

## پروژه نهایی سیستم های کنترل خطی

نیمسال دوم ۱۴۰۲

نام دانشجو: درسا امیری ابیانه

شماره دانشجویی: ۴۰۱۴۱۱۲۲۸

نام استاد: دکتر سهیل گنجه فر

## مقدمه

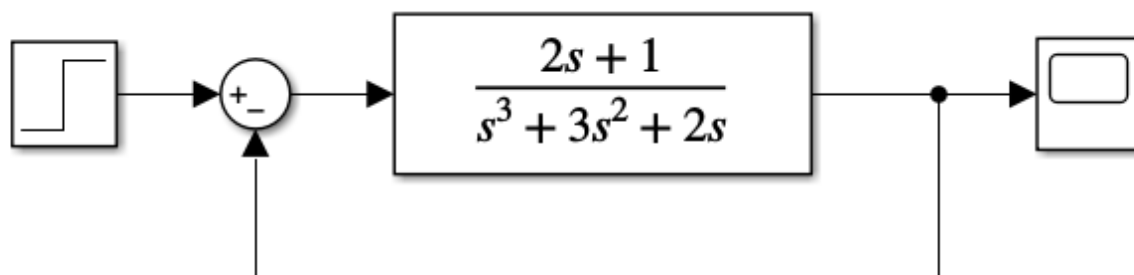
پروژه‌ی حاضر به بررسی و طراحی کنترل‌کننده در سیستم‌های کنترل خطی می‌پردازد. هدف این پروژه بهبود پاسخ زمانی سیستمی مشخص با استفاده از روش‌های مختلف طراحی کنترل‌کننده است. در ابتدا سیستم بدون کنترل‌کننده شبیه‌سازی شده و پاسخ آن به ورودی پله واحد بررسی می‌شود. سپس با طراحی و اعمال کنترل‌کننده‌های مناسب، سعی در بهبود مشخصات دینامیکی سیستم داریم.

## فهرست مطالب

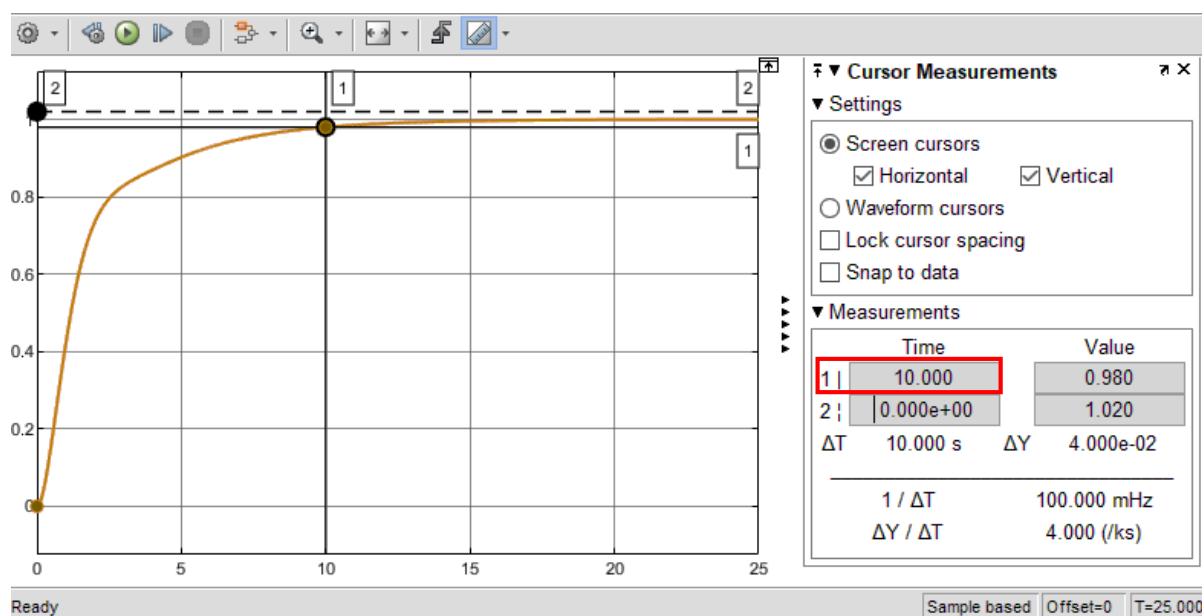
۴	شبیه سازی سیستم پیش از طراحی کنترل کننده .....
۵	طراحی کنترل کننده .....
۵	الف) بالازدگی حداکثر ۳۰٪ و زمان نشست ۳ ثانیه .....
۷	با گین $K \simeq 1.2$ .....
۷	پیش فاز یا Phase Lead: .....
۱۰	کنترل کننده تناسبی (P) با روش زیگلر نیکولز .....
۱۰	کنترل کننده تناسبی- انتگرال گیر (PI) با روش زیگلر نیکولز: .....
۱۰	کنترل کننده تناسبی- مشتق گیر – انتگرال گیر (PID) با روش زیگلر نیکولز: .....
۱۲	ب) $Kv \geq 34$ : .....
۱۲	پس فاز یا Phase Lag برای: .....
۱۴	Root Locus سیستم .....
۱۴	الف) پیش از اعمال کنترل کننده .....
۱۶	ب) پس از اعمال کنترل کننده .....
۱۶	گین $K \simeq 1.2$ : .....
۱۷	پیش فاز .....
۲۱	پس فاز .....
۲۶	پاسخ سیستم به ورودی پله واحد .....
۳۶	پاسخ سیستم به ورودی شیب .....
۴۶	استفاده از sisotool برای طراحی کنترل کننده .....
۴۸	تحلیل کلی پاسخ ها .....
۵۲	کل کد مطلب جهت ران کردن در Matlab Live Editor .....

## شبیه سازی سیستم پیش از طراحی کنترل کننده

در این قسمت سیستم شبیه سازی شده و پاسخ آن به ورودی پله واحد پیش از طراحی کنترل کننده بررسی گشته است .



شکل ۱\_ شبیه سازی سیستم پیش از طراحی کنترل کننده در Simulink



شکل ۲\_ زمان نشست سیستم پیش از طراحی کنترل کننده

همان طور که مشاهده می شود زمان نشست سیستم (با شاخص ۲٪) پیش از طراحی کنترل کننده برابر ۱۰ ثانیه می باشد. همچنین پاسخ زمانی سیستم فاقد بالازدگی می باشد. زمان صعود سیستم تقریباً برابر ۵ ثانیه و زمان تاخیر آن تقریباً ۱,۱۷ ثانیه می باشد.

همانطور که مشاهده می شود مقادیر بدست آمده برای زمان نشست و صعود زیاد می باشند پس با طراحی کنترل کننده های متفاوت تلاش شده تا پاسخ سیستم بهبود داده شود.

## طراحی کنترل کننده

الف) بالازدگی حداکثر ۳۰٪ و زمان نشست ۳ ثانیه  
وقتی گفته می شود بالازدگی حداکثر ۳۰٪ برای اینکه پاسخ سریعتر باشد، بالازدگی همان ۳۰٪ در نظر گرفته می شود پس باتوجه به محاسبات زیر خواهیم داشت :

Equation 1\_ بالازدگی

$$\text{overshoot} = e^{\frac{-\pi}{\tan \beta}} \times 100\%$$

Equation 2\_ زاویه با محور حقیقی

$$\tan \beta = \frac{\sqrt{1 - \xi^2}}{\xi}$$

$$\Rightarrow 30\% = e^{\frac{-\pi}{\tan \beta}} \times 100\%$$

$$\Rightarrow \ln 0.3 = -\frac{\pi}{\tan \beta}$$

$$\Rightarrow \beta \simeq 69.03^\circ \simeq 1.20 \text{ rad}$$

Equation 3\_ نسبت میرایی (Damping ratio)

$$\beta = \cos^{-1} \xi$$

$$\Rightarrow \xi = \cos \beta$$

$$\Rightarrow \xi \simeq 0.36$$

Equation 4\_ زمان نشست (Settling Time)

$$t_s = \frac{4}{\xi \omega_n} \Big|_{2\%} = 3$$

$$\Rightarrow \xi \omega_n = \frac{4}{3} \simeq 1.33$$

Equation 5\_ زمان صعود (Raise Time)

$$t_r = \frac{\pi \cdot \beta}{\omega_d}$$

Equation 6\_ زمان پیک (Pick Time)

