Design Project: Report

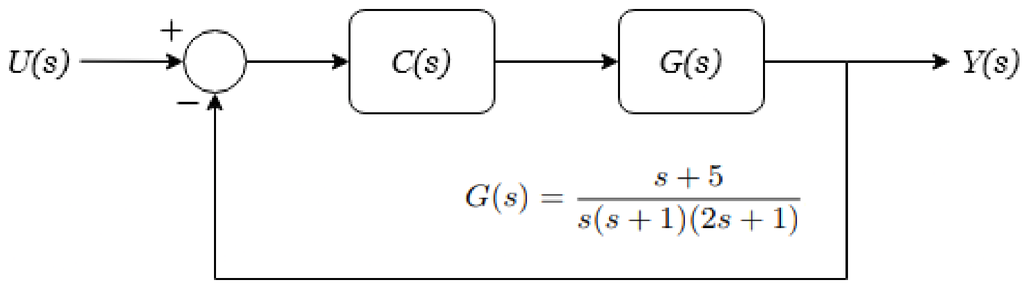
ECE 382

Nicholas Zigan

12/10/2023

Introduction:

In this report, we will design a compensator, C(s), for the system depicted below. This report contains four sections; in each section, the compensator is designed in a different way. In the first section, we design a simple proportional controller. In the second section we will design a lag/lead compensator using the root locus. For the third section, we design a PID controller, and simulate the system in Simulink. In the fourth and final section, we design a lead controller using the frequency response.



Model of the system to be used in this report

Part 1: Proportional Compensation

Outline:

(a) **C(s) = 1** *(system is uncompensated)*

- (i) Find zeros and poles of closed-loop transfer function
- (ii) Plot the step response of the system
- (iii) Determine if the system is stable or not

(b) **C(s) = K** *(proportional compensation)*

- (i) Find a gain, K, that will satisfy the following:
  - ➢ The closed-loop system becomes stable
  - ➢ Has good balance between low overshoot and settling time
- (ii) Plot the resultant step response of the system

Work/Results:

Editor - C:\Users\ntzig\Documents\MATLAB\proj.m

sys_of_equations.m | proj.m | h6.m

```
s = tf('s');

G = (s+5)/(s)/(s+1)/(2*s+1);
C = 1;

OLTF = G*C;
CLTF = feedback(OLTF, 1);

zero(CLTF)
pole(CLTF)
step(CLTF)
```

Figure 1.1: MATLAB code

Zeros	s = -5
Poles	s = -1.7468 + j0 s = 0.1234 + j1.1899 s = 0.1234 - j1.1899

Table 1.1: Zeros and Poles of CLTF

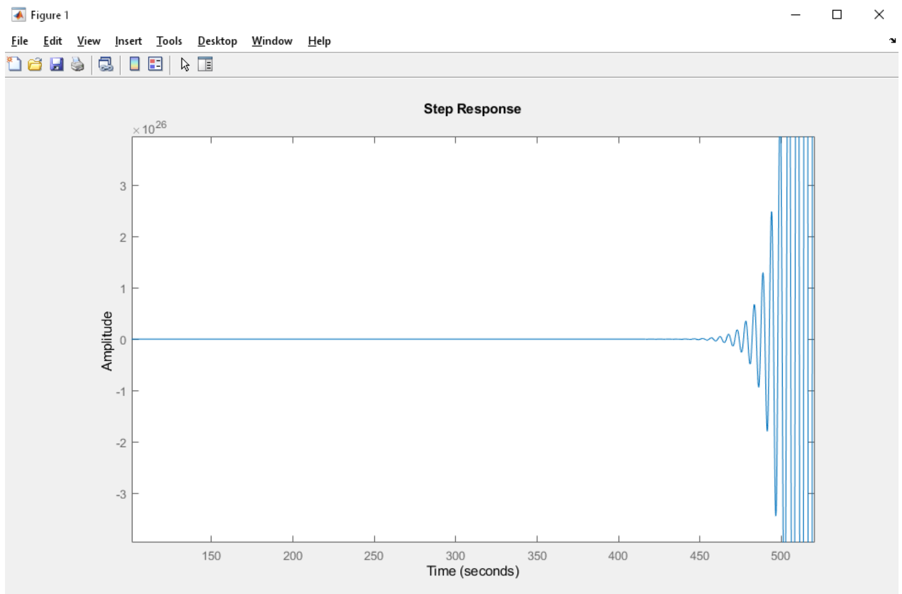


Figure 1.2: Step response of uncompensated system

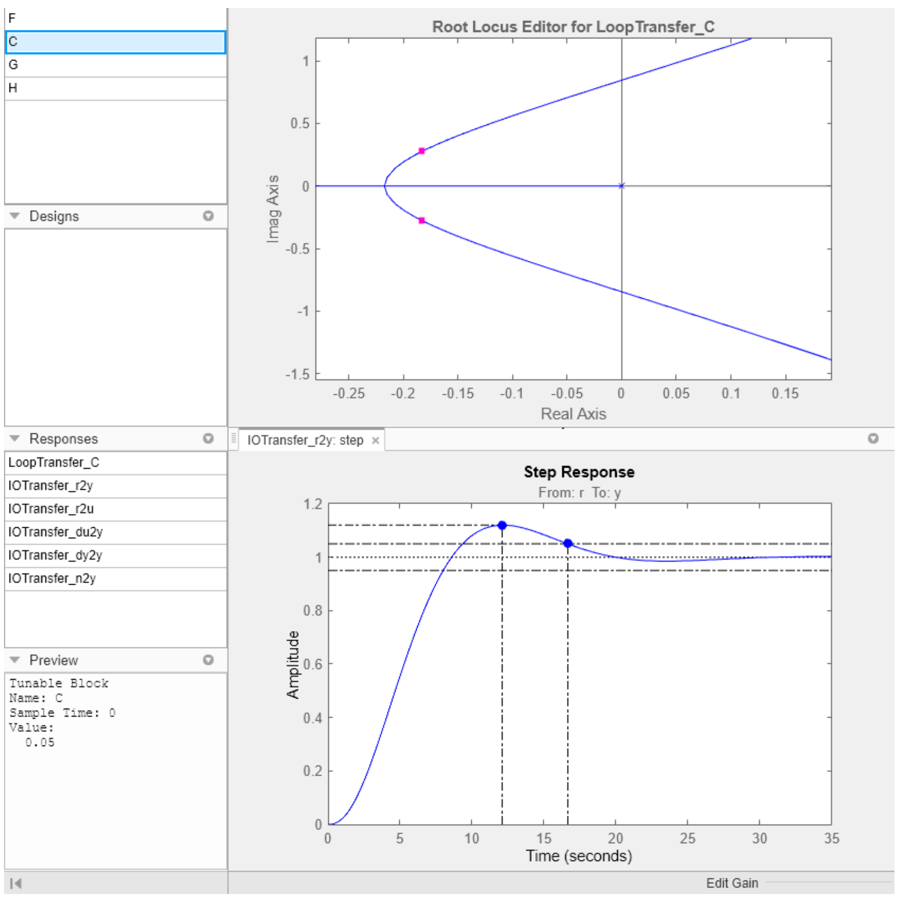


Figure 1.3: Root locus & Step response using sisotool; C(s) = 0.05

(c) C(s) = 1

(i) Zeros & Poles:

- ➢ See Table 1.1 above

(ii) **Uncompensated** step response:

- ➢ See Figure 1.2 above

(iii) Figure 1.2 above depicts the uncompensated step response of the system. The step response diverges exponentially; therefore, the uncompensated system is **unstable**.

