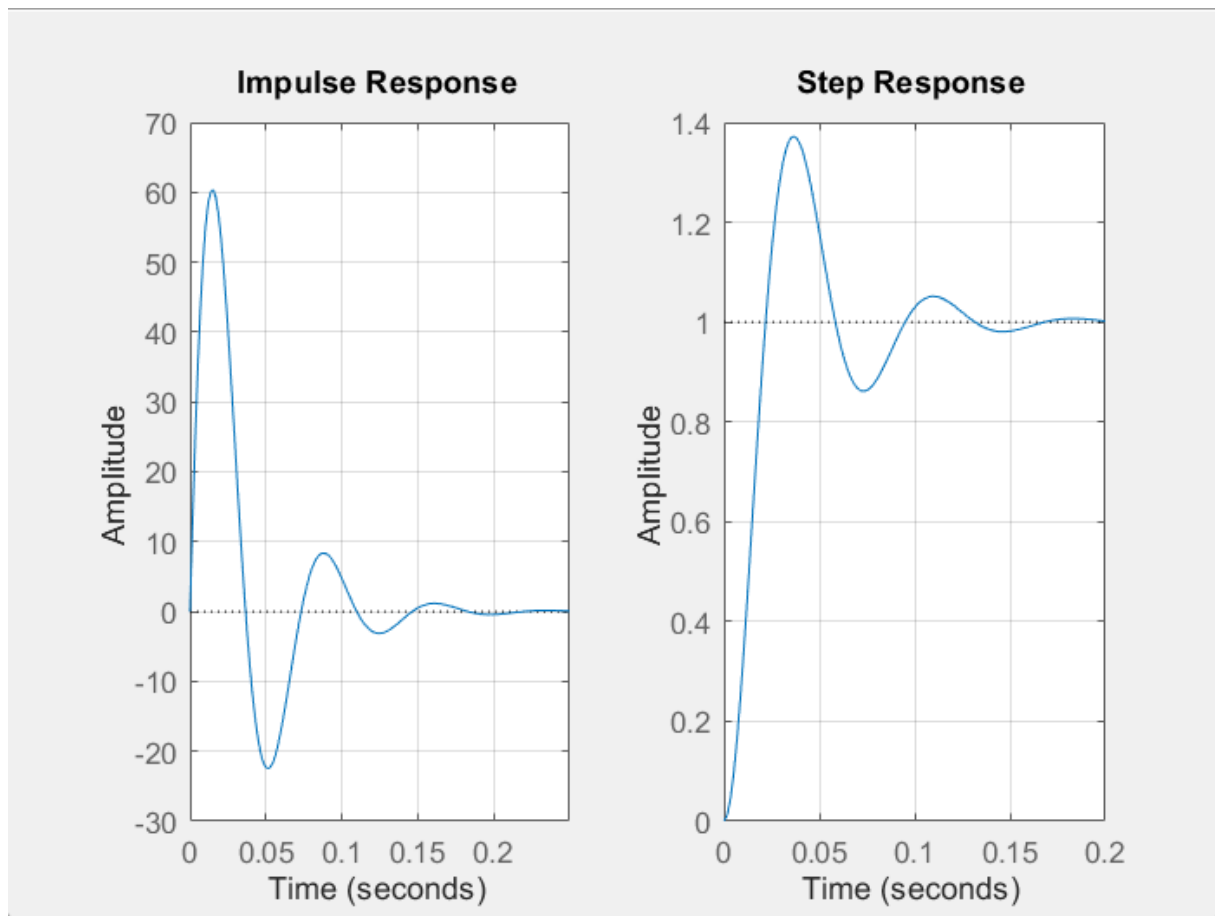


Figure 14: Underdamped Response at 60Hz

For Frequency = 90 Hz, Damping Factor = 0.3



*Figure 15: Underdamped Response at 90 Hz*

At constant damping, increasing frequency results in faster oscillations and a quicker system response. Lower frequencies produce slower system movements and longer settling times.

## Task 4: MATLAB SIMULINK

Q: Find the step and impulse response of second order system with natural frequency being last two digits of your roll number(55) and damping factor 0.5.

