Faculty of Enigeering

Cairo University

Digital Control System

MATLAB Control Project

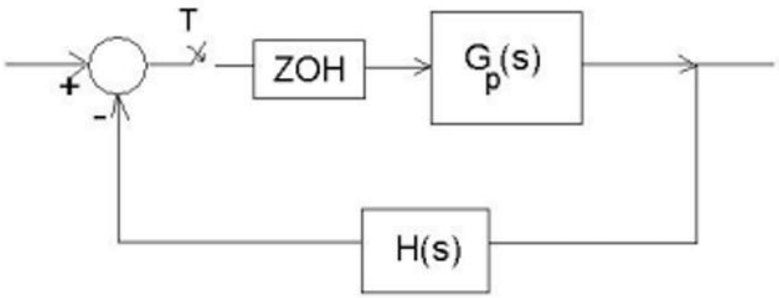
Presented for **ELC 4040 - Fall 2024**

Presented to:

Dr. Meena Elia Samouil Girgis

T.A. Saleh Ramadan

(4th Year Electronics and Electrical Communication Engineers)	
مجدي أحمد عباس عبد الحميد الابرق	Sec: 3 / I.D: 9210899 / BN: 32 / StdNo: 137
محمد إبراهيم محمد علي	Sec: 3 / I.D: 9210906 / BN: 34 / StdNo: 139



MATLAB Control Project

System Before Compensator

S to W Domain Conversion

- Continuous Forward Loop Gain & Backward Loop Gain

```
G_cont =

        4
  ---------------
  s^2 + 2.5 s + 1
```

```
H_cont =

        1
  ----------
  0.05 s + 1
```

- Continuous to Discrete Open Loop Gain (Z Transform) with $T_{sampling} = 0.1$

```
GH_disc =

  0.008086 z^2 + 0.01983 z + 0.002663
  -----------------------------------
  z^3 - 1.905 z^2 + 1.018 z - 0.1054

Sample time: 0.1 seconds
```

- Continuous to Discrete Closed Loop Gain (Z Transform)

```
Closed_Loop =

  0.008086 z^2 + 0.01983 z + 0.002663
  -----------------------------------
  z^3 - 1.897 z^2 + 1.038 z - 0.1027
```

- Get From the Characteristic Equation:
  Poles, Damping Ratio, Time Constant & Natural Frequency

Pole	Magnitude	Damping	Frequency (rad/seconds)	Time Constant (seconds)
8.86e-01 + 1.75e-01i	9.03e-01	4.65e-01	2.20e+00	9.77e-01
8.86e-01 - 1.75e-01i	9.03e-01	4.65e-01	2.20e+00	9.77e-01
1.26e-01	1.26e-01	1.00e+00	2.07e+01	4.83e-02

- Get Transient Parameters: (Rise Time, Settling Time, Overshoot, Peak Time)

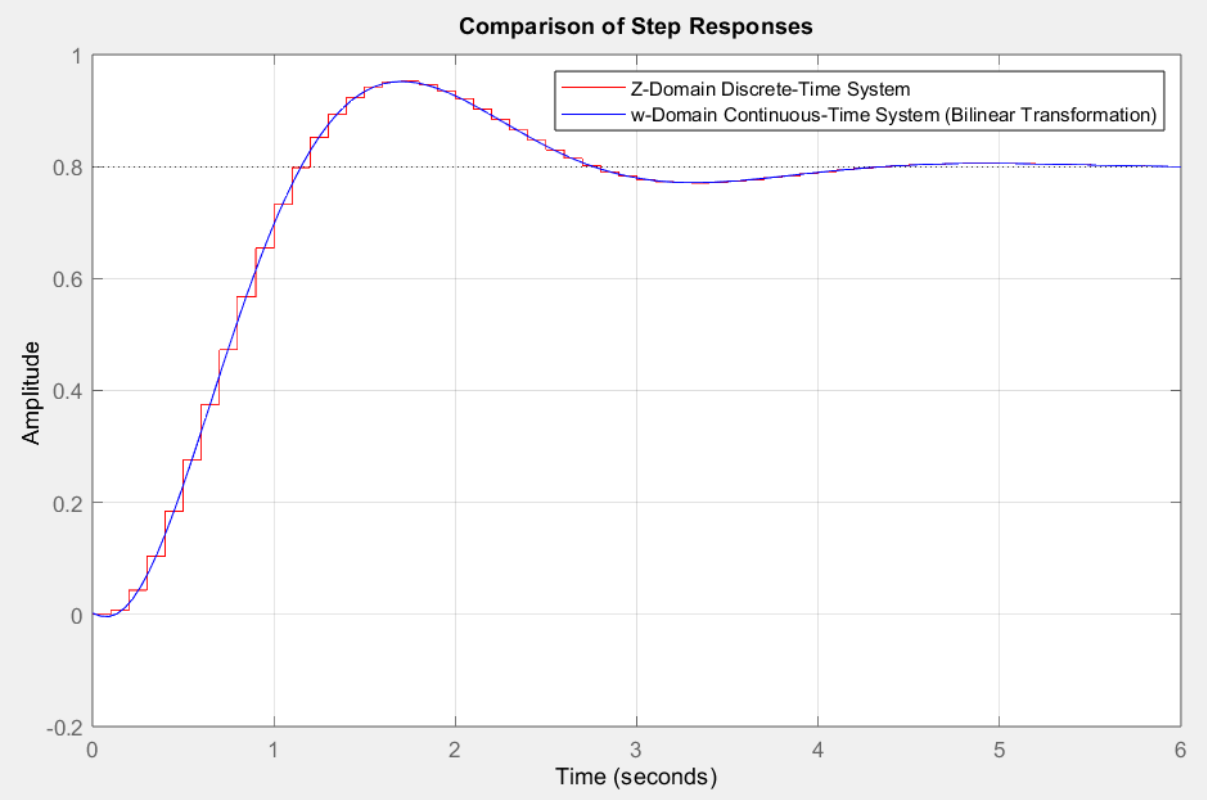
```
          RiseTime: 0.7000
      SettlingTime: 3.9000
       SettlingMin: 0.7320
       SettlingMax: 0.9525
         Overshoot: 19.0584
        Undershoot: 0
              Peak: 0.9525
          PeakTime: 1.7000
```

- Bilinear Transformation (Discrete z-domain to continuous w-domain)