(d) C(s) = K

(i) Using the sisotool in MATLAB, I adjusted the gain until I felt as though I had achieved a good balance between the maximal overshoot, and settling time.

- ➢ **C(s) = 0.05**
- ➢ M_P = 12%
- ➢ t_S = 16.6 seconds

(ii) **Compensated** step response:

- ➢ See Figure 1.3 above

Part 2: Root Locus Design

Outline:

(e) Plot the root locus of **uncompensated** system

(f) Design C(s) to meet the following requirements:

(i) Maximal overshoot; $\mathbf{M_P < 15\%}$

(ii) Settling time (5%); $\mathbf{t_S < 8}$ **seconds**

(iii) $\mathbf{e_{SS} < 0.1}$ *(unit ramp input)*

(g) Plot the root locus of the **compensated** system

(h) Plot the following: *(verify objectives are met)*

(i) Step response

(ii) Ramp response

Work/Results:

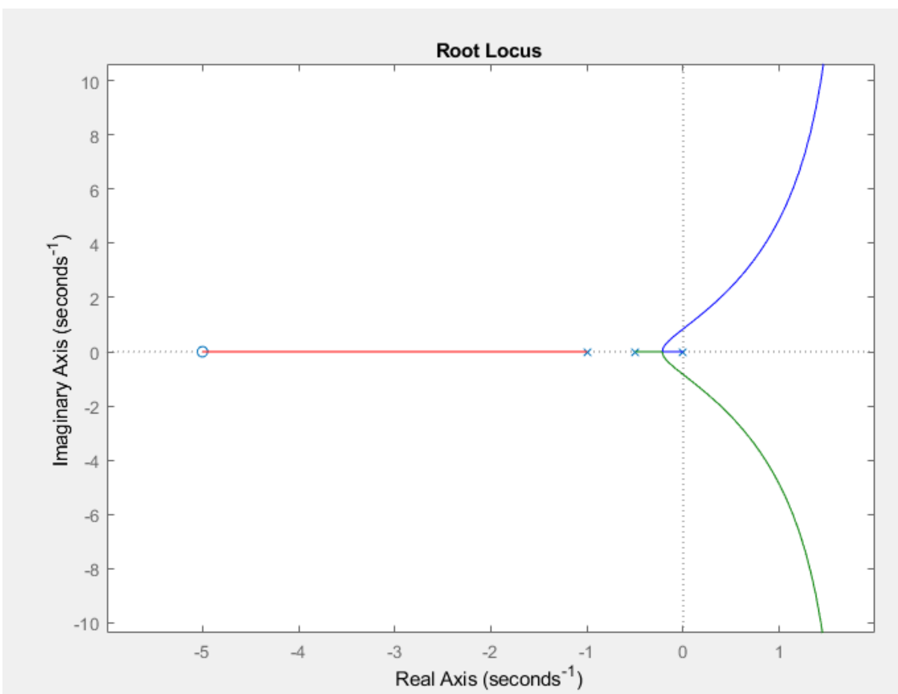


Figure 2.1: Root locus of uncompensated system

```
s = tf('s');

G = (s+5)/(s)/(s+1)/(2*s+1);

% Lead/Lag Compensator
C_lead = (s+0.5)/(s+5);
C_lag  = (s+0.01)/(s+0.00025);

OLTF = G*C_lead*C_lag;
CLTF = feedback(OLTF, 1);


%step(CLTF)   % Step response
step(CLTF/s) % Ramp response
grid on

stepinfo(CLTF,"SettlingTimeThreshold", 0.05)
```