$$t_p = \frac{\pi}{\omega_d}$$

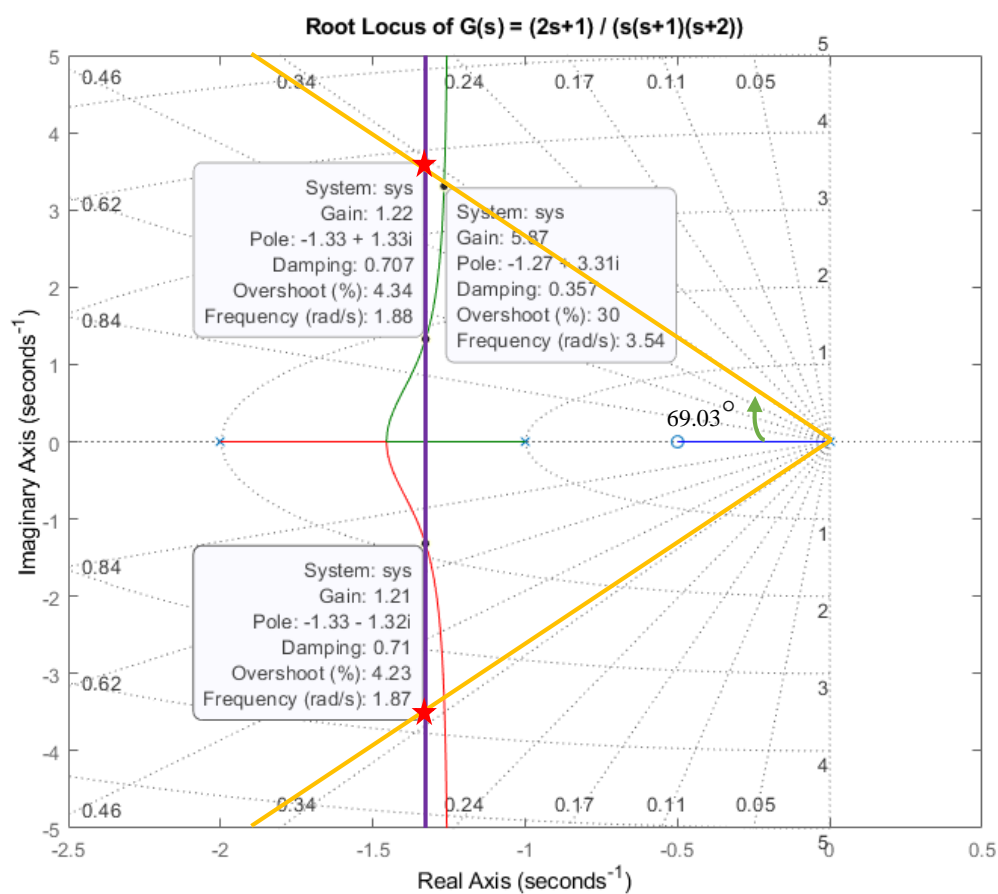
Equation 7\_ فرکانس طبیعی میرا (Damped Natural Frequency)

$$\omega_d = \omega_n \sqrt{1 - \xi^2}$$

$$\Rightarrow \omega_d = \frac{4}{3\xi} \sqrt{1 - \xi^2} = \frac{4}{3} \tan \beta = 3.48$$

$$\Rightarrow t_r = 1.09 \text{ sec}$$

$$\Rightarrow t_p = 0.90 \text{ sec}$$



شکل ۳ - Root Locus سیستم با اعمال شروط بالازدگی ۳۰٪ و زمان نشست ۳ ثانیه

همانطور که مشاهده میشود مکان هندسی ریشه ها در سمت راست نقاط برخورد واقع شده اند پس برای اینکه ★ ها روی نمودار قرار گیرند، باید نمودار به سمت چپ منتقل شود که این امر با یک بهره  $K$  به تنهایی ممکن نیست.

با گین  $K \approx 1.2$

دقت شود از آنجایی که بالازدگی حداکثر ۳۰ درصد است باتوجه اطلاعات نقاط در شکل ۳ پیداست به ازای  $K \approx 1.2$  ،  $\text{Overshoot} = 4.2 \sim 4.3\%$  و زمان نشست همان ۳ ثانیه بدست می‌آید که مطلوب است. پس باید انتظار داشت بدلیل کاهش مقدار بلازدگی، پاسخ سیستم کندتر از حالاتی که کنترل کننده را به ازای حداکثر مقدار overshoot طراحی میکنیم باشد.

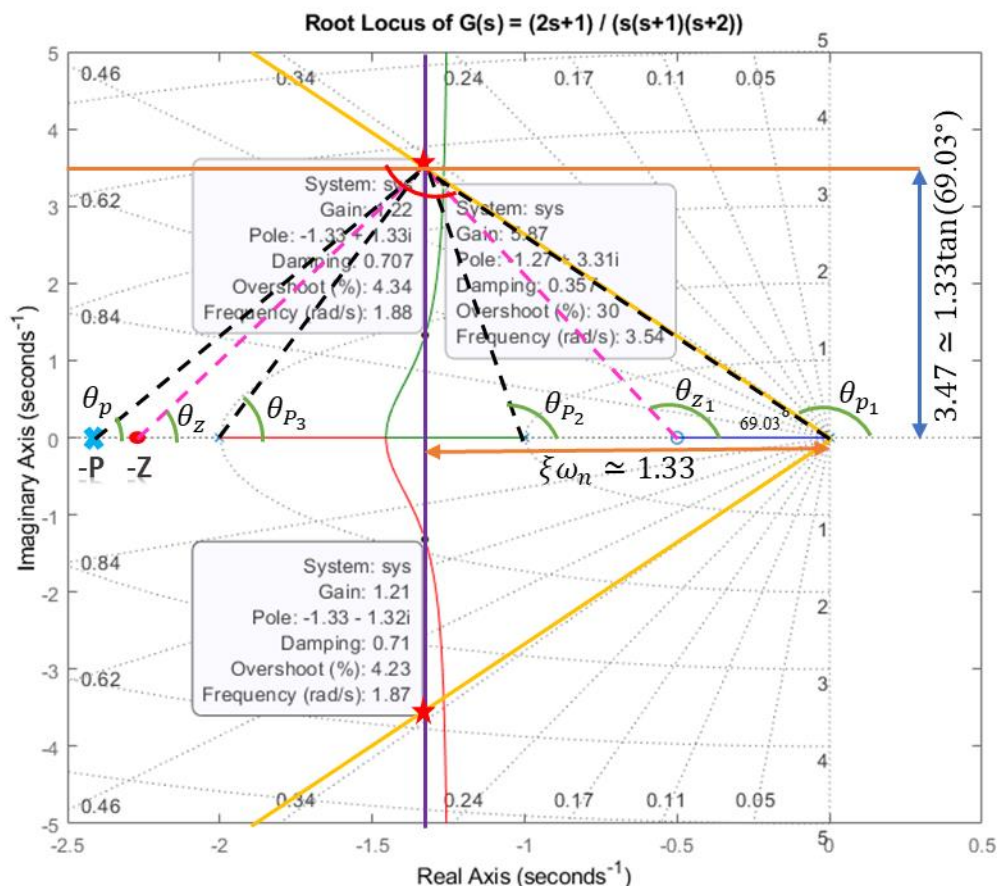
پیش فاز یا Phase Lead:

از آنجایی که در طراحی پیش فاز قطب ها و صفرهای اعمالی به سیستم هر جایی می‌توانند باشد پس در حالات مختلف بای در نظر گرفتن این صفر و قطب ها این کنترل کننده را طراحی کرده و پاسخ سیستم را با در نظر گرفتن هر کدام از آنها بررسی میکنیم:

Equation <sup>۸</sup> \_ شرط زوایا در طراحی پیش فاز

$$\begin{aligned} \theta_z - \theta_p + \theta_{z_1} - \theta_{p_1} - \theta_{p_2} - \theta_{p_3} &= 180 \\ \Rightarrow \varphi = \theta_z - \theta_p &= 180 + \theta_{p_1} + \theta_{p_2} + \theta_{p_3} - \theta_{z_1} \\ \Rightarrow \varphi &= 180 + 110.97 + 95.44 + 79.07 - 103.46 \quad \Rightarrow \varphi = 2.02^\circ \end{aligned}$$

پس زاویه ای که پیش فاز به سیستم اضافه خواهد کرد برابر  $\varphi = 2.02^\circ$  خواهد بود.



شکل ۴ \_ زوایای مشخص شده جهت محاسبه زاویه اعمال شونده به سیستم از طرف پیش فاز در حالات دیگر

(a) صفر اعمال شونده به سیستم روی قطب ۲- قرار گیرد (کنترل کننده پیش فاز ۱):

پس در این حالت  $-Z = -2 \leftarrow Z = 2$  است. حال می توان مقادیر  $K, P$  را نیز محاسبه کرد:

$$\varphi = \theta_z - \theta_p = 2.02^\circ, \theta_z = \theta_{p_s} = 79.07^\circ \Rightarrow \theta_p = 77.05^\circ$$

$$\frac{3.47}{P - 1.33} = \tan 77.05 \Rightarrow P = 0.80$$

$$\Rightarrow G_c(s) = k_c \frac{s + 2}{s + 2.13}$$

$$k_c \left| \frac{2s + 1}{s(s + 1)(s + 2.13)} \right|_{s=-1.33 \pm 3.47j} = 1 \Rightarrow k_c \simeq 6.46$$

(b) صفر اعمال شونده به سیستم روی نقطه ۱,۵- قرار گیرد (کنترل کننده پیش فاز ۲):

پس در این حالت  $-Z = -1.5 \leftarrow Z = 1.5$  است. حال می توان مقادیر  $K, P$  را نیز محاسبه کرد:

$$\theta_z - \theta_p = 2.02^\circ \Rightarrow \theta_p = 85.18$$

$$\frac{3.47}{P - 1.33} = \tan 85.18 \Rightarrow P = 1.62$$

$$\Rightarrow G_c(s) = k_c \frac{s + 1.5}{s + 1.62}$$

$$k_c \left| \frac{(s + 1.5)(2s + 1)}{s(s + 1)(s + 2)(s + 1.62)} \right|_{s=-1.33 \pm 3.47j} = 1 \Rightarrow k_c \simeq 6.43$$

(c) صفر اعمال شونده به سیستم روی نقطه ۲,۵- قرار گیرد (کنترل کننده پیش فاز ۳):