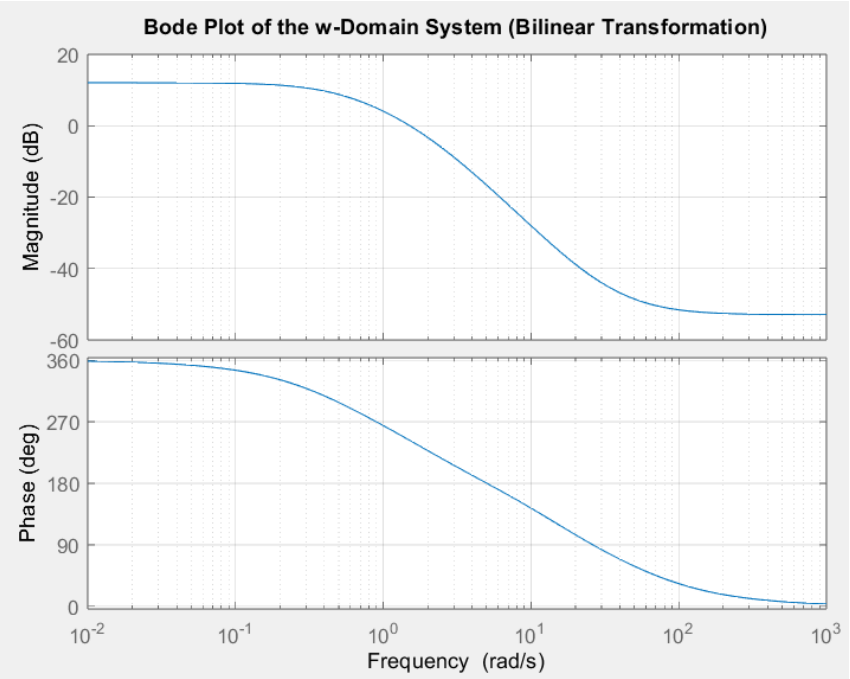
```
Transfer Function in w-domain (Bilinear Transformation):

GH_w =

  0.002253 s^3 - 0.09891 s^2 - 1.959 s + 60.71
  ---------------------------------------------
       s^3 + 17.73 s^2 + 38.97 s + 15.18
```

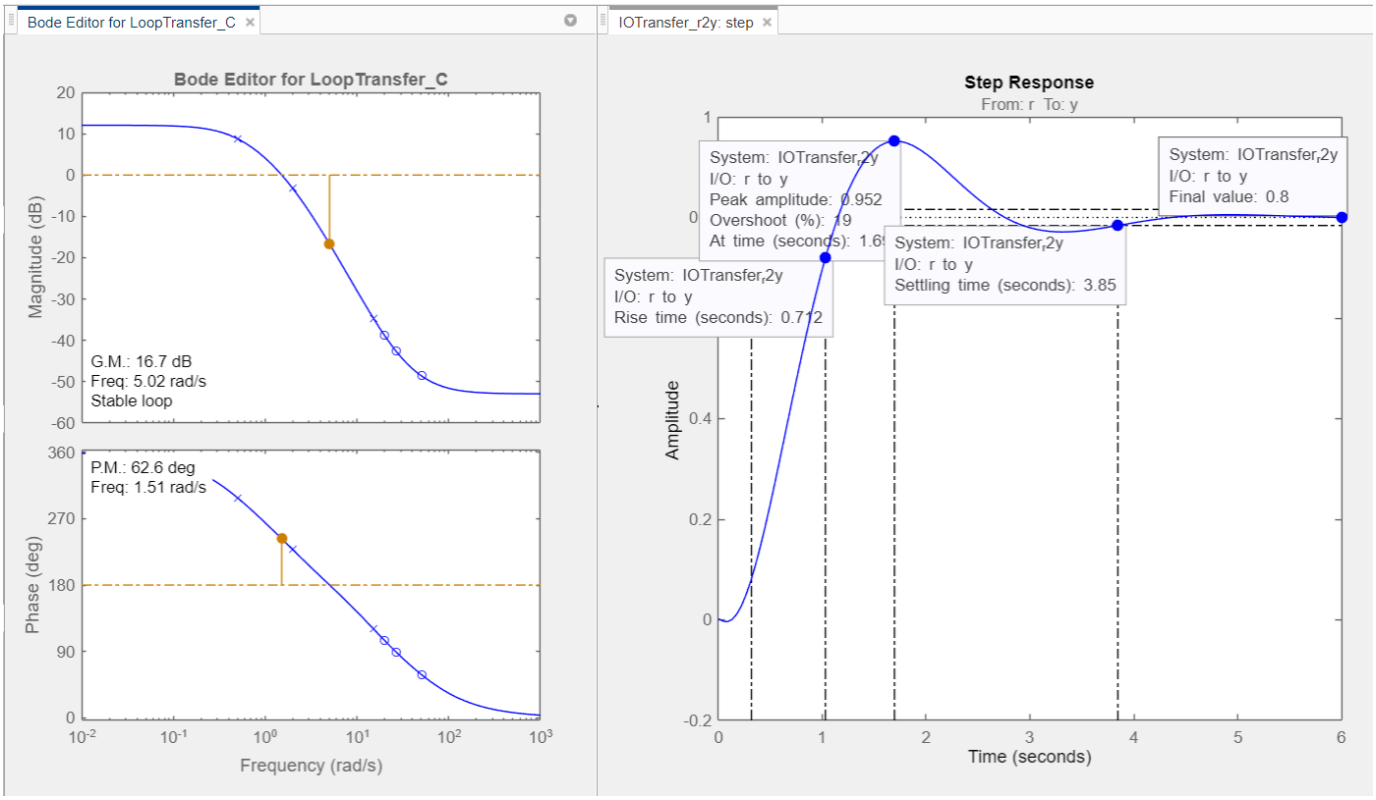
Step Response

- Step Response Showing Comparison between Z-Domain and W-Domain



- W-Domain Bode Plot





- W-Domain Open Loop Gain with unit feedback system for steady-state error calculation

```
GH_w_open =

  0.002253 s^3 - 0.09891 s^2 - 1.959 s + 60.71
  ---------------------------------------------
       s^3 + 17.73 s^2 + 38.97 s + 15.18
```