Figure 2.2: MATLAB code

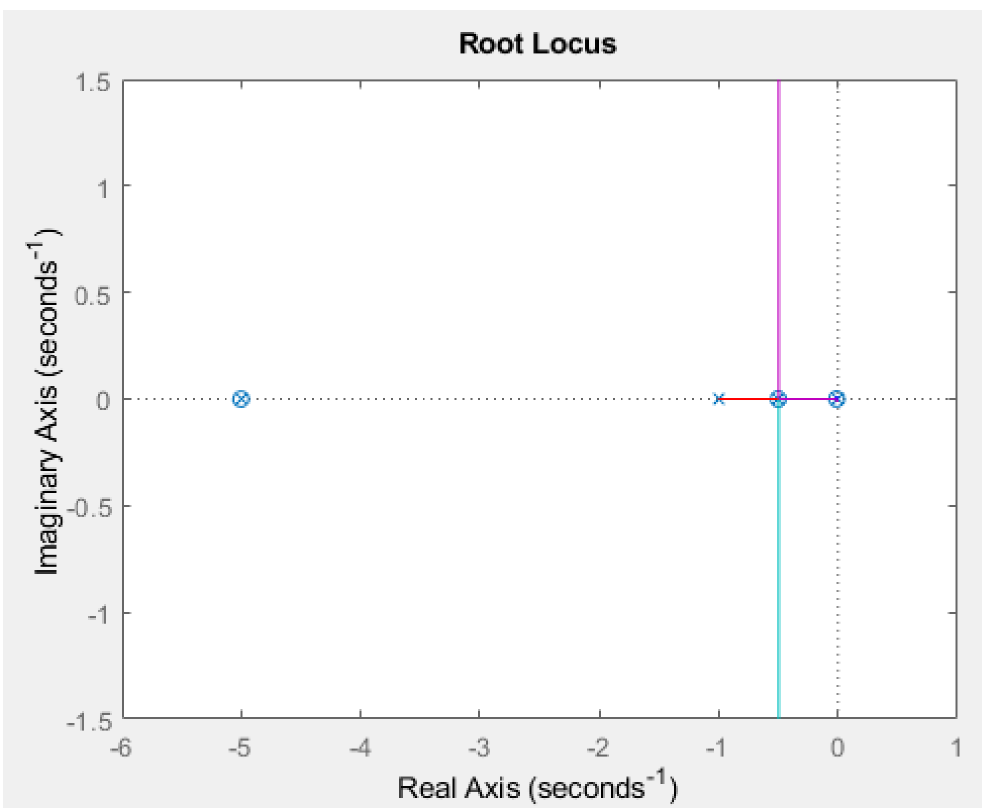


Figure 2.3: Root locus of compensated system

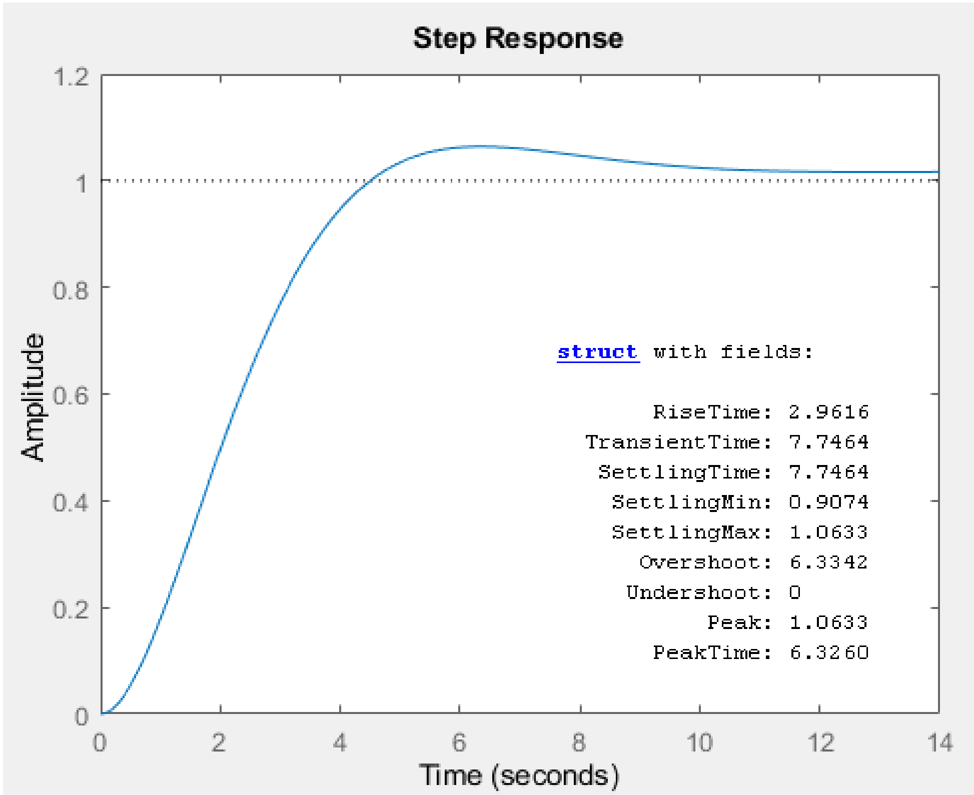


Figure 2.4: Step response of compensated system

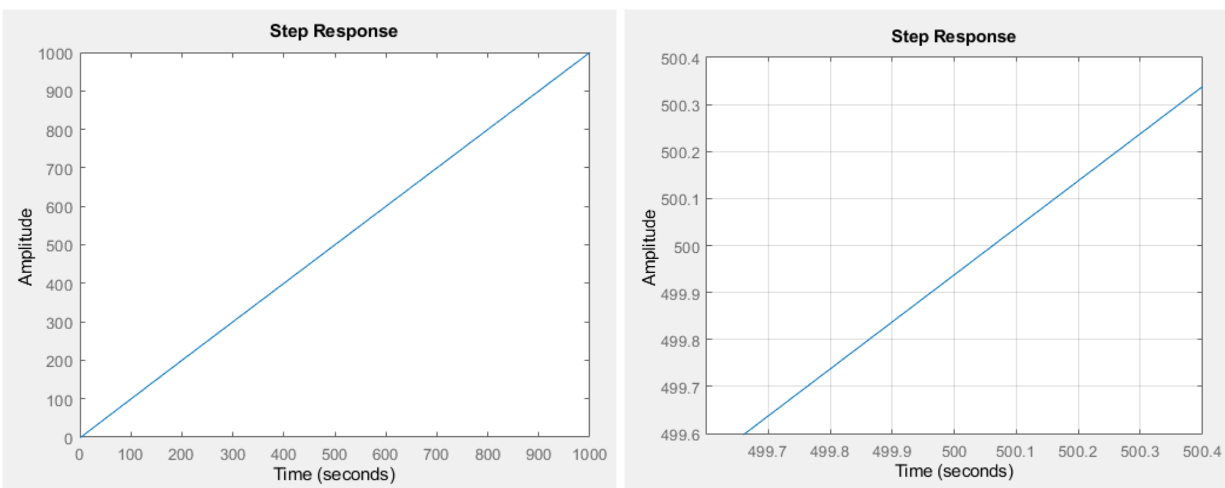


Figure 2.5: Ramp response of the compensated system

(a) Plot **uncompensated** system:

- ➢ See figure 2.1 above

(b) Design C(s) to meet the requirements:

(i) $C(s) = \frac{(s+0.5)}{(s+5)} \cdot \frac{(s+0.01)}{(s+0.00025)}$

- ➢ Figure 2.2 contains the MATLAB code
- ➢ Figure 2.4 contains stepinfo

(c) Plot the root locus of the **compensated** system:

- ➢ See Figure 2.3 above

(d) Plot the following: *(verify objectives are met)*

(i) Step response:

- ➢ See Figure 2.4 above

(ii) Ramp response:

- ➢ See Figure 2.5 above

Figures **2.6 through 2.9** below shows the **handwritten work** done to design the compensator