پس در این حالت  $-Z = -2.5 \leftarrow Z = 2.5$  است. حال میتوان مقادیر  $K, P$  را نیز محاسبه کرد:

$$\theta_2 = \arctan\left(\frac{3.47}{2.5 - 1.33}\right) = 71.37^\circ$$

$$\theta_z - \theta_p = 2.02, \theta_z = 71.37^\circ \Rightarrow \theta_p = 69.35$$

$$\frac{3.47}{P - 1.33} = \tan 69.35 \Rightarrow P = 2.64$$

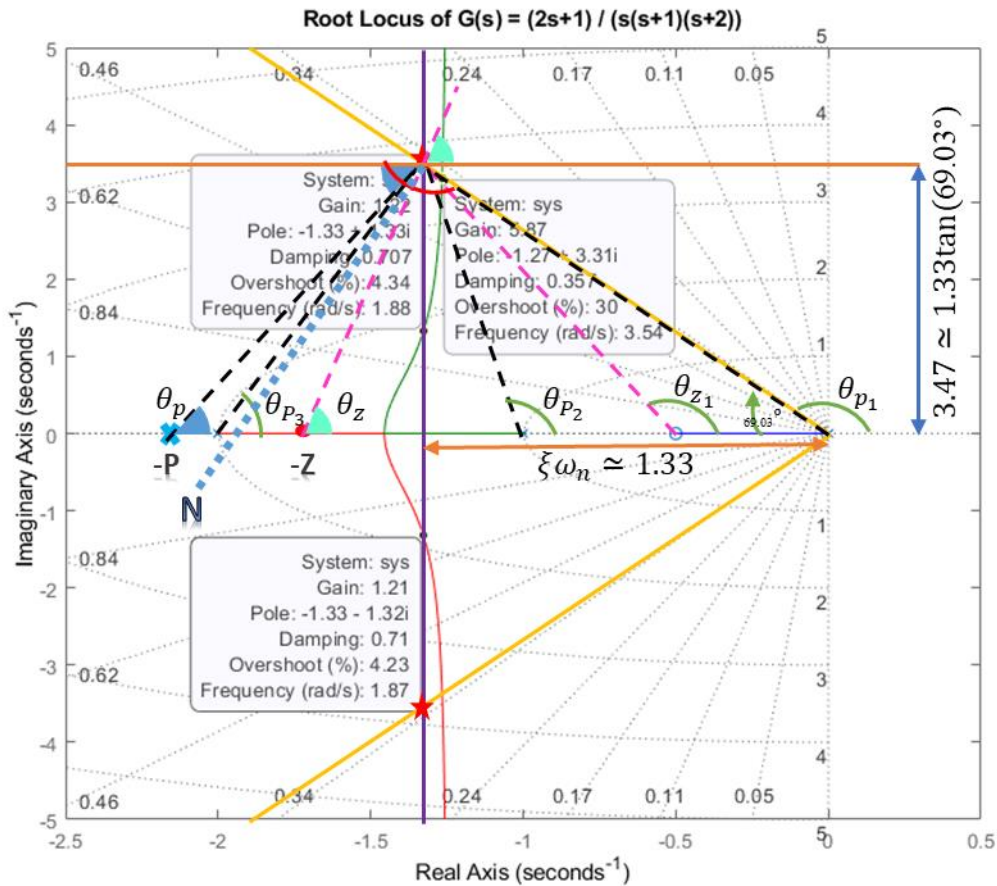
$$G_c(s) = k_c \frac{s + 2.5}{s + 2.64}$$

$$K_c \left| \frac{(s + 2.5)(2s + 1)}{s(s + 1)(s + 2)(s + 2.64)} \right|_{s=-1.33 \pm 3.47j} = 1 \Rightarrow k_c \simeq 6.50$$



(d) روش هندسی (کنترل کننده پیش فاز با روش هندسی):

در این روش باید پس از رسم نیمساز و اعمال زاویه  $\frac{\varphi}{2}$  در جهت ساعت گرد و پادساعتگرد از نیمساز، مقادیر  $Z$ ,  $P$  را از روی زوایای محاسبه شده برای صفر ها و قطب ها و  $K$  را همانند موارد فوق از شرط اندازه محاسبه کرد:



شکل ۵\_ زوایای مشخص شده جهت محاسبه زاویه اعمال شونده به سیستم از طرف پیش فاز در روش هندسی

$$\frac{180 - 69.03}{2} = 55.485$$

$$\frac{\varphi}{2} = 1.01 \Rightarrow \begin{cases} \theta_p = 55.485 - 1.01 = 54.475 \\ \theta_z = 55.485 + 1.01 = 56.495 \end{cases}$$

$$\tan 54.475^\circ = \frac{3.47}{P - 1.33} \Rightarrow P \simeq 3.81$$

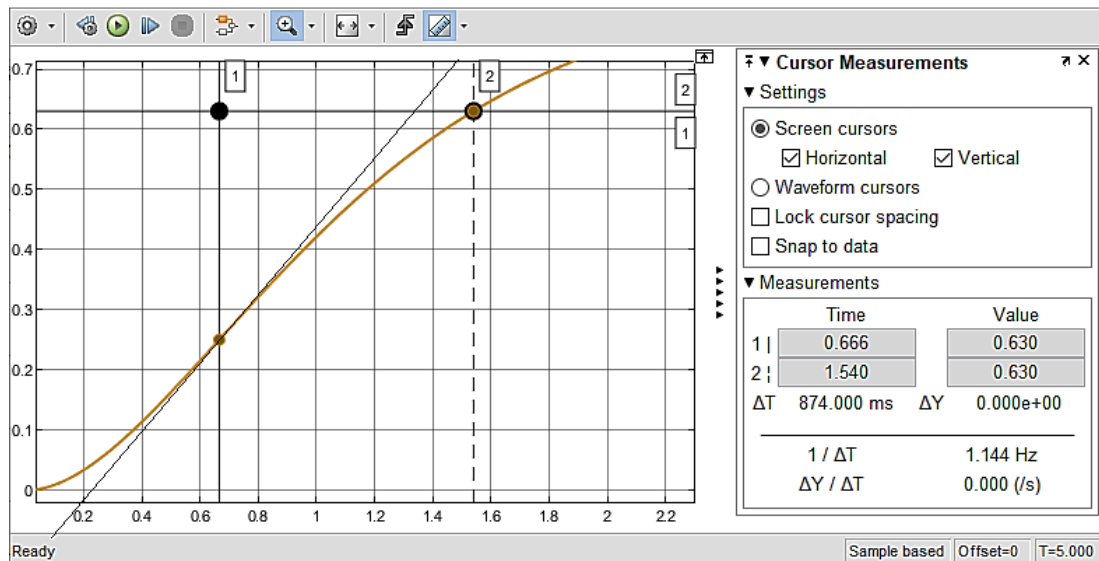
$$\tan 56.495^\circ = \frac{3.47}{z - 1.33} \Rightarrow z \simeq 3.63$$

$$\Rightarrow G_c(s) = k_c \frac{s + 3.63}{s + 3.81}$$

$$K_c \left| \frac{(s + 3.63)(2s + 1)}{s(s + 1)(s + 2)(s + 3.81)} \right|_{s = -1.33 \pm 3.47j} = 1 \Rightarrow k_c \simeq 6.57$$

### کنترل کننده تناسبی (P) با روش زیگلر نیکولز

در این روش سعی می‌شود تا با تعریف تابع انتقالی از روی پاسخ پله سیستم، سیستم اصلی را با سیستم مرتبه اولی سپس با توجه به پارامترهای محاسبه شده، کنترل کننده ای طراحی کرد. از شکل ۱ می‌توان برداشت کرد پاسخ نهایی سیستم به ورودی پله واحد در حالت گذرا برابر ۱ خواهد بود حال برای محاسبه باقی پارامترهای مورد نیاز داریم:



شکل ۶\_ بررسی قسمتی از پاسخ پله سیستم اولیه جهت تقریب زدن آن با تابع انتقال مرتبه اول

$$\tau_d \simeq 0.2, \quad T + \tau_d \simeq 1.54 \Rightarrow T = 1.34 \quad k' = 1 = k$$

$$\Rightarrow G(s) = \frac{e^{-0.2s}}{1 + 1.34s} \quad G_c(s) = K_p \left( 1 + T_d s + \frac{1}{T_i s} \right)$$

$$(P): \begin{cases} K_{p1} = \frac{T}{\tau_d} = \frac{1.34}{0.2} \\ T_i = \infty \\ T_d = 0 \end{cases} \Rightarrow G_c(s) = 6.7$$

کنترل کننده تناسبی- انتگرال گیر (PI) با روش زیگلر نیکولز:

$$(PI): \begin{cases} K_{p2} = 0.9 K_{p1} = 6.03 \\ T_i = \frac{\tau_d}{0.3} \approx 0.67 \\ T_d = 0 \end{cases} \Rightarrow G_c(s) = 6.03 \left( 1 + \frac{1}{0.67s} \right)$$