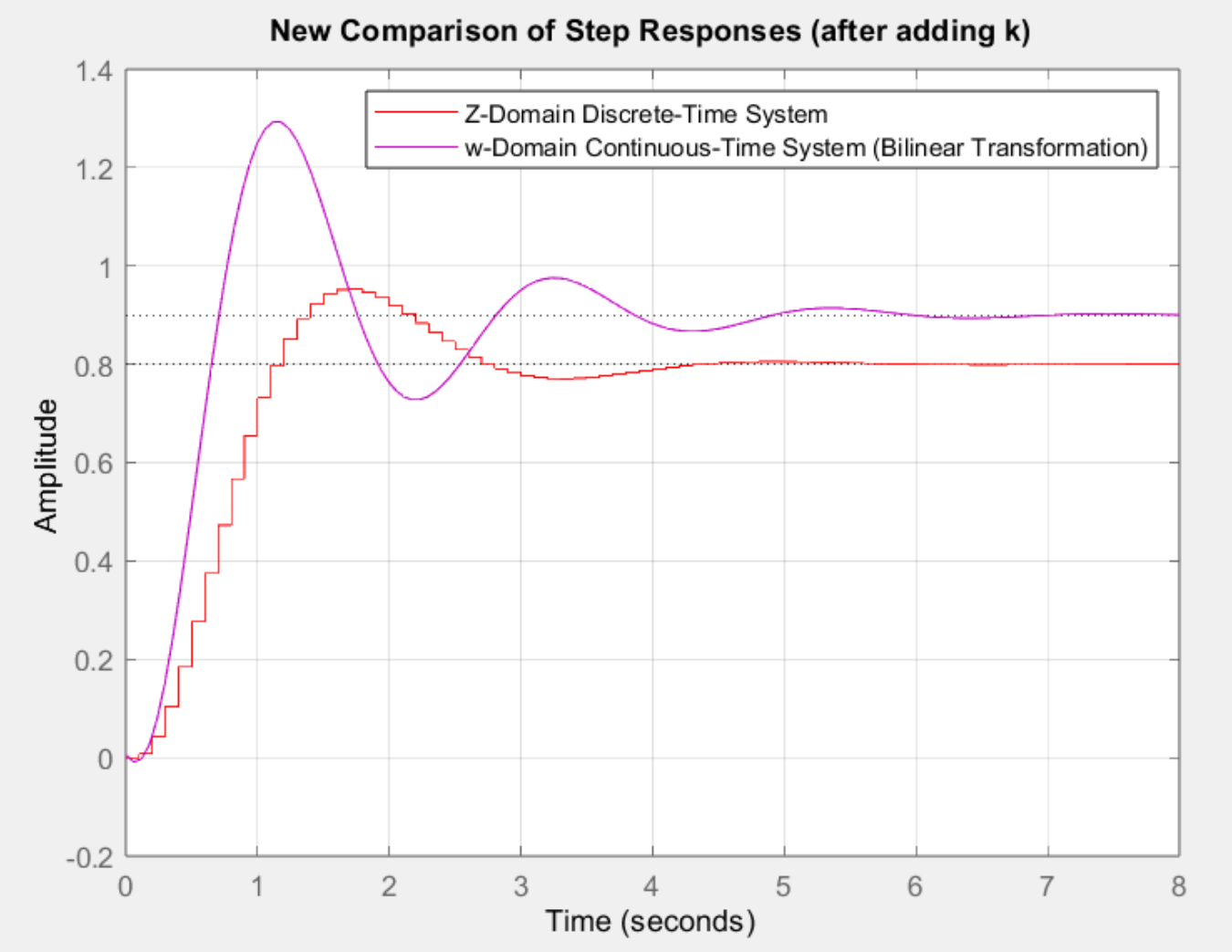
- Stability Metrics

```
*** Stability Metrics:
Gain Margin (GM): 16.69 dB
Phase Margin (PM): 62.64 degrees
Gain Crossover Frequency (w_gc): 5.02 rad/s
Phase Crossover Frequency (w_pc): 1.51 rad/s

 Open-Loop DC Gain (K_dc): 4.0000
System Type: 0
Steady-State Error (SSE) for Step Input: 0.2000
```

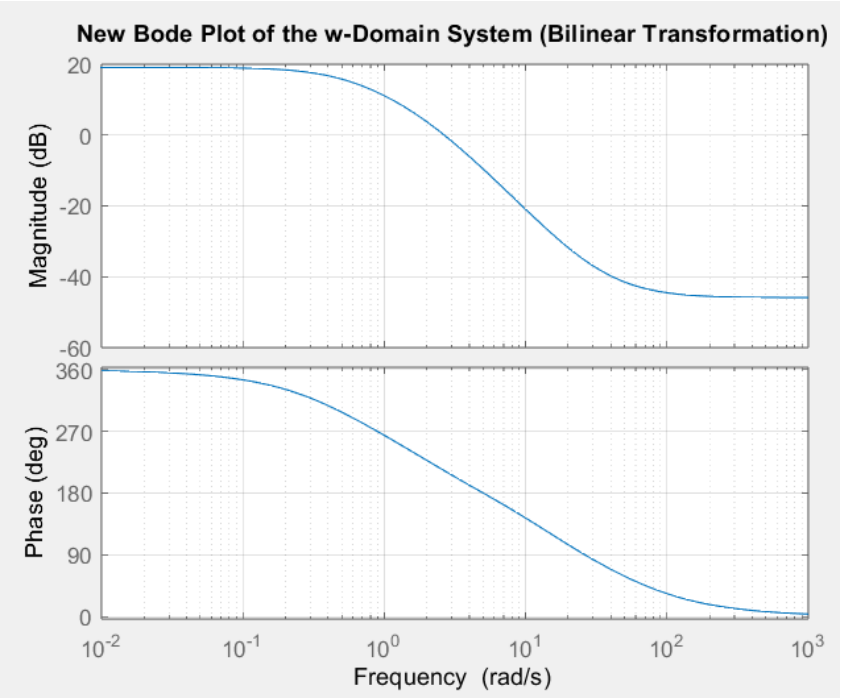
- Adding **K (DC gain)** to the cont. open loop gain (GH) in w-domain