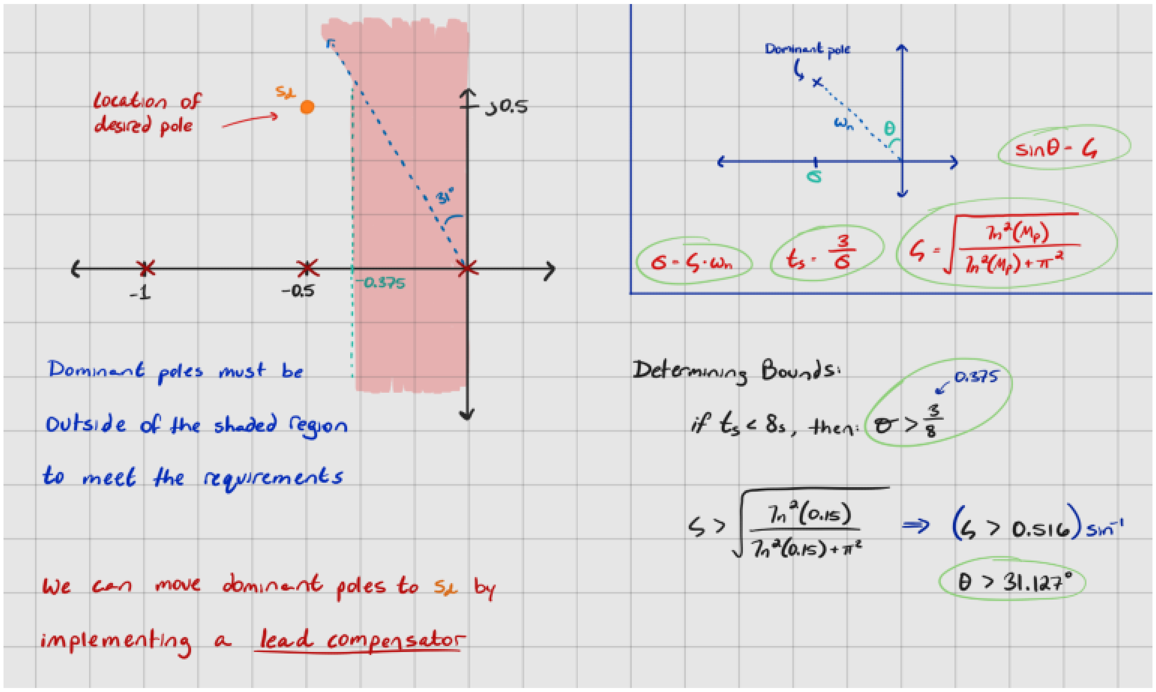


Figure 2.6: Plotting **constraints** on root locus

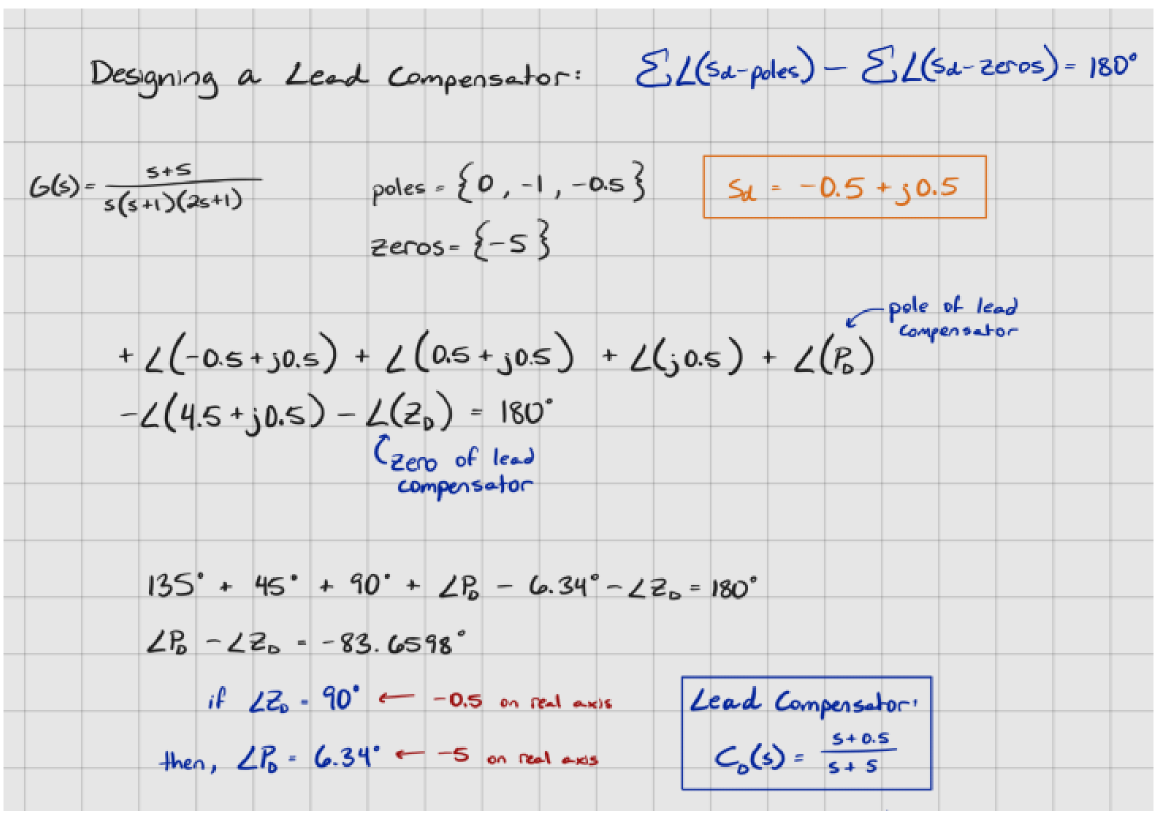


Figure 2.7: Design process for the **lead** compensator

$$G(s) = \frac{s+5}{s(s+1)(2s+1)}$$

$$C(s) = \frac{s+0.5}{s+5} = \frac{\frac{1}{2}(2s+1)}{s+5}$$

$$\boxed{e_{ss}\{\text{unit ramp input}\} = \frac{1}{K_v} \qquad K_v = \lim_{s\to 0}\left[s(\text{OLTF})\right]}$$

$$G(s)\cdot C(s) = \frac{(1/2)}{s(s+1)} \qquad K_v = \frac{1}{2}, \quad \text{Hence: } e_{ss} = 2$$

We need to design a lag compensator so: $e_{ss} < 0.1$

currently: $e_{ss} = \frac{1}{K_v} = \frac{1}{(1/2)}$ need $K_v > 10$ for $e_{ss} < 0.1$

Lag compensator DC gain $\times\ (1/2) > 10$

Lag compensator DC gain > 20

to be certain that I meet criteria I will design lag compensator to have a DC gain of 40

The zero of the compensator is placed at $\frac{1}{50}^{\text{th}}$ of the dominant poles location

Hence; zero: $s = -0.01$

pole: $s = -0.00025$ ← gives us DC gain of 40

$$\boxed{C_L(s) = \frac{s+0.01}{s+0.00025}} = \frac{\frac{1}{100}(100s+1)}{\frac{1}{4000}(4000s+1)} = \frac{40(100s+1)}{(4000s+1)}$$

Figure 2.8: Design Process of **lag** compensator

Final Lag/Lead Compensator:

$$\boxed{C(s) = \left(\frac{s+0.01}{s+0.00025}\right)\left(\frac{s+0.5}{s+5}\right)}$$

$$K_v = \lim_{s\to 0}\left[s\,G(s)C(s)\right] = 20$$

$$e_{ss} = 0.05$$

Figure 2.9: Final compensator & Theoretical e_{ss} for unit ramp input

Part 3: PID Compensator Design

Outline:

(a) Design a PID controller that satisfies the following: (can use tuning rule or MATLAB tools)

- (i) Maximal overshoot; $\mathbf{M_P < 15\%}$
- (ii) Settling time (5%); $\mathbf{t_s < 8}$ **seconds**

(b) Use Simulink to simulate the **compensated** close-loop system

- (i) Verify that the design meets the objectives
- (ii) Include a screenshot of the constructed system diagram and its step response