کنترل کننده تناسبی- مشتق گیر - انتگرال گیر (PID) با روش زیگلر نیکولز:

$$(PID): \begin{cases} K_{p3} = 1.2 k_{p1} = 8.04 \\ T_i = 2\tau_d = 0.4 \\ T_d = 0.5\tau_d = 0.1 \end{cases} \Rightarrow G_c(s) = 8.04 \left( 1 + 0.1s + \frac{1}{0.4s} \right)$$

```

% Define the transfer function  $G(s) = 1 / (1 + 1.34s)$ 
numerator = 1; % Numerator of the transfer function
denominator = [1 1.34]; % Denominator of the transfer function
G = tf(numerator, denominator); % Create transfer function G(s)

% Define the delay
delay = 0.2; % Time delay in seconds

% Pade approximation for the delay
[num_delay, den_delay] = pade(delay, 1); % 1st-order Pade approximation
Delay_approx = tf(num_delay, den_delay); % Create transfer function for
delay approximation

% Combine transfer function with delay approximation
G_with_delay = series(G, Delay_approx); % Series combination of G(s) and
delay

% Define controllers
% P Controller
Kp1 = 1.34 / 0.2; % Proportional gain for P controller
Gc1 = Kp1; % Transfer function of P controller

% PI Controller
Kp2 = 0.9 * Kp1; % Proportional gain for PI controller
Ti2 = 0.2 / 0.3; % Integral time constant for PI controller
Gc2 = Kp2 * (1 + tf(1, [Ti2 0])); % Transfer function of PI controller

% PID Controller
Kp3 = 1.2 * Kp1; % Proportional gain for PID controller
Ti3 = 2 * 0.2; % Integral time constant for PID controller
Td3 = 0.5 * 0.2; % Derivative time constant for PID controller
Gc3 = Kp3 * (1 + tf([Td3 0], 1) + tf(1, [Ti3 0])); % Transfer function of
PID controller

```

$$(ب) \quad K_v \geq 34 :$$

همواره سعی می‌شود که کنترل کننده ای طراحی شود تا پاسخ سیستم به ازای ماکزیمم خطای ممکن بهبود یابد، از آنجایی که در رابطه خطای حالت دائم با ورودی شیب داریم  $e_{ss} = \frac{1}{k_v}$  پس  $K_{v, new} = 34$  به ازای کمترین مقدار خود، بیشترین خطای ممکن را بوجود می‌آورد و از آنجایی که تلاش بر تغییر خطای حالت دائم می‌باشد پس باید یک Phase Lag طراحی کنیم پس خواهیم داشت:

$$\frac{k_{v, New}}{k_{v, old}} = \frac{z}{P} \quad k_{v, New} = 34$$

$$k_v \geq 34 \Rightarrow \frac{1}{k_v} \leq \frac{1}{34} \simeq 0.03$$

$$\left. \begin{array}{l} z = \frac{\alpha}{10} \\ \alpha = \xi \omega_n \simeq 1.333 \end{array} \right\} \Rightarrow z = 0.1333$$

$$k_{v, old} = \lim_{s \rightarrow 0} sG(s)G_c(s)H(s)$$

پس فاز یا Phase Lag برای:

$$K \simeq 1.2 \text{ کنترل کننده}$$

$$\begin{aligned} k_{v, old} &= \lim_{s \rightarrow 0} s \times \frac{1.2(2s+1)}{s(s+1)(s+2)} \simeq 0.6 \\ \Rightarrow \frac{34}{0.6} &= \frac{0.1333}{P} \rightarrow P \simeq 0.0024 \\ \Rightarrow \hat{G}_c(s) &= \frac{s+0.1333}{s+0.0024} \end{aligned}$$

کنترل کننده پیش فاز ۱

$$k_{v, old} = \lim_{s \rightarrow 0} s \times \frac{6.46(s+2)(2s+1)}{s(s+1)(s+2)(s+2.13)} \simeq 3.03$$

$$\Rightarrow \frac{34}{3.03} = \frac{0.1333}{P} \Rightarrow P \simeq 0.0119$$

$$\Rightarrow \hat{G}_c(s) = \frac{s+0.1333}{s+0.0119}$$

کنترل کننده پیش فاز ۲

$$k_{v, \text{old}} = \lim_{s \rightarrow 0} s \times \frac{6.43(s + 1.5)(2s + 1)}{s(s + 1)(s + 2)(s + 1.62)} \simeq 2.98$$

$$\Rightarrow \frac{34}{2.98} = \frac{0.1333}{P} \rightarrow P \simeq 0.0117$$

$$\Rightarrow \hat{G}_c(s) = \frac{S + 0.1333}{S + 0.0117}$$

کنترل کننده پیش فاز ۳

$$k_{v, \text{old}} = \lim_{s \rightarrow 0} s \times \frac{6.5(s + 2.5)(2s + 1)}{s(s + 1)(s + 2)(s + 2.64)} \simeq 3.08$$

$$\Rightarrow \frac{34}{3.08} = \frac{0.1333}{P} \Rightarrow P \simeq 0.0121$$

$$\Rightarrow \hat{G}_c(s) = \frac{S + 0.1333}{S + 0.0121}$$

کنترل کننده پیش فاز با روش هندسی

$$k_{v, \text{old}} = \lim_{s \rightarrow 0} s \times \frac{6.57(s + 3.63)(2s + 1)}{s(s + 1)(s + 2)(s + 3.81)} \simeq 3.13$$

$$\Rightarrow \frac{34}{3.13} = \frac{0.1333}{P} \Rightarrow P \simeq 0.0123$$

$$\hat{G}_c(s) = \frac{S + 0.1333}{S + 0.0123}$$

## Root Locus سیستم

الف) پیش از اعمال کنترل کننده

\*\*\*Matlab Code\*\*\*

```
% Define the numerator and denominator of the transfer function
num = [2 1]; % Numerator coefficients of '2s+1'
den = conv([1 0], conv([1 1], [1 2])); % Denominator coefficients of
's(s+1)(s+2)'
```

```
% Create the transfer function model
sys = tf(num, den)
```

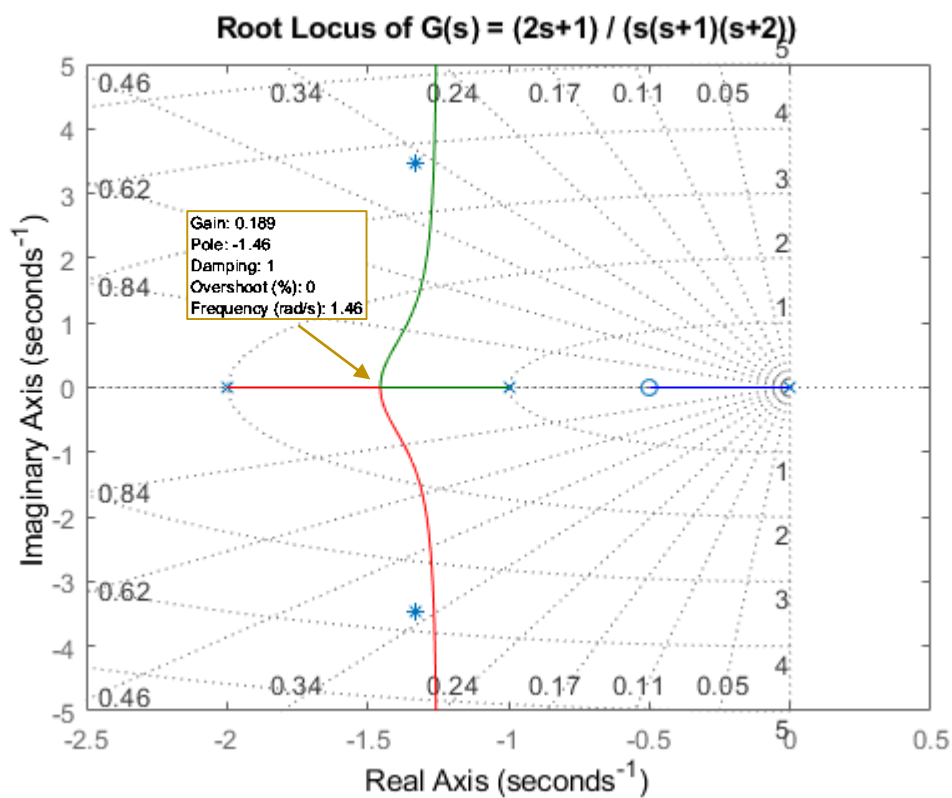
```
sys =

      2 s + 1
-----
s^3 + 3 s^2 + 2 s
```

Continuous-time transfer function.