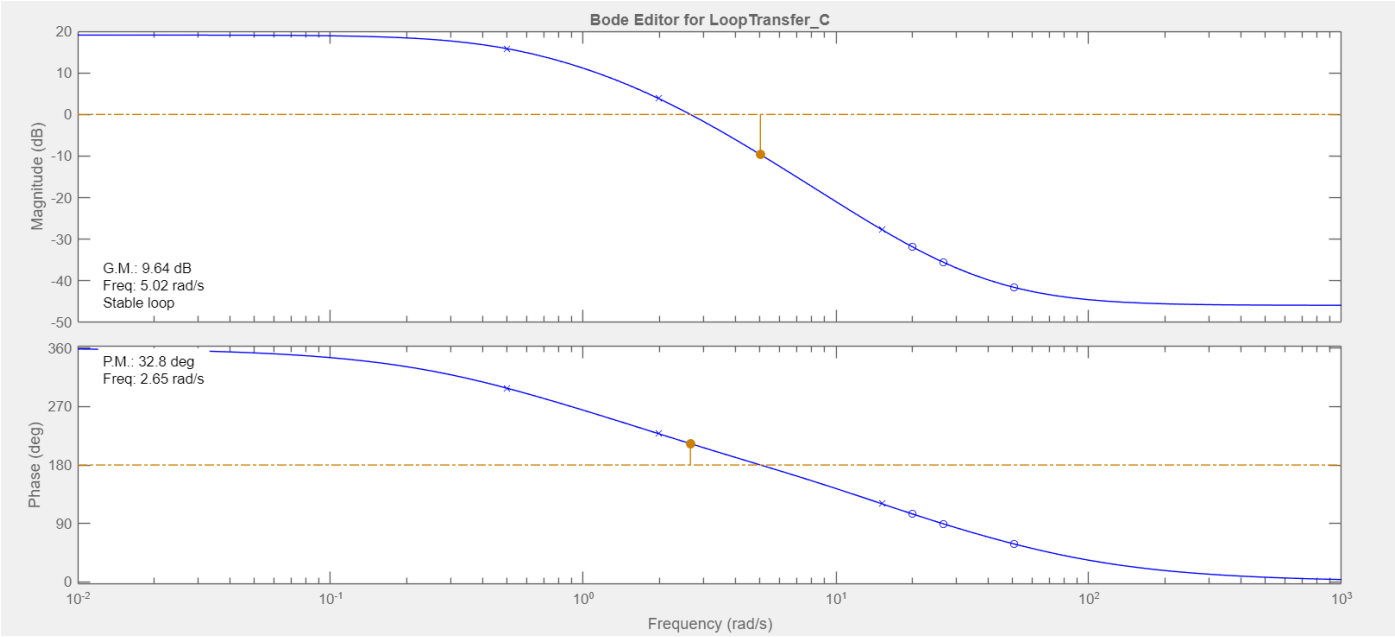
```
new_GH_w =

k =                0.00507 s^3 - 0.2226 s^2 - 4.407 s + 136.6
                   ------------------------------------------
                      s^3 + 17.73 s^2 + 38.97 s + 15.18

    2.2500
```

- New Step Response Showing Comparison between Z & W Domains.



- New W-Domain Bode Plot





- W-Domain Open Loop Gain with unit feedback system for steady-state error calculation

```
new_GH_w_open =

  0.00507 s^3 - 0.2226 s^2 - 4.407 s + 136.6
  ------------------------------------------
     s^3 + 17.73 s^2 + 38.97 s + 15.18
```

- New Stability Metrics

```
*** New Stability Metrics:
Gain Margin (GM): 9.64 dB
Phase Margin (PM): 32.80 degrees
Gain Crossover Frequency (w_gc): 5.02 rad/s
Phase Crossover Frequency (w_pc): 2.65 rad/s

 New Open-Loop DC Gain (K_dc): 9.0000
New Steady-State Error (SSE) for Step Input: 0.1000
```

- **Comments Before Adding Compensator:**
  - **Before Adding k:**

$$G_p(s) = \frac{4}{(2s+1)(0.5s+1)} \quad , \quad H(s) = \frac{1}{(0.05s+1)} \quad , \quad \lim_{s\to 0} G_p(s)\,H(s) = 4$$

  - **After Adding k to achieve $e_{ss} = 10\,\% = 0.1$**

$$Type('0') \rightarrow e_{ss} = \lim_{s\to 0} \frac{1}{1+k_p} = 0.1 \rightarrow k_p = \mathbf{9} = \mathbf{4k} \rightarrow \mathbf{k = 2.25}$$

Therefore, this gain may make worse transient response and be more oscillatory, since damping ratio will decrease, Max Peak will increase and may make the system unstable, but it achieves the required $e_{ss} = 10\,\%$. As Shown in the Figures above after k increase:

$$|k| \uparrow \rightarrow PM \downarrow \Rightarrow Stability \downarrow$$

  - **Solution:**

Since System is Stable as $(\omega_{gc} > \omega_{pc})$ and there is no requirement on ω_{gc}, So add $+ve$ phase to enhance PM, where: $PM_{after\,k} = 32.8° < 50°$
As $PM_{required} \geq 50°$ so, we need to increase PM to improve stability
Therefore, **Use Lead Compensator.**

Criteria	Lead Compensator	Lag Compensator
Transient Response	Improves (faster rise/settling time)	Slightly slows response
Phase Margin	Increases (positive phase lead)	Decreases (phase lag)
Stability	Improves	May degrade
Frequency Range	High-frequency improvement	Low-frequency improvement
Steady-State Error	Minimal improvement	Significant improvement
Overshoot	Reduces	May increase slightly