Work/Results:

```
%% System Setup
s = tf('s');

G = (s+5)/(s)/(s+1)/(2*s+1);

%% Ziegler-Nicholas Tuning Method
% Kp = 9/35;
% Ti = 3.72;
% Td = 0.93;
% C = Kp*(1 + 1/(Ti*s) + Td*s);

%% pidTuner results
Kp = 0.26239;
Ti = 17.6628;
Td = 4.4157;
C = Kp*(1 + 1/(Ti*s) + Td*s);

%% Transfer Functions
OLTF = G*C;
CLTF = feedback(OLTF, 1);

%% Operations
%pidTuner(G)
step(CLTF)    % Step response
grid on

stepinfo(CLTF,"SettlingTimeThreshold", 0.05)
```

Figure 3.1: MATLAB code

Coefficients	Tuned
Kp	0.26239
Ti	17.6628
Td	4.4157

Table 3.1: Tuned PID parameters using pidTuner tool

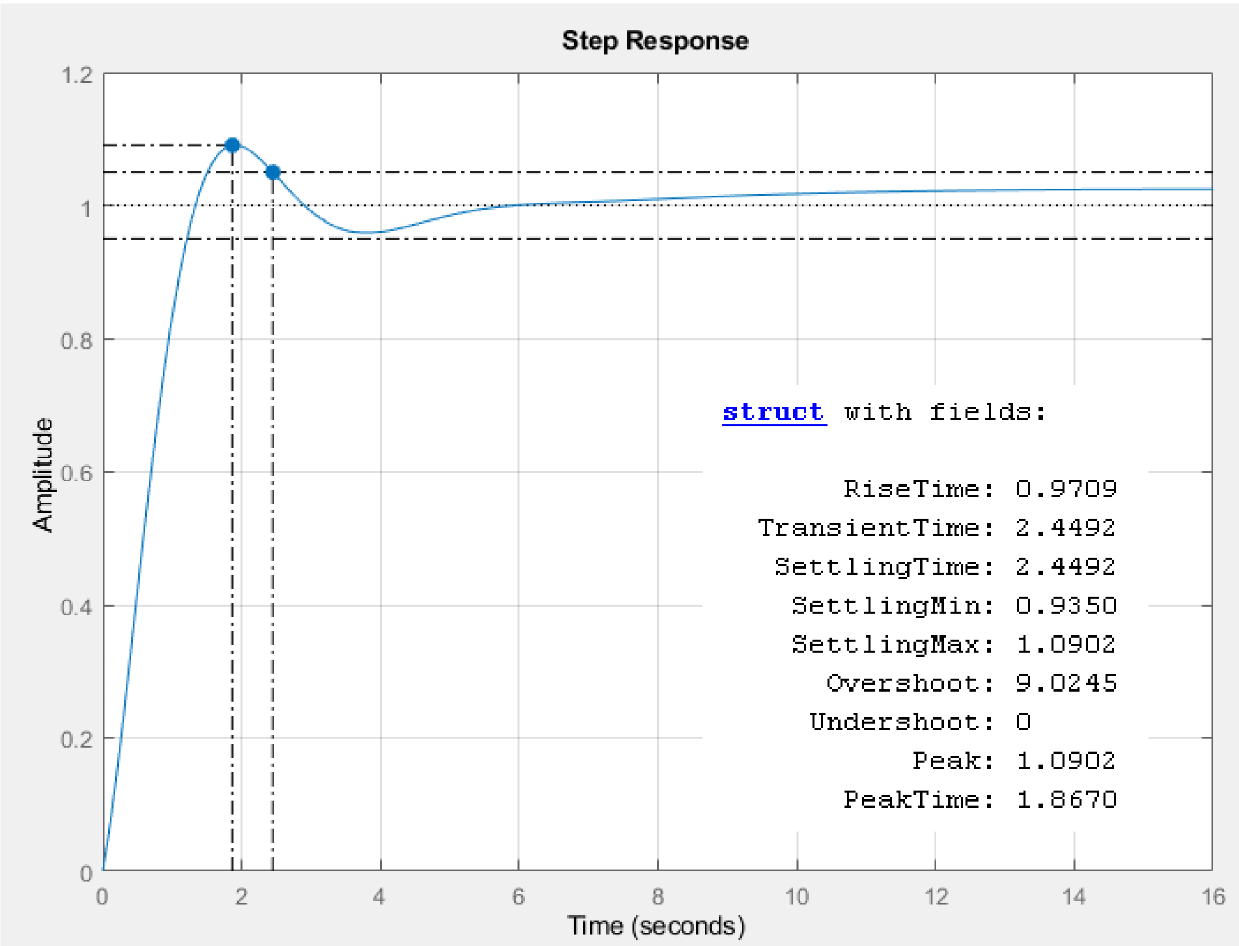


Figure 3.2: Step response of **compensated** system & stepinfo

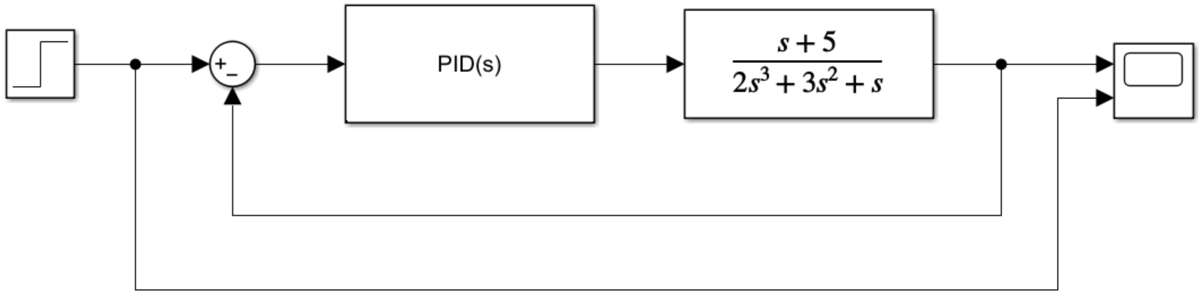


Figure 3.3: Simulink model

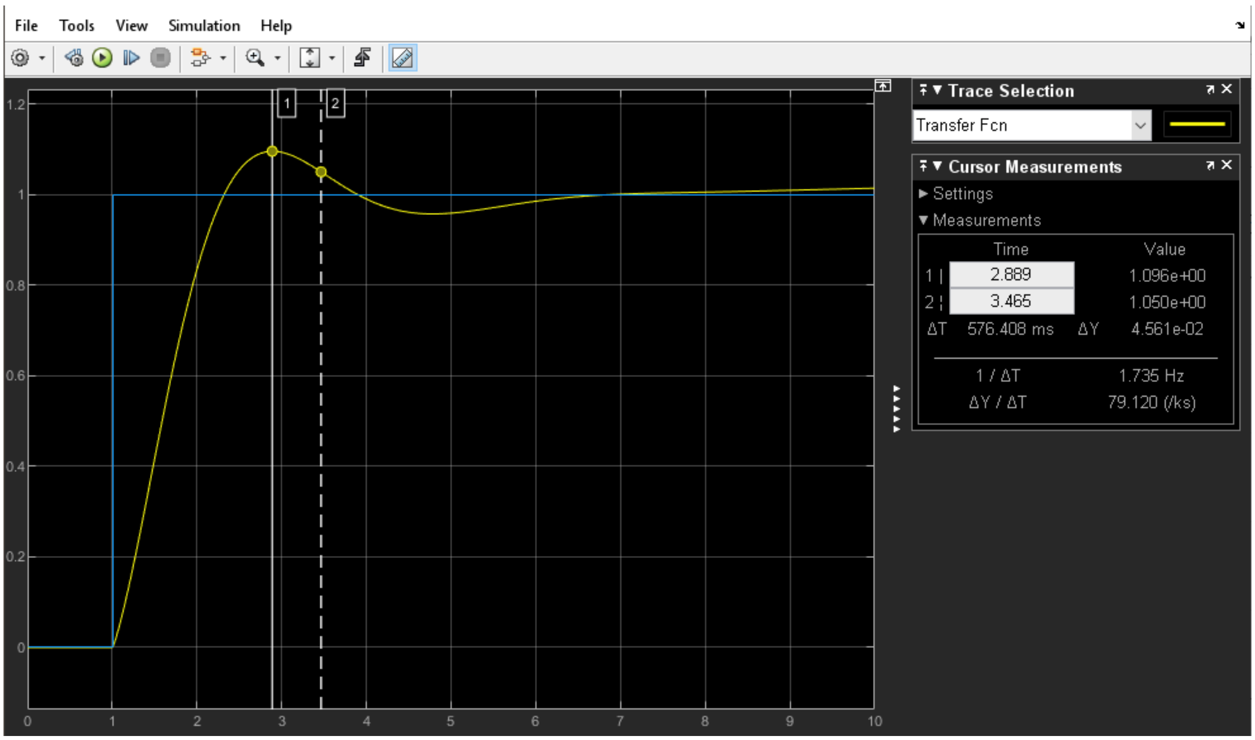


Figure 3.4: Step response in simulink

Parameter	**Measured Value**
`Maximal Overshoot,` M_p:	`9.6%`
`Settling Time,` t_s:	`3.465 seconds`

Table 3.2: Measured **overshoot** and **settling time** from simulink step response

(a) I began by designing a PID controller using the Ziegler-Nichols 2nd Method (coefficients shown in Figure 3.1); however, after looking at the step response, it was clear that this compensator would not meet the design objectives. Therefore, I used the pidTuner tool that MATLAB provides. With this tool I was able to satisfy the design objectives (Figure 3.2). The coefficients found with the pidTuner are shown in Table 3.1.