Model Properties

```
% Generate and plot the root locus
figure;
rlocus(sys);
hold on
points = [-1.33+3.47j, -1.33-3.47j]; % Points to mark on the root locus plot
plot(points, '*'); % Plotting points on the root locus plot
title('Root Locus of G(s) = (2s+1) / (s(s+1)(s+2))');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```

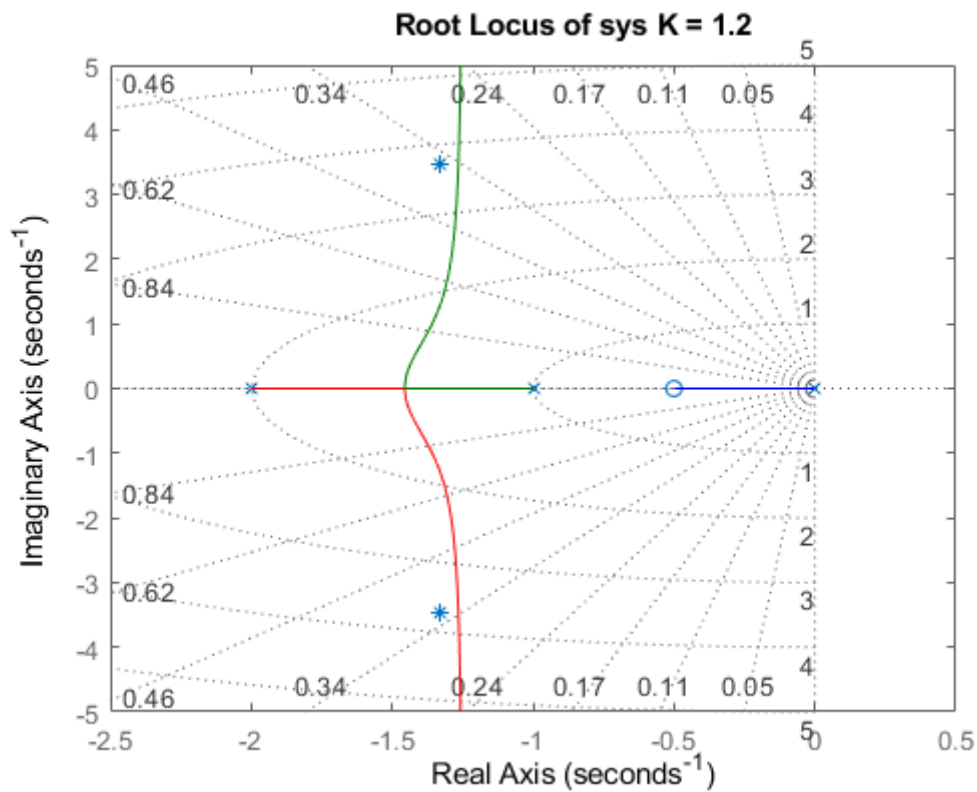


شکل ۷- Root Locus سیستم حلقه بسته با فیدبک منفی واحد پیش از طراحی کنترل کننده

باتوجه root locus در این حالت مشاهده می شود که سیستم به ازای تمامی مقادیر حقیقی  $K$  همواره پایدار بوده

ب) پس از اعمال کنترل کننده  
گین  $K \simeq 1.2$ :

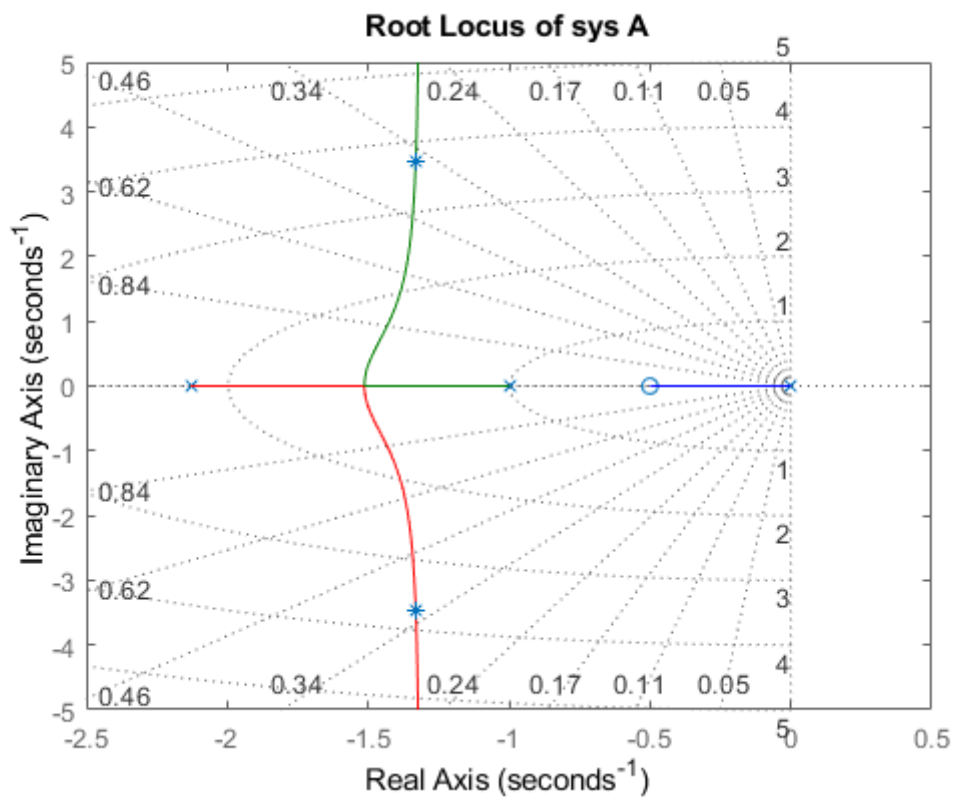
```
% Root locus plot for sys_0 (Controller with K = 1.2)
figure;
rlocus(sys_0);
hold on
points = [-1.33+3.47j, -1.33-3.47j]; % Points to mark on the root locus
plot
plot(points, '*');
title('Root Locus of sys K = 1.2');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```



شکل 8 Root Locus سیستم با گین 1.2

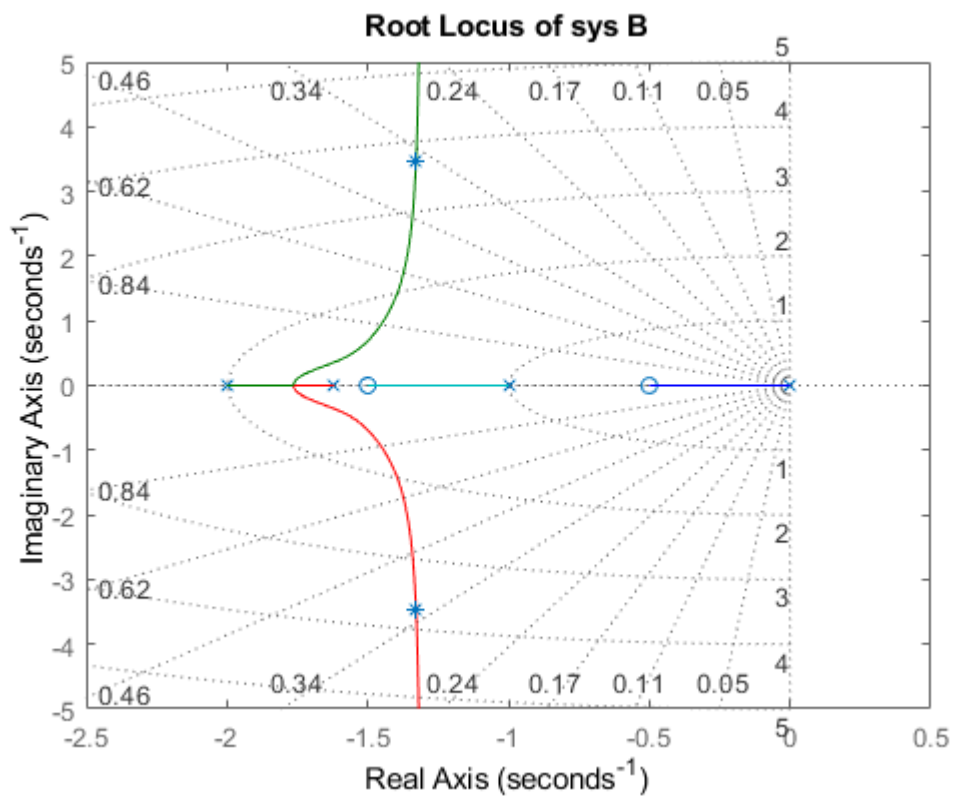


```
% Root locus plots for systems with phase-lead compensators
% Root locus plot for sys_A
figure;
rlocus(sys_A);
hold on
plot(points, '*');
title('Root Locus of sys A');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
```



شکل ۹ \_ Root Locus سیستم پس از اعمال کنترل کننده پیش فاز ۱

```
% Root locus plot for sys_B
figure;
rlocus(sys_B);
hold on
plot(points, '*');
title('Root Locus of sys B');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```

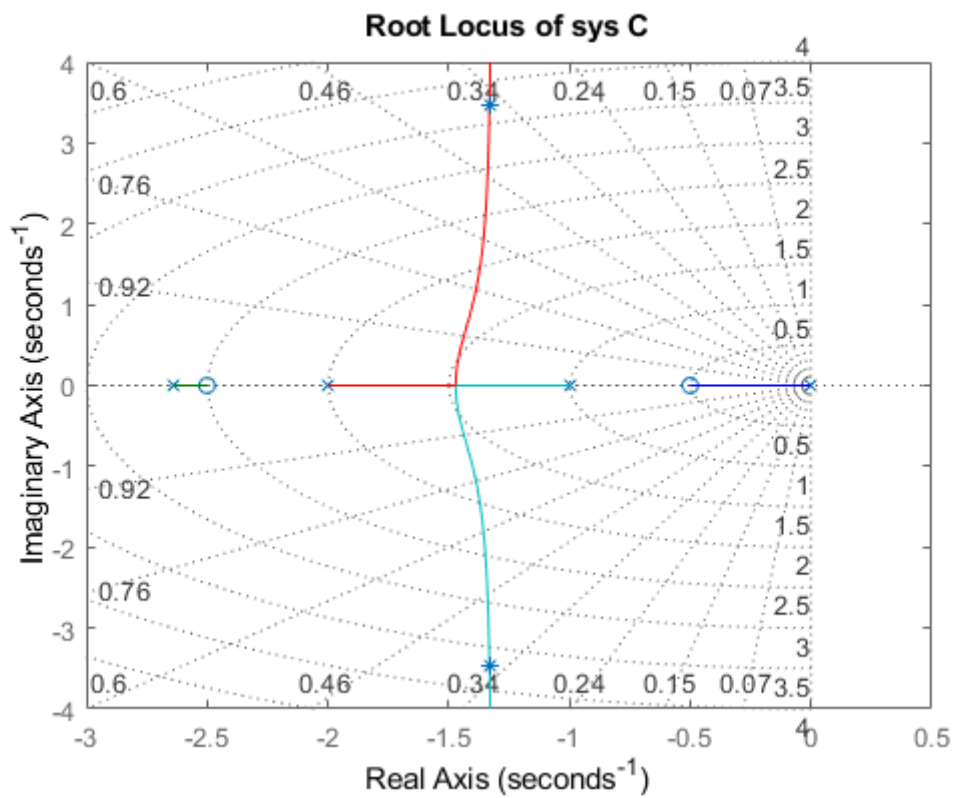


شکل 10 Root Locus سیستم پس از اعمال کنترل کننده پیش فاز ۲