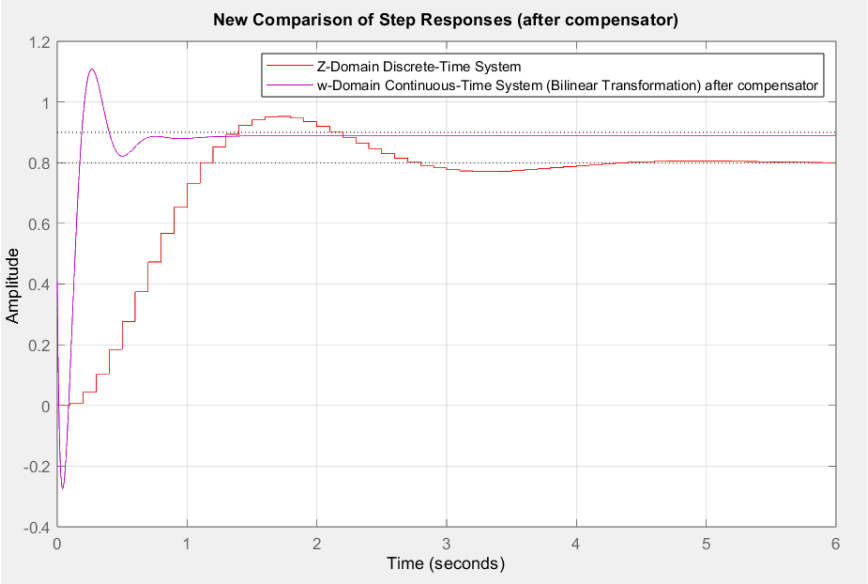
System After Compensator

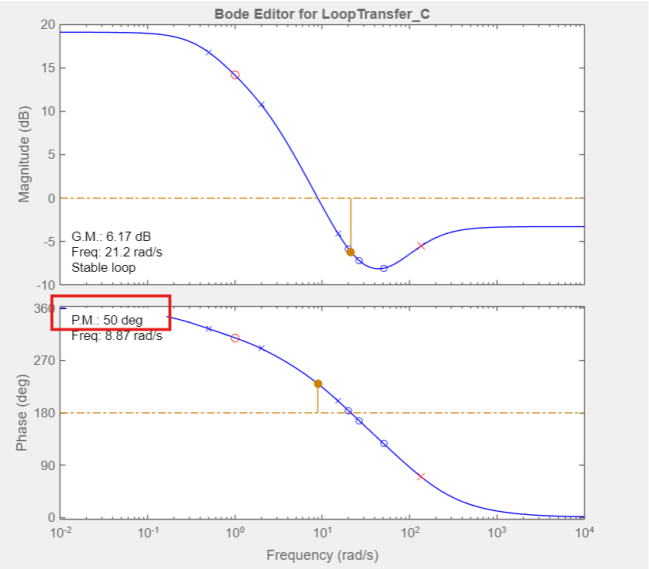
Use Lead Compensator

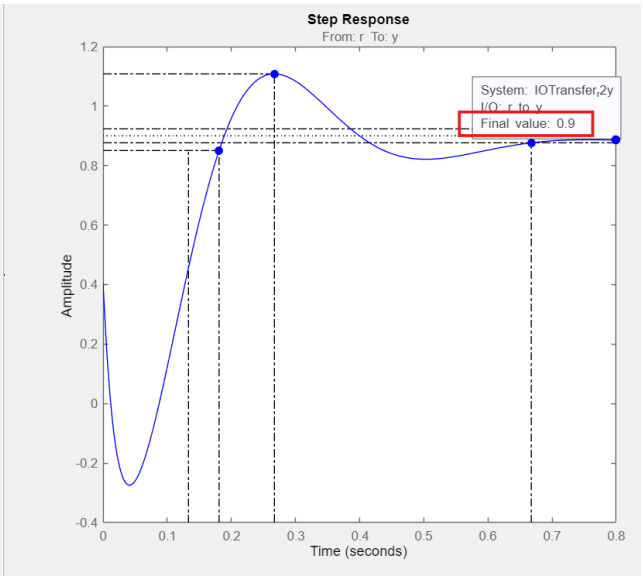
- Lead Compensator Loop Gain

```
comp_tf =                comp_GH_W =

  135.23 (s+1)          0.68567 (s-50.54) (s-20) (s+26.67) (s+1)
  ------------          ----------------------------------------
   (s+135.2)            (s+15.23) (s+1.993) (s+0.4999) (s+135.2)
```

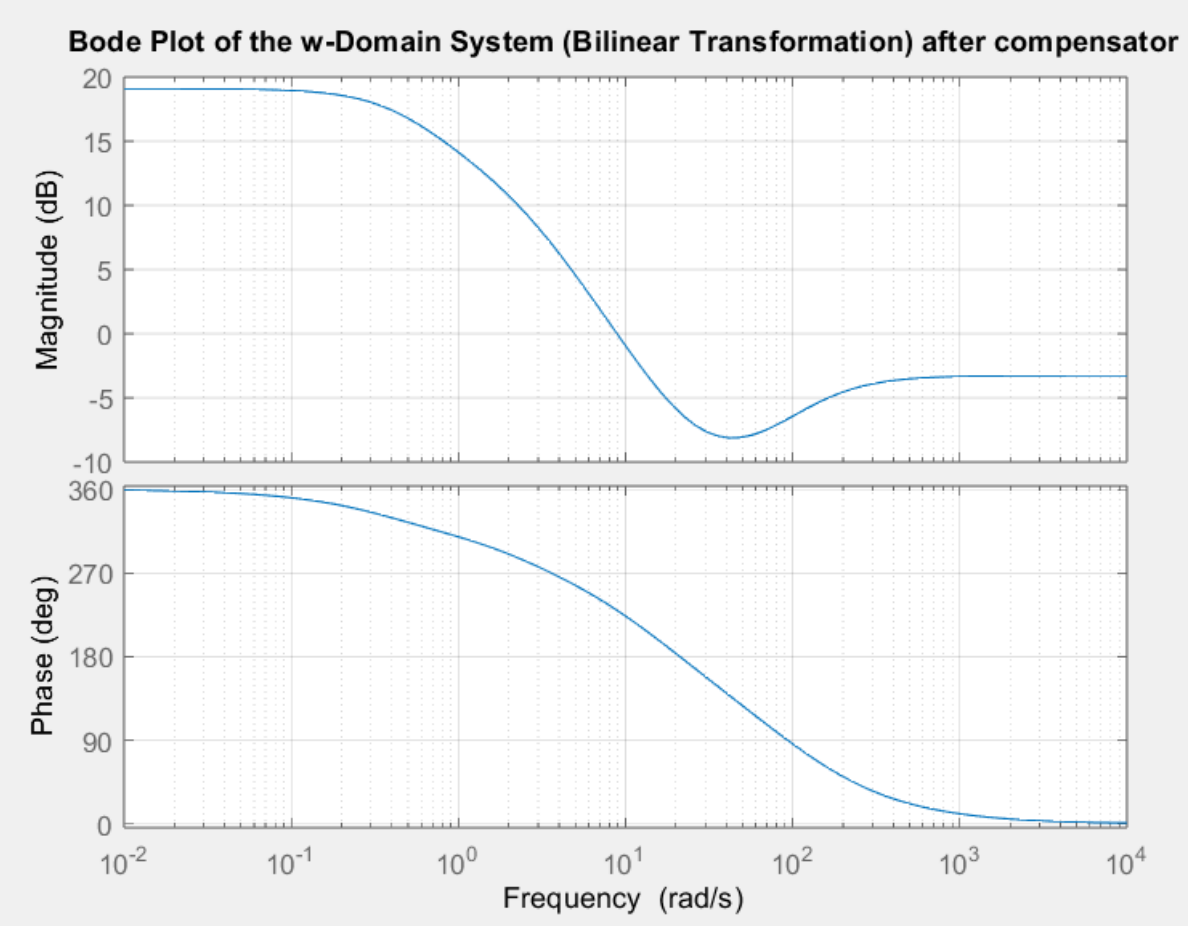
- Step Response Showing Comparison between Z-Domain and W-Domain After Lead Compensator