(i) **PID Controller**:

➢ $C(s) = 0.26239 \times (1 + \frac{1}{17.6628s} + 4.4157s)$

(b) Using Simulink, I created a model of the system. For the PID controller, I used the coefficient found previously with pidTuner.

(i) Table 3.2 above shows that the simulated step response meets the design requirements

➢ Step response can be found in Figure 3.4

Part 4: Frequency Response Design

Outline:

(a) Plot the **Bode** diagram and **Nyquist** plot of the **uncompensated OLTF**, G(s)

(i) Determine gain and phase margin if possible

(b) Design C(s) so that the CLTF meets the following requirements:

(i) Phase margin; **PM ≥ 40°**

(ii) e_{ss} **< 0.1** *(unit ramp input)*

(c) Plot the Bode diagram and Nyquist plot of the **compensated OLTF**, C(s)G(s)

➢ Use the C(s) from **(b)**

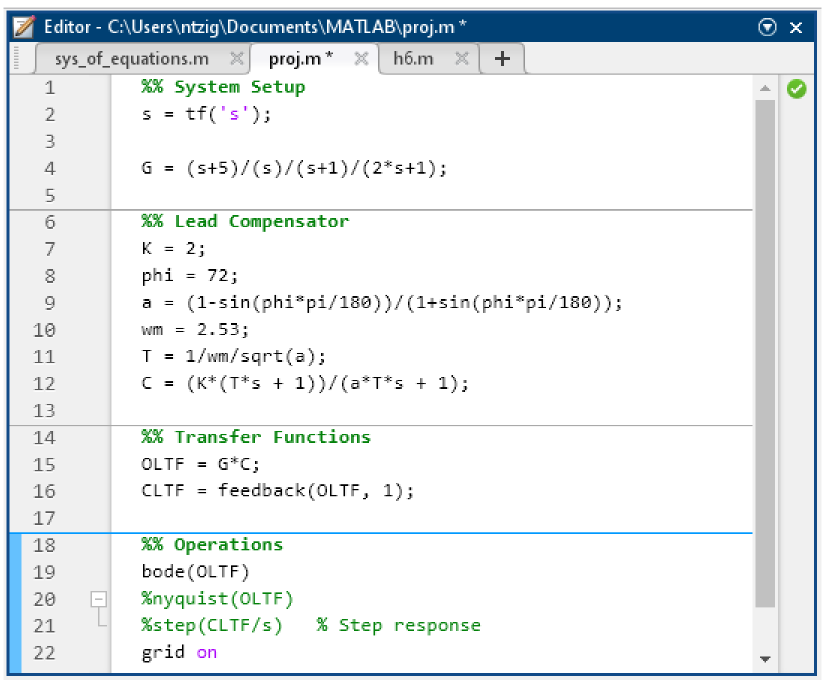
(ii) Plot the follow:

➢ Unit step response

➢ Unit ramp response

(iii) Verify that the plots meet the design objectives

Work/Results:




```
%% System Setup
s = tf('s');

G = (s+5)/(s)/(s+1)/(2*s+1);

%% Lead Compensator
K = 2;
phi = 72;
a = (1-sin(phi*pi/180))/(1+sin(phi*pi/180));
wm = 2.53;
T = 1/wm/sqrt(a);
C = (K*(T*s + 1))/(a*T*s + 1);

%% Transfer Functions
OLTF = G*C;
CLTF = feedback(OLTF, 1);

%% Operations
bode(OLTF)
%nyquist(OLTF)
%step(CLTF/s)    % Step response
grid on
```