```

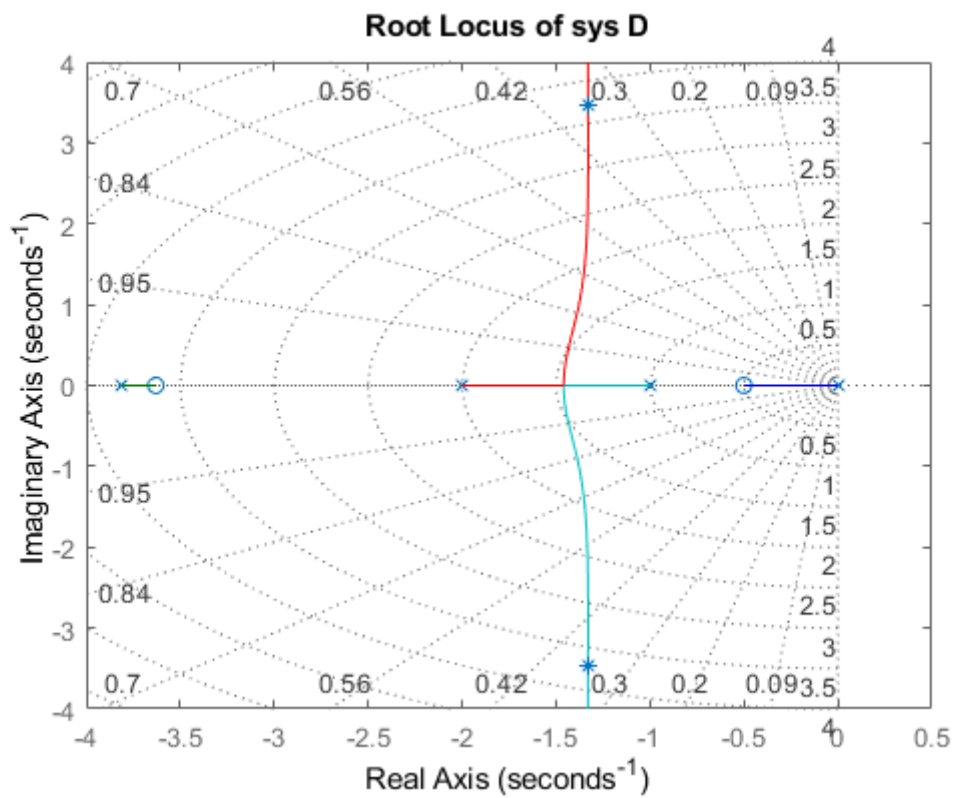
% Root locus plot for sys_C
figure;
rlocus(sys_C);
hold on
plot(points, '*');
title('Root Locus of sys C');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

```



شکل 11 \_ Root Locus سیستم پس از اعمال کنترل کننده پیش فاز ۳

```
% Root locus plot for sys_D
figure;
rlocus(sys_D);
hold on
plot(points, '*');
title('Root Locus of sys D');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```



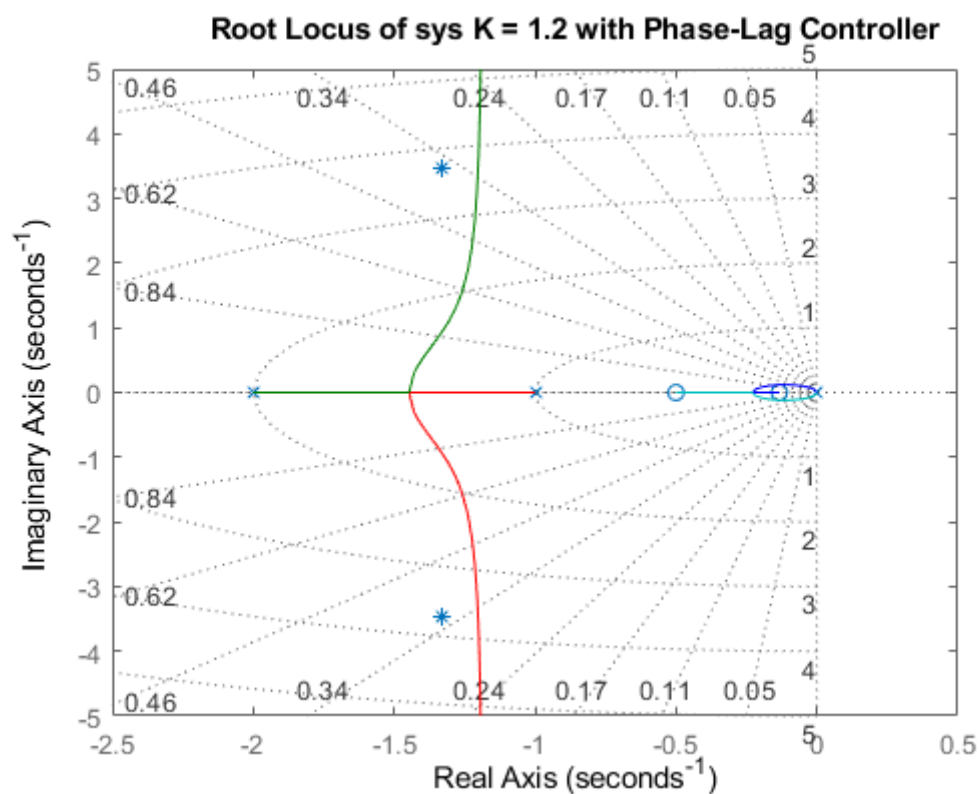
شکل 12 \_ Root Locus سیستم پس از اعمال کنترل کننده پیش فاز با روش هندسی

پس فاز

```
% Define systems with added phase-lag controllers
sys_00 = series(tf([1 0.1333], [1 0.0041]), sys_0);
sys_AA = series(tf([1 0.1333], [1 0.0119]), sys_A);
sys_BB = series(tf([1 0.1333], [1 0.0117]), sys_B);
sys_CC = series(tf([1 0.1333], [1 0.0121]), sys_C);
sys_DD = series(tf([1 0.1333], [1 0.0123]), sys_D);
```

کنترل کننده  $K \simeq 1.2$

```
% Root locus plot for sys_00
figure;
rlocus(sys_00);
hold on
plot(points, '*');
title('Root Locus of sys K = 1.2 with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```

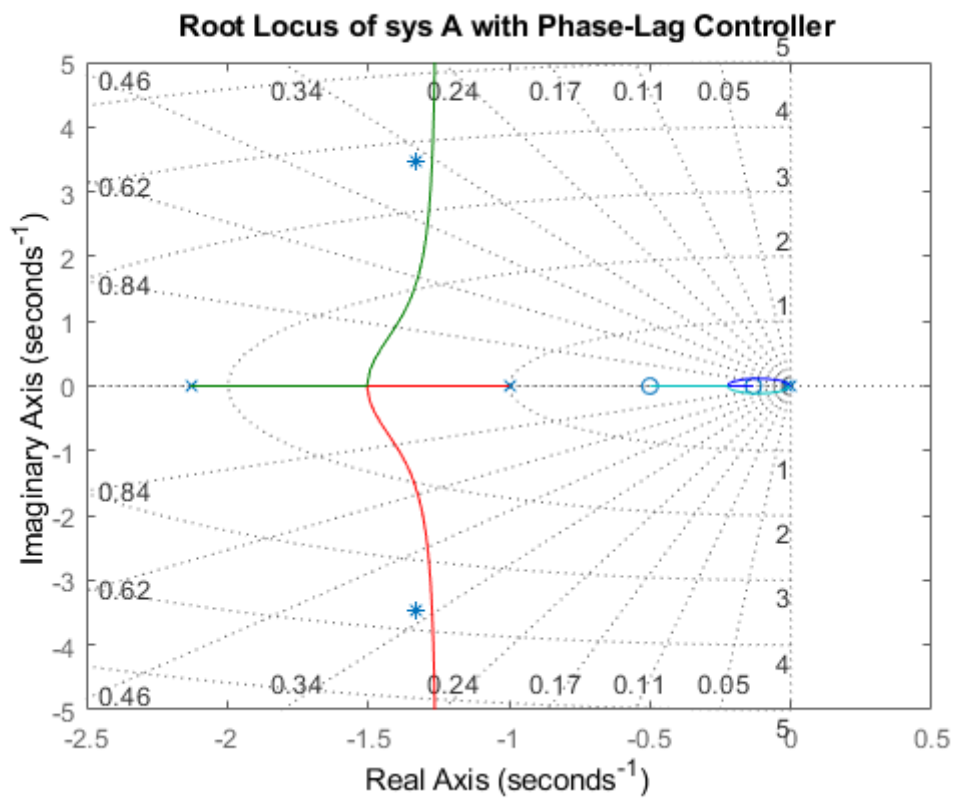


شکل 13 \_ Root Locus سیستم پس از اعمال کنترل کننده پس فاز ۱ به پیش فاز ۱