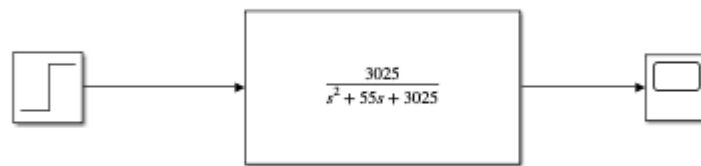


Figure 16: Step Response SIMULINK Block

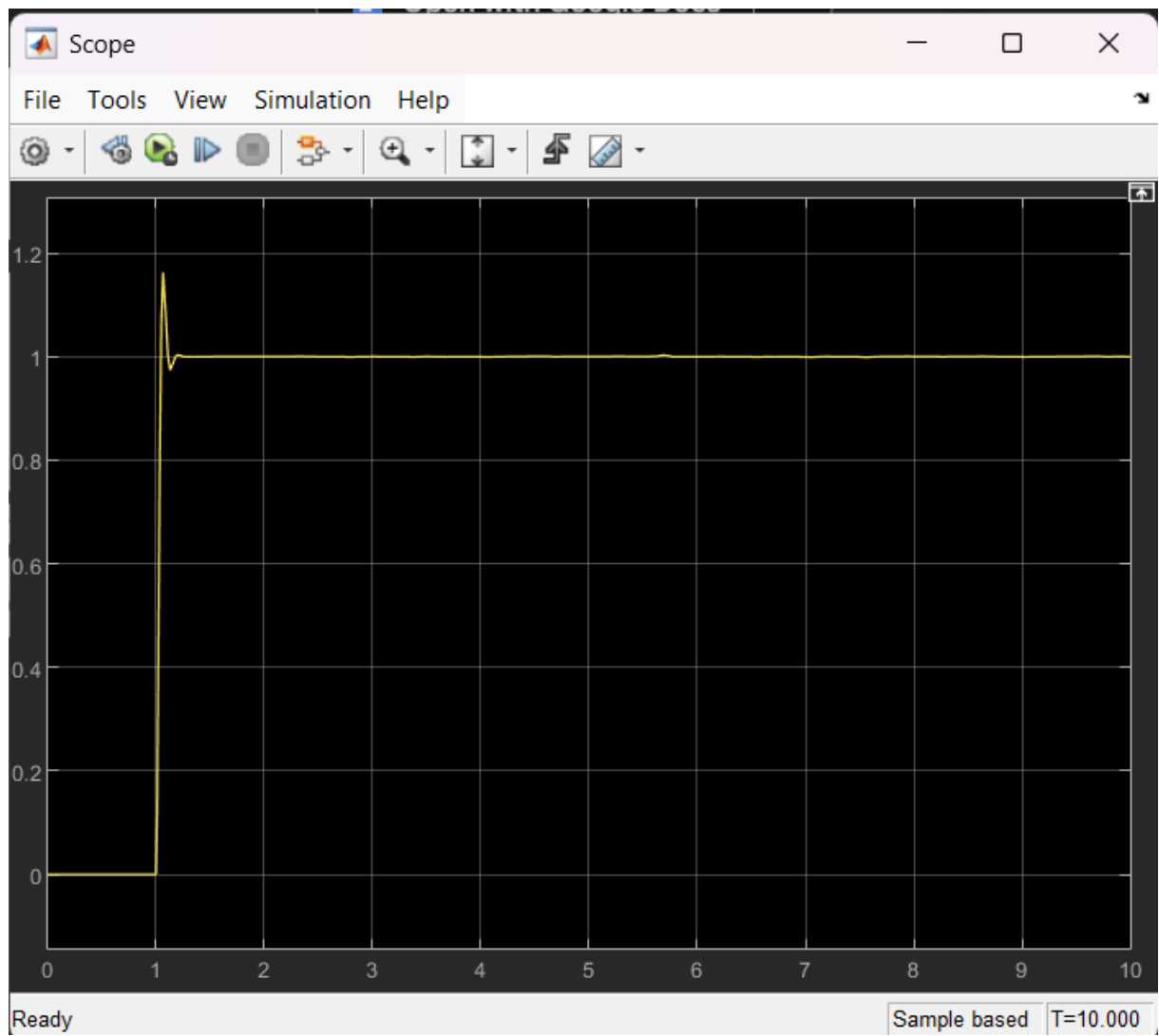


Figure 17: Step Response SIMULINK

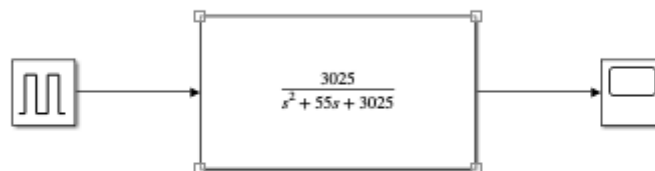


Figure 18: Impulse Response SIMULINK Block



*Figure 19: Impulse Response SIMULINK Block*

## Conclusion

The lab successfully demonstrated the behavior of a second-order system to both step and impulse inputs using MATLAB and Simulink. System parameters like damping and frequency heavily influence time-domain responses like overshoot, rise time, and settling time.