Figure 4.1: MATLAB code

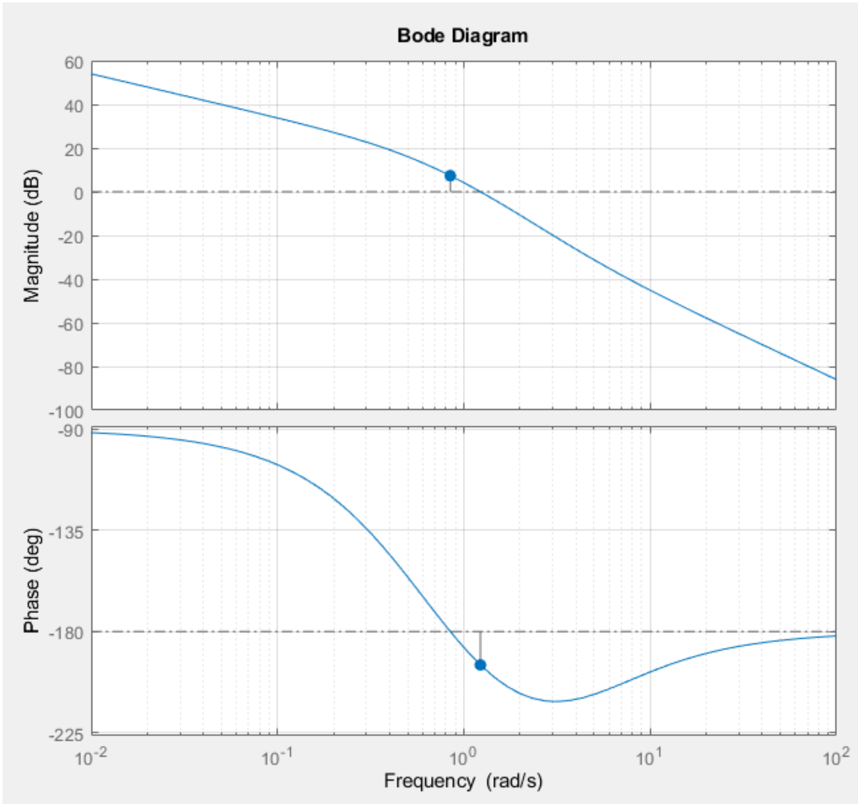


Figure 4.2: Bode diagram of **uncompensated** system

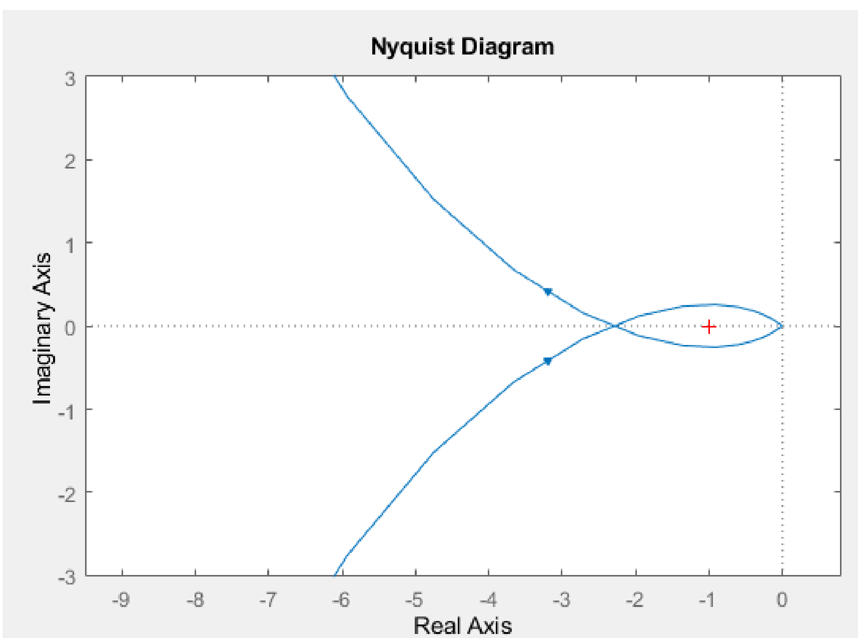


Figure 4.3: Nyquist plot of **uncompensated** system

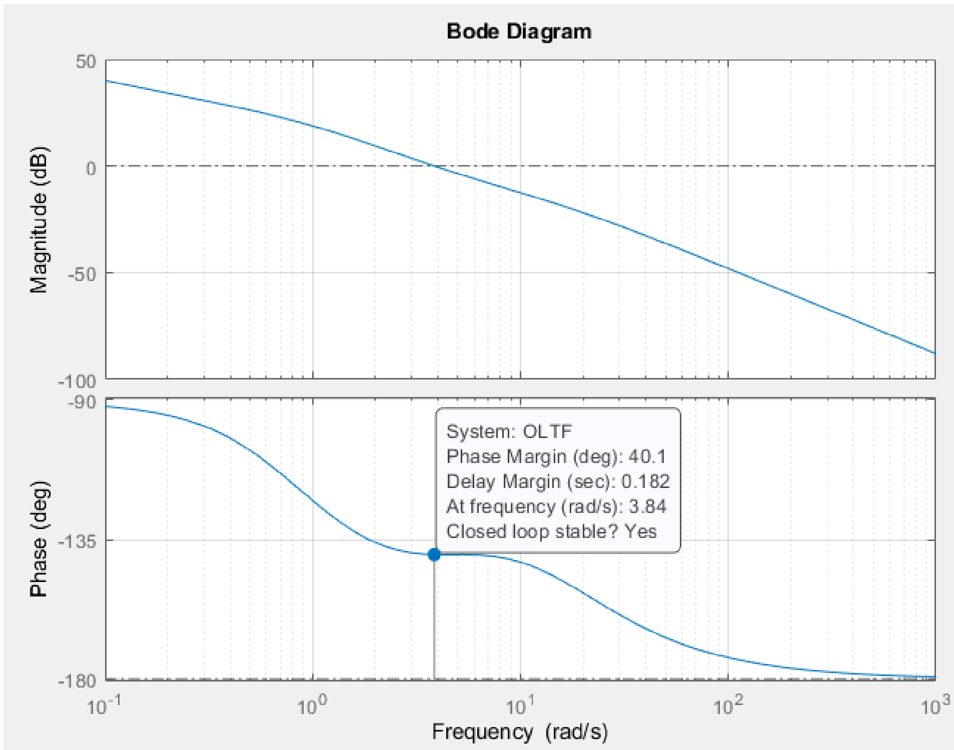


Figure 4.4: Bode diagram of the **compensated** system

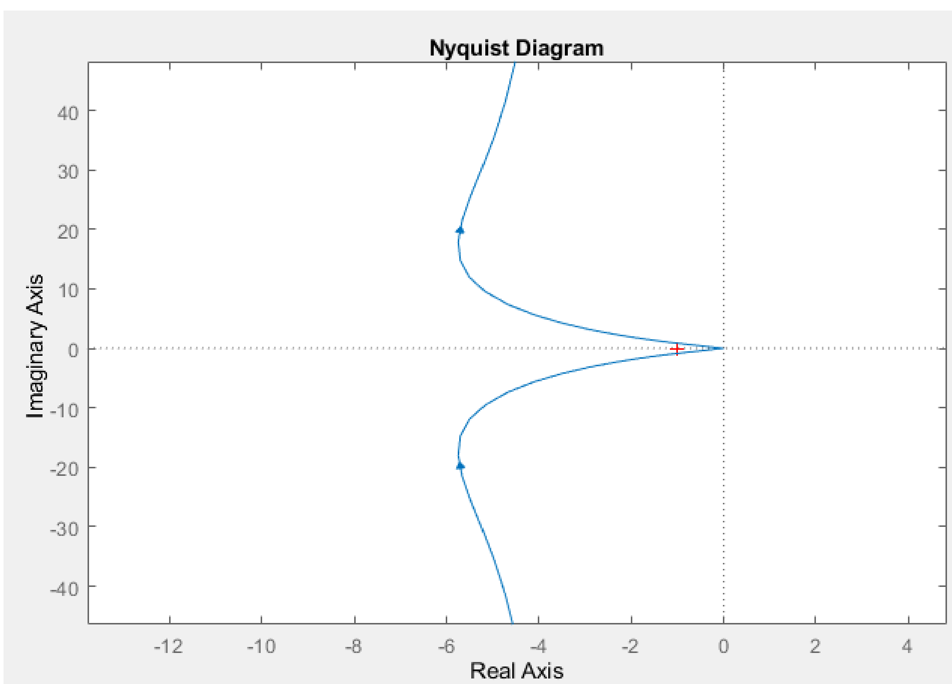


Figure 4.5: Nyquist plot of the **compensated** system

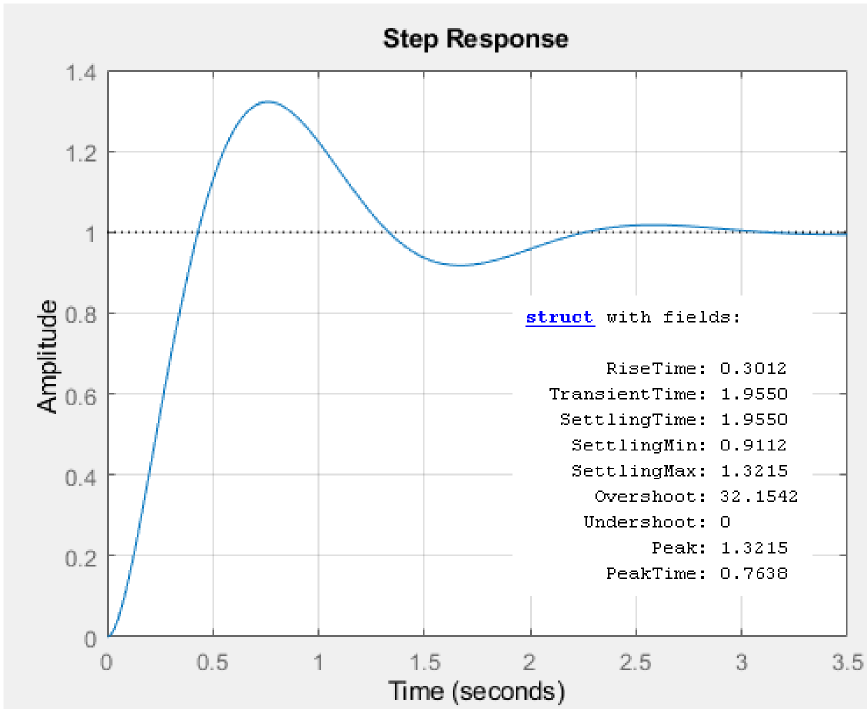


Figure 4.6: Step response of **compensated** system

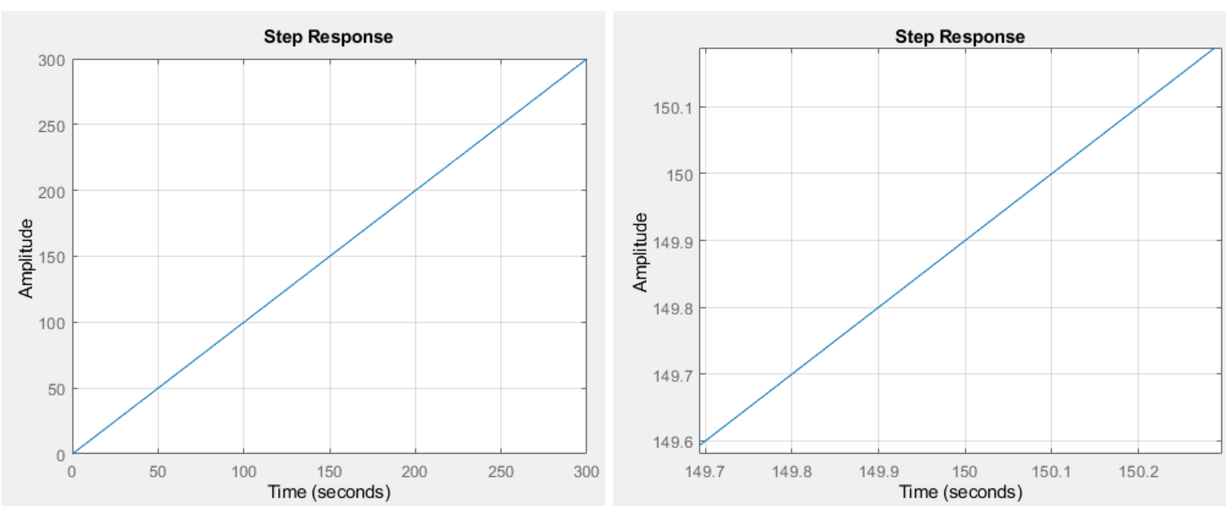


Figure 4.7: Ramp response of **compensated** system (*image on the right show* e_{ss})

(a) **Uncompensated** open-loop transfer function:

(i) **Bode** diagram shown in Figure **4.2**

➢ GM and PM are both negative, showing that the system is unstable

(ii) **Nyquist** plot shown in Figure **4.3**

(b) When design C(s), I initially started with a phase lead of: $\phi_m = (40° - (-14.9°)) = 55°$. However, after looking at the resultant bode diagram, it was clear that I needed to increase the amount of phase lead to meet the requirement: PM ≥ 40°. After some experimentation, I landed on a phase lead of $\phi_m = 72°$.

➢ $C(s) = 2 \times \frac{2.4956s+1}{0.0626s+1}$

(ii) Phase margin can be seen in Figure 4.4 above

(iii) e_{SS} can be observed in Figure 4.7 above

(c) Plot the Bode diagram and Nyquist plot of the **compensated OLTF**, C(s)G(s)

(i) Unit step response shown in Figure 4.6

(ii) Unit ramp response shown in Figure 4.7

(iii) Does the system meet the objectives?

- ◻ Phase Margin ≥ 40°
  - ➢ The bode diagram shown in Figure 4.4 shows a **PM = 40.1°**
- ◻ $e_{SS} \leq 0.1$
  - ➢ The DC gain, K, for C(s) was chosen to be 2 since the DC gain of G(s) was 5; therefore, $\mathbf{K_V = 10}$ & $\mathbf{e_{SS} = 0.1}$
  - ➢ The e_{SS} can be confirmed in Figure 4.7