- W-Domain Bode Plot After Lead Compensator

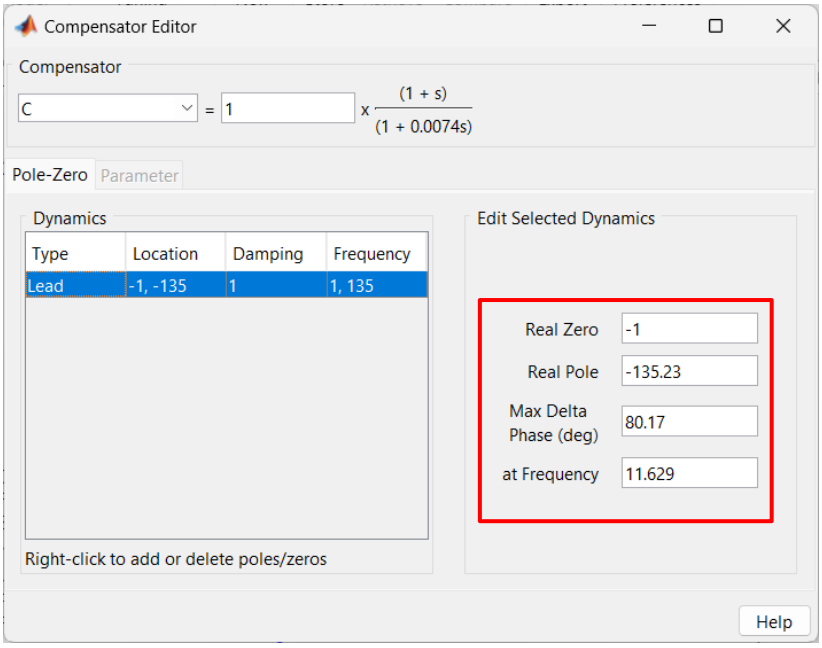


- W-Domain Open Loop Gain with unit feedback system for steady-state error calculation

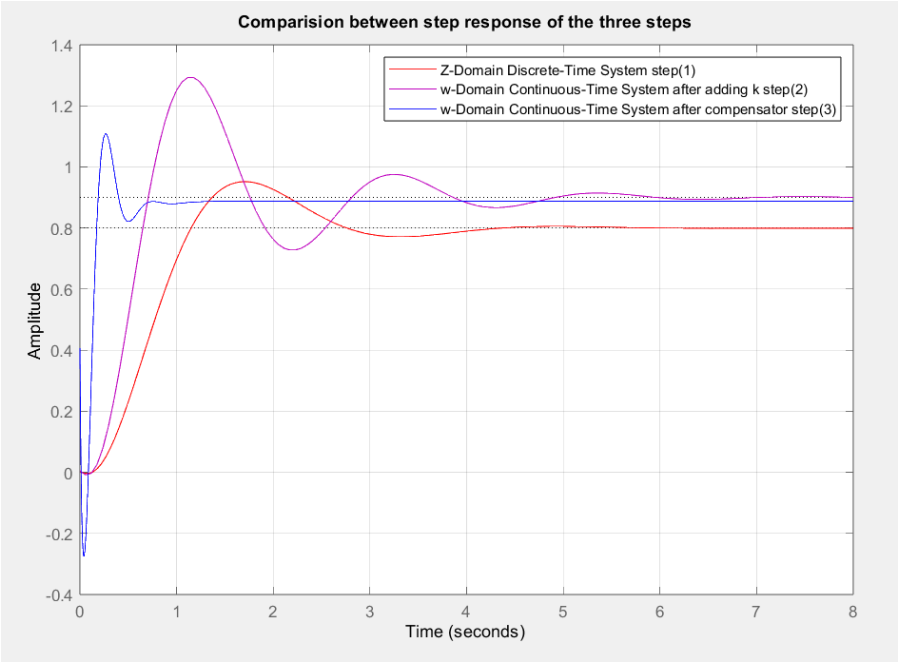
```
comp_GH_W_open =

  0.68567 (s-50.54) (s-20) (s+26.65) (s+1)
  ----------------------------------------
  (s+15.23) (s+1.993) (s+0.4999) (s+135.2)
```

- Stability Metrics After Lead Compensator

```
*** After Compensator Stability Metrics:
Gain Margin (GM): 6.17 dB
Phase Margin (PM): 50.00 degrees
Gain Crossover Frequency (w_gc): 21.16 rad/s
Phase Crossover Frequency (w_pc): 8.87 rad/s

 New Open-Loop DC Gain (K_dc): 9.0000
After Compensator Steady-State Error (SSE) for Step Input: 0.1000
```



- Bode Plots Summary:



- **Comments After Adding Lead Compensator:**

  As Shown in the Above Figures After Adding Lead Compensator:

  - **Increased Stability Margin**

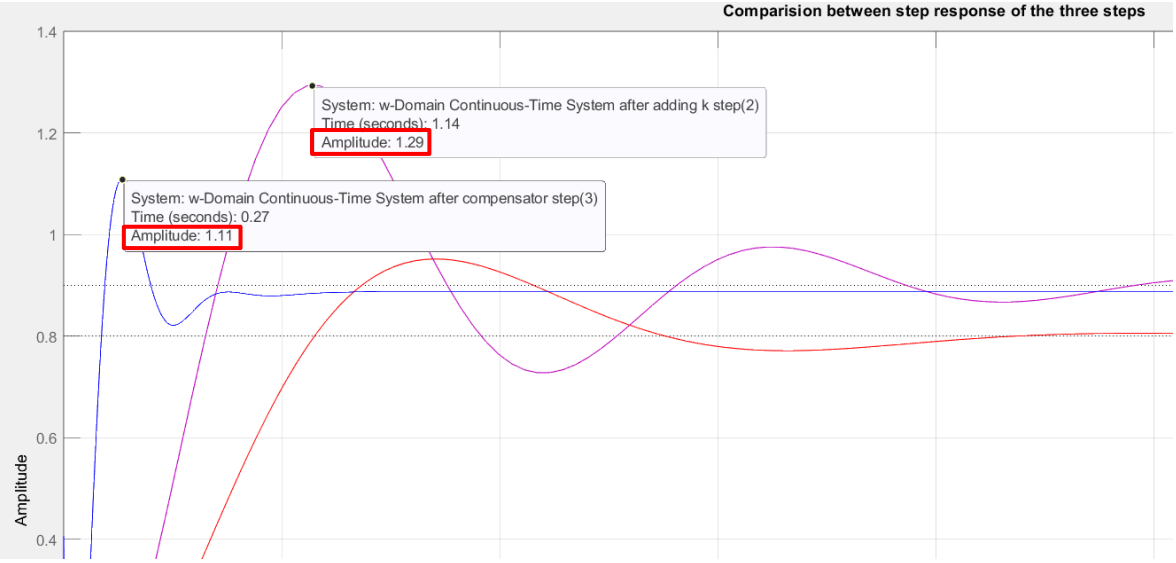
  - A lead compensator adds phase lead (positive phase) to the system, which helps counteract the phase lag introduced by the plant or other components.
  - This improves the system's phase margin $(PM = PM_{req} = 50°)$, Enhance the Stability.