```

% Root locus plot for sys_AA
figure;
rlocus(sys_AA);
hold on
plot(points, '*');
title('Root Locus of sys A with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

```

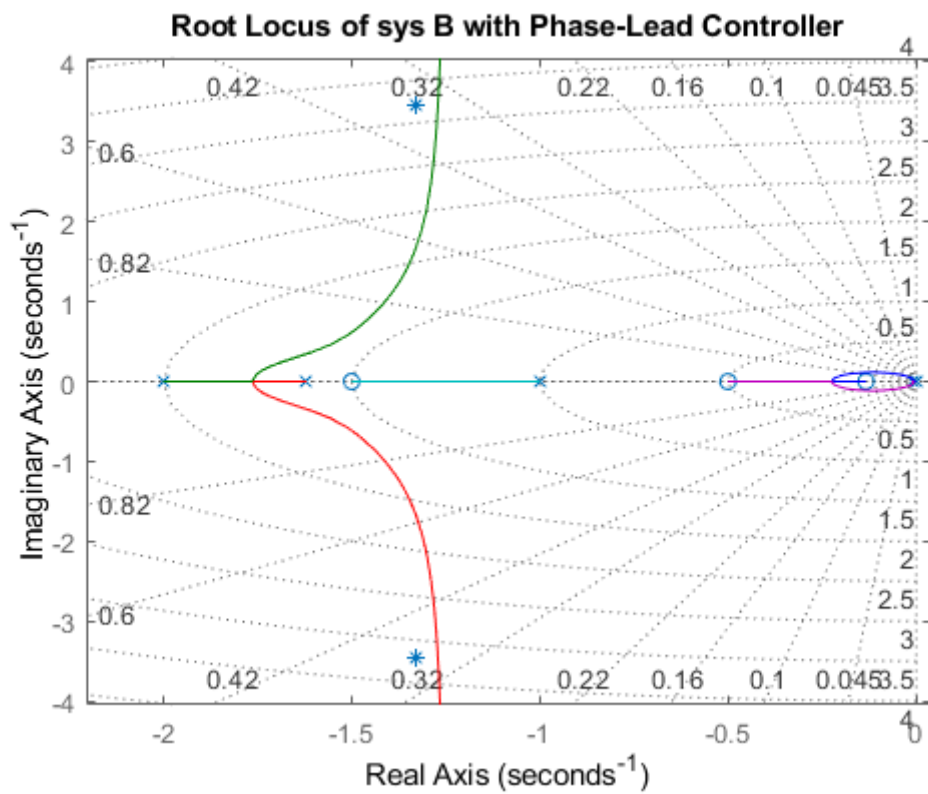


شکل 14 \_ Root Locus سیستم پس از اعمال کنترل کننده پس فاز ۲ به پیش فاز ۲

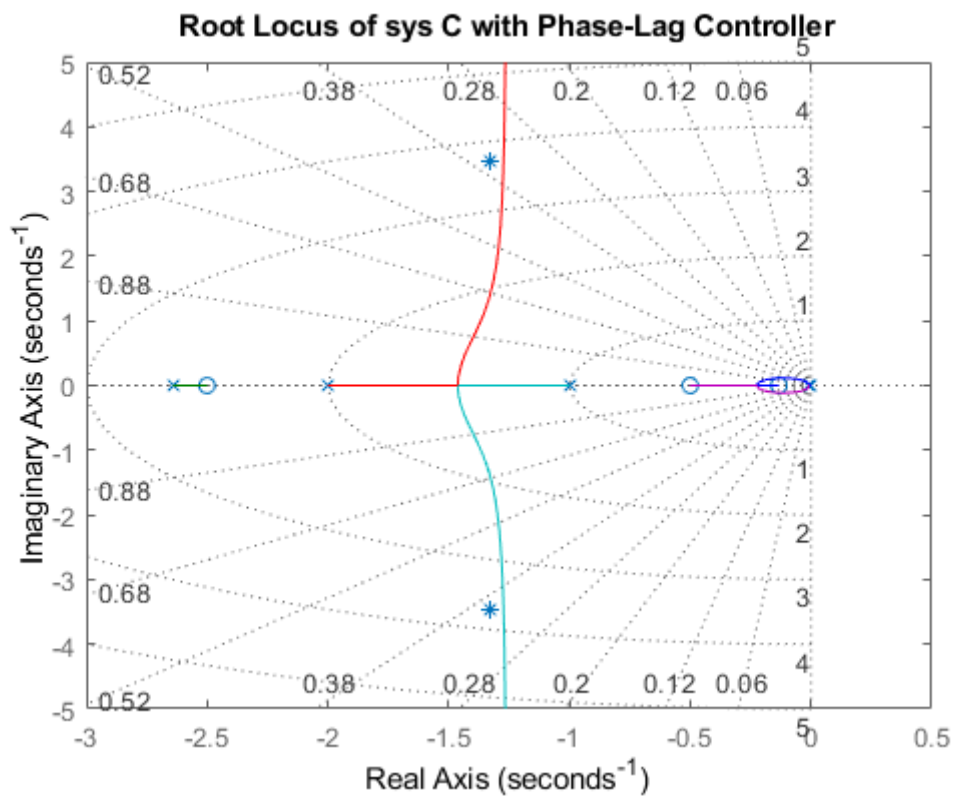
```

% Root locus plot for sys_BB
figure;
rlocus(sys_BB);
hold on
plot(points, '*');
title('Root Locus of sys B with Phase-Lead Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

```



```
% Root locus plot for sys_CC
figure;
rlocus(sys_CC);
hold on
plot(points, '*');
title('Root Locus of sys C with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off
```



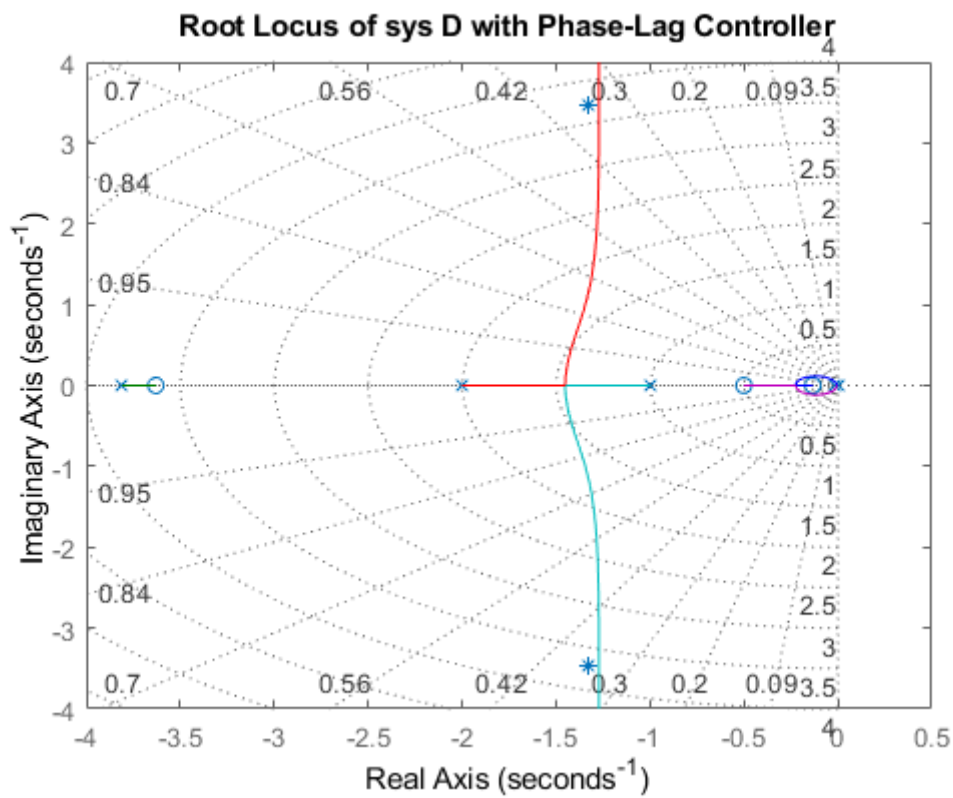
شکل ۵-۱ Root Locus سیستم پس از اعمال کنترل کننده پس فاز ۳ به پیش فاز ۳



```

% Root locus plot for sys_DD
figure;
rlocus(sys_DD);
hold on
plot(points, '*');
title('Root Locus of sys D with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

```



شکل 16 \_ Root Locus سیستم پس از اعمال کنترل کننده پس فاز به پیش فاز با روش هندسی

## پاسخ سیستم به ورودی پله واحد

کد متلب:

```
% Create closed-loop systems with unit feedback
cl_sys_0 = feedback(sys_0, 1); % Closed-loop system with initial gain
adjustment
cl_sys_A = feedback(sys_A, 1); % Closed-loop system with compensator A
cl_sys_B = feedback(sys_B, 1); % Closed-loop system with compensator B
cl_sys_C = feedback(sys_C, 1); % Closed-loop system with compensator C
cl_sys_D = feedback(sys_D, 1); % Closed-loop system with compensator D
cl_sys = feedback(sys, 1); % Closed-loop Main system

% Time vector for simulation
t = 0:0.01:10;

%% Step responses
[y_0, t_0] = step(cl_sys_0, t); % Step response of closed-loop system with
initial gain adjustment
[y_A, t_A] = step(cl_sys_A, t); % Step response of closed-loop system with
compensator A
[y_B, t_B] = step(cl_sys_B, t); % Step response of closed-loop system with
compensator B
[y_C, t_C] = step(cl_sys_C, t); % Step response of closed-loop system with
compensator C
[y_D, t_D] = step(cl_sys_D, t); % Step response of closed-loop system with
compensator D
[y, t] = step(cl_sys, t); % Step response of closed-loop Main system