- **Faster Response**

- By increasing the bandwidth of the system, a lead compensator speeds up the transient response. This allows the system to reach the desired state more quickly (Reach $e_{ss} = 0.1$ Faster than Before).

- **Improved Damping**

- A lead compensator reduces Overshoot by $\frac{1.29-1.11}{1.29} \times 100\ \% \approx 14\ \%$ and oscillations in the response, improving the damping of the system.



- **Enhanced Gain Crossover Frequency**

- By increasing the gain crossover frequency $(\omega_{gc} = 5.02 \rightarrow 21.16)$ (where the open-loop gain is 1), the lead compensator enhances the speed and responsiveness of the control system.

```
*** New Stability Metrics:
Gain Margin (GM): 9.64 dB
Phase Margin (PM): 32.80 degrees
Gain Crossover Frequency (w_gc): 5.02 rad/s
Phase Crossover Frequency (w_pc): 2.65 rad/s

 New Open-Loop DC Gain (K_dc): 9.0000
New Steady-State Error (SSE) for Step Input: 0.1000

*** After Compensator Stability Metrics:
Gain Margin (GM): 6.17 dB
Phase Margin (PM): 50.00 degrees
Gain Crossover Frequency (w_gc): 21.16 rad/s
Phase Crossover Frequency (w_pc): 8.87 rad/s

 New Open-Loop DC Gain (K_dc): 9.0000
After Compensator Steady-State Error (SSE) for Step Input: 0.1000
```

MATLAB Code

```
%% Define System Parameters
clc; clear all; close all; %#ok < CLALL >
% Continuous Forward Loop Gain
G_cont = tf([0 4], [1 2.5 1]);
% Continuous Backward Loop Gain
H_cont = tf([0 1], [0.05 1]);
% Sampling time
T = 0.1;
%% Conversion from S − domain to W − domain
% Continuous to Discrete Open Loop Gain (Z Transform)
% c2d(continuous loop gain, sampling time, ZOH block)
GH_disc = c2d(G_cont * H_cont, T, 'zoh');
% Discrete Closed Loop Gain
Closed_Loop = feedback(GH_disc, 1);
% Get From the Characteristic Equation: Poles, Damping Ratio, Time Constant & ωn
damp(Closed_Loop);
% Get Transient Parameters: (Rise Time, Settling Time, Overshoot, Peak Time)
stepinfo(Closed_Loop);
% Bilinear Transformation (Discrete z − domain to continuous w − domain)
GH_w = d2c(GH_disc, 'tustin');
% Display the w − domain transfer function
disp('Transfer Function in w − domain (Bilinear Transformation): ');
GH_w; % Open − loop gain in w − domain
%% Plot Step Response of both Z and W − domains
% Step Response Comparison
figure;
step(Closed_Loop, 'r'); % Step response of the discrete − time system (Closed loop)
hold on;
step(feedback(GH_w, 1), 'b'); % Step response of the w − domain system
legend('Z − Domain Discrete − Time System', 'w − Domain Continuous
        − Time System (Bilinear Transformation)');
title('Comparison of Step Responses');
grid on;
%% Bode Plot of W − domain
% Bode Plot of the w − domain system (open − loop)
figure('name', "Bode Plot of the w − domain system (open − loop)");
bode(GH_w); % Open − loop Bode plot in the w − domain
title('Bode Plot of the w − Domain System (Bilinear Transformation)');
grid on;
%% Calculate Stability Metrics (w_gc, w_pc, PM, GM)
[GM, PM, w_gc, w_pc] = margin(GH_w);
% Display results
disp(' *** Stability Metrics: ');
```

```
fprintf('Gain Margin (GM): %.2f dB\n', 20 * log10(GM)); % Convert GM to dB
fprintf('Phase Margin (PM): %.2f degrees\n', PM);
fprintf('Gain Crossover Frequency (w_gc): %.2f rad/s\n', w_gc);
fprintf('Phase Crossover Frequency (w_pc): %.2f rad/s\n', w_pc);
%% Steady - State Error (SSE) before adding K(dc gain) {step 1}
% Open - loop transfer function (GH_w)
s = tf('s');
GH_w_open = GH_w * 1; % Assume unit feedback system for steadystate error calculation
% Steady - State Error Calculation
K_dc = dcgain(GH_w_open); % Compute DC gain of the open - loop transfer function
fprintf('\n Open - Loop DC Gain (K_dc): %.4f\n', K_dc);
% Determine system type (number of poles at s = 0)
[num, den] = tfdata(GH_w_open, 'v'); % Get numerator and denominator
sys_type = sum(den == 0); % Count the number of integrators (poles at s = 0)
if sys_type == 0
  % Type 0 System (No integrators): SSE = 1 / (1 + K_dc)
  SSE = 1 / (1 + K_dc);
elseif sys_type == 1
  % Type 1 System (1 integrator): SSE = 0 for step input
  SSE = 0;
elseif sys_type >= 2
  % Type 2 or higher systems (>= 2 integrators): Perfect tracking for step input
  SSE = 0;
end
fprintf('System Type: %d\n', sys_type);
fprintf('Steady - State Error (SSE) for Step Input: %.4f\n\n', SSE);
% Steady - State Error (SSE) after adding K(dc gain) to the cont. open loop gain (GH) in w - domain {step 2}
k = 2.25;
new_GH_w = k * GH_w;
% Calculate New Stability Metrics (w_gc, w_pc, PM, GM)
[GM, PM, w_gc, w_pc] = margin(new_GH_w);
% Display results
disp(' *** New Stability Metrics: ');
fprintf('Gain Margin (GM): %.2f dB\n', 20 * log10(GM)); % Convert GM to dB
fprintf('Phase Margin (PM): %.2f degrees\n', PM);
fprintf('Gain Crossover Frequency (w_gc): %.2f rad/s\n', w_gc);
fprintf('Phase Crossover Frequency (w_pc): %.2f rad/s\n', w_pc);
% Open - loop transfer function (GH_w)
s = tf('s');
new_GH_w_open = new_GH_w * 1;
% Assume unit feedback system for steady - state error calculation
% Steady - State Error Calculation
K_dc = dcgain(new_GH_w_open); % Compute DC gain of the open - loop transfer function
fprintf('\n New Open - Loop DC Gain (K_dc): %.4f\n', K_dc);
% Determine system type (number of poles at s = 0)
[num, den] = tfdata(new_GH_w_open, 'v'); % Get numerator and denominator
sys_type = sum(den == 0); % Count the number of integrators (poles at s = 0)
```

```
if sys_type == 0
   % Type 0 System (No integrators): SSE = 1 / (1 + K_dc)
   new_SSE = 1 / (1 + K_dc);
elseif sys_type == 1
   % Type 1 System (1 integrator): SSE = 0 for step input
   new_SSE = 0;
elseif sys_type >= 2
   % Type 2 or higher systems (>= 2 integrators): Perfect tracking for step input
   new_SSE = 0;
end
% fprintf('System Type: %d\n', sys_type);
fprintf('New Steady - State Error (SSE) for Step Input: %.4f\n\n', new_SSE);
%% Plot new Step Response of both Z and W - domains
% Step Response Comparison
figure;
step(Closed_Loop, 'r'); % Step response of the discrete - time system (Closed loop)
hold on;
step(feedback(new_GH_w, 1), 'm'); % Step response of the w - domain system
legend('Z - Domain Discrete - Time System', 'w - Domain Continuous
            - Time System (Bilinear Transformation)');
title('New Comparison of Step Responses (after adding k)');
grid on;
%% Bode Plot of new W - domain (after adding k)
plot_bode_w_domain(new_GH_w);
%% Adding Lead Compensator in w - domain (step 3)
s = tf('s');
%comp_tf = (1 + s)/(1 + 0.0074s);
%comp_tf = zpk([-1], [-135.1351 ], []);
zeros = -1;
poles = -135.23;
gain = 135.23;
comp_tf = zpk(zeros, poles, gain);
comp_GH_W = new_GH_w * comp_tf;
% Calculate New Stability Metrics (w_gc, w_pc, PM, GM)
[GM, PM, w_gc, w_pc] = margin(comp_GH_W);
% Display results
disp(' *** After Compensator Stability Metrics: ');
fprintf('Gain Margin (GM): %.2f dB\n', 20 * log10(GM)); % Convert GM to dB
fprintf('Phase Margin (PM): %.2f degrees\n', PM);
fprintf('Gain Crossover Frequency (w_gc): %.2f rad/s\n', w_gc);
fprintf('Phase Crossover Frequency (w_pc): %.2f rad/s\n', w_pc);
% Open - loop transfer function (GH_w)
s = tf('s');
comp_GH_W_open = comp_GH_W * 1;
% Assume unit feedback system for steady - state error calculation
```

```
% Steady − State Error Calculation
K_dc = dcgain(comp_GH_W_open);
% Compute DC gain of the open − loop transfer function
fprintf('\n New Open − Loop DC Gain (K_dc): %. 4f\n', K_dc);
% Determine system type (number of poles at s = 0)
[num, den] = tfdata(comp_GH_W_open, 'v'); % Get numerator and denominator
sys_type = sum(den == 0); % Count the number of integrators (poles at s = 0)
if sys_type == 0
  % Type 0 System (No integrators): SSE = 1 / (1 + K_dc)
  comp_SSE = 1 / (1 + K_dc);
elseif sys_type == 1
  % Type 1 System (1 integrator): SSE = 0 for step input
  comp_SSE = 0;
elseif sys_type >= 2
  % Type 2 or higher systems (>= 2 integrators): Perfect tracking for step input
  comp_SSE = 0;
end
% fprintf('System Type: %d\n', sys_type);
fprintf('After Compensator Steady − State Error (SSE) for Step Input: %. 4f\n', comp_SSE);
%% Plot new Step Response of both Z and W − domains after adding compensator
% Step Response Comparison
figure;
step(Closed_Loop, 'r'); % Step response of the discrete − time system (Closed loop)
hold on;
step(feedback(comp_GH_W, 1), 'm');
% Step response of the w − domain system after compensator
legend('Z − Domain Discrete − Time System', 'w − Domain Continuous
        − Time System (Bilinear Transformation) after compensator');
title('New Comparison of Step Responses (after compensator)');
grid on;
%% Bode Plot of new W − domain (after compensator)
% Bode Plot of the w − domain system (open − loop)
figure('name', "Bode Plot of the w − domain system (open − loop)");
bode(comp_GH_W); % Open − loop Bode plot in the w − domain
title('Bode Plot of the w − Domain System (Bilinear Transformation) after compensator');
grid on;
%% Comparision between step response of the three steps
figure('name', "Comparision between step response of the three steps");
step(feedback(GH_w, 1), 'r'); % Step response of the discrete − time system (Closed loop)
hold on;
step(feedback(new_GH_w, 1), 'm');
% Step response of the w − domain system after adding k
step(feedback(comp_GH_W, 1), 'b');
% Step response of the w − domain system after compensator
legend('Z − Domain Discrete − Time System step(1)', 'w − Domain Continuous
        − Time System after adding k step(2)', 'w − Domain Continuous
        − Time System after compensator step(3)');
```

```
title('Comparision between step response of the three steps');
grid on;
%% user defined Functions
function plot_bode_w_domain(new_GH_w)
    % Function to plot the Bode plot of the w - domain system (open - loop)
    %
    % Inputs:
    %   - new_GH_w: Transfer function of the open - loop w - domain system
    %
    % Outputs:
    %   - None (Generates a Bode plot)

    % Create a new figure for the Bode plot
    figure('Name', "New Bode Plot of the w - domain system (open - loop)");

    % Generate the Bode plot
    bode(new_GH_w);

    % Add title and grid
    title('New Bode Plot of the w - Domain System (Bilinear Transformation)');
    grid on;
end
```

Please See the README File to Run this Code

The End Thank You