% Display step information
info_0 = stepinfo(cl_sys_0, 'SettlingTimeThreshold', 0.02) % Step information
of closed-loop system with initial gain adjustment
info_0 =
    RiseTime: 3.6800
    TransientTime: 8.7277
    SettlingTime: 8.7277
    SettlingMin: 0.9003
    SettlingMax: 0.9998
    Overshoot: 0
    Undershoot: 0
    Peak: 0.9998
    PeakTime: 22.1264
```

```
info_A = stepinfo(cl_sys_A, 'SettlingTimeThreshold', 0.02) % Step information  
of closed-loop system with compensator A
```

```
info_A =  
    RiseTime: 0.4016  
    TransientTime: 3.6489  
    SettlingTime: 3.6489  
    SettlingMin: 0.8867  
    SettlingMax: 1.2334  
    Overshoot: 23.3444  
    Undershoot: 0  
    Peak: 1.2334  
    PeakTime: 0.8996
```

```
info_B = stepinfo(cl_sys_B, 'SettlingTimeThreshold', 0.02) % Step information  
of closed-loop system with compensator B
```

```
info_B =  
    RiseTime: 0.4030  
    TransientTime: 3.6993  
    SettlingTime: 3.6993  
    SettlingMin: 0.8855  
    SettlingMax: 1.2304  
    Overshoot: 23.0399  
    Undershoot: 0  
    Peak: 1.2304  
    PeakTime: 0.9014
```

```
info_C = stepinfo(cl_sys_C, 'SettlingTimeThreshold', 0.02) % Step information  
of closed-loop system with compensator C
```

```
info_C =  
    RiseTime: 0.4001  
    TransientTime: 3.6105  
    SettlingTime: 3.6105  
    SettlingMin: 0.8869  
    SettlingMax: 1.2364  
    Overshoot: 23.6383  
    Undershoot: 0  
    Peak: 1.2364  
    PeakTime: 0.8997
```

```
info_D = stepinfo(cl_sys_D, 'SettlingTimeThreshold', 0.02) % Step information  
of closed-loop system with compensator D
```

```
info_D =  
    RiseTime: 0.3978  
    TransientTime: 2.3797  
    SettlingTime: 2.3797  
    SettlingMin: 0.8867  
    SettlingMax: 1.2403  
    Overshoot: 24.0295
```

```
Undershoot: 0
Peak: 1.2403
PeakTime: 0.9004
```

```
info = stepinfo(cl_sys, 'SettlingTimeThreshold', 0.02) % Step information of
closed-loop Main system
```

```
info =
    RiseTime: 4.5641
  TransientTime: 9.9974
    SettlingTime: 9.9974
    SettlingMin: 0.9003
    SettlingMax: 0.9983
    Overshoot: 0
    Undershoot: 0
        Peak: 0.9983
    PeakTime: 17.7866
```

```
% Display step response plots
```

```
figure;
subplot(6,1,1);
plot(t_0, y_0);
grid on;
title('Step Response of Closed-Loop sys_0');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_0.RiseTime, info_0.SettlingTime, info_0.Overshoot));
```

```
subplot(6,1,2);
plot(t_A, y_A);
grid on;
title('Step Response of Closed-Loop sys_A');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_A.RiseTime, info_A.SettlingTime, info_A.Overshoot));
```

```
subplot(6,1,3);
plot(t_B, y_B);
grid on;
title('Step Response of Closed-Loop sys_B');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_B.RiseTime, info_B.SettlingTime, info_B.Overshoot));
```

```
subplot(6,1,4);
plot(t_C, y_C);
```

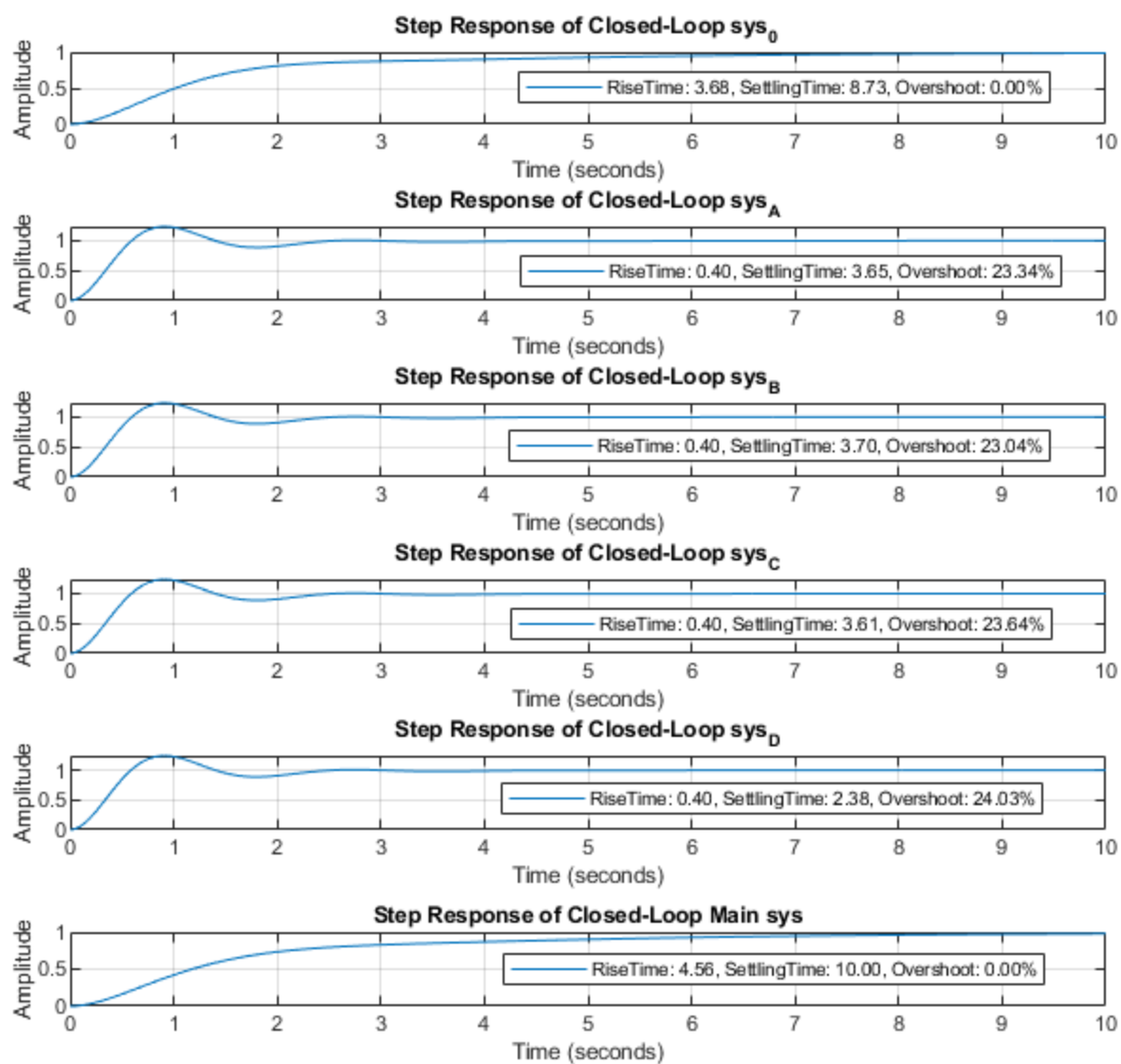
```

grid on;
title('Step Response of Closed-Loop sys_C');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_C.RiseTime, info_C.SettlingTime, info_C.Overshoot));

subplot(6,1,5);
plot(t_D, y_D);
grid on;
title('Step Response of Closed-Loop sys_D');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_D.RiseTime, info_D.SettlingTime, info_D.Overshoot));

subplot(6,1,6);
plot(t, y);
grid on;
title('Step Response of Closed-Loop Main sys');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info.RiseTime, info.SettlingTime, info.Overshoot));

```



شکل 17 \_ پاسخ پله سیستم های بدون کنترل کننده، با کنترل کننده پیشفاز و گین ۱/۲

```

% Create closed-loop systems with unit feedback
cl_sys_00 = feedback(sys_00, 1); % Closed-loop system with initial gain
adjustment
cl_sys_AA = feedback(sys_AA, 1); % Closed-loop system with compensator A
cl_sys_BB = feedback(sys_BB, 1); % Closed-loop system with compensator B
cl_sys_CC = feedback(sys_CC, 1); % Closed-loop system with compensator C
cl_sys_DD = feedback(sys_DD, 1); % Closed-loop system with compensator D

% Time vector for simulation
t = 0:0.01:10;

% Compute step responses
[y_00, t_00] = step(cl_sys_00, t);
[y_AA, t_AA] = step(cl_sys_AA, t);
[y_BB, t_BB] = step(cl_sys_BB, t);
[y_CC, t_CC] = step(cl_sys_CC, t);
[y_DD, t_DD] = step(cl_sys_DD, t);

% Compute step response characteristics
info_00 = stepinfo(y_00, t_00);
info_AA = stepinfo(y_AA, t_AA);
info_BB = stepinfo(y_BB, t_BB);
info_CC = stepinfo(y_CC, t_CC);
info_DD = stepinfo(y_DD, t_DD);

% Plot step responses
figure;
subplot(5,1,1);
plot(t_00, y_00);
grid on;
title('Step Response with Lag Controller sys_00');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_00.RiseTime, info_00.SettlingTime, info_00.Overshoot));

subplot(5,1,2);
plot(t_AA, y_AA);
grid on;
title('Step Response with Lag Controller sys_AA');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_AA.RiseTime, info_AA.SettlingTime, info_AA.Overshoot));

subplot(5,1,3);
plot(t_BB, y_BB);
grid on;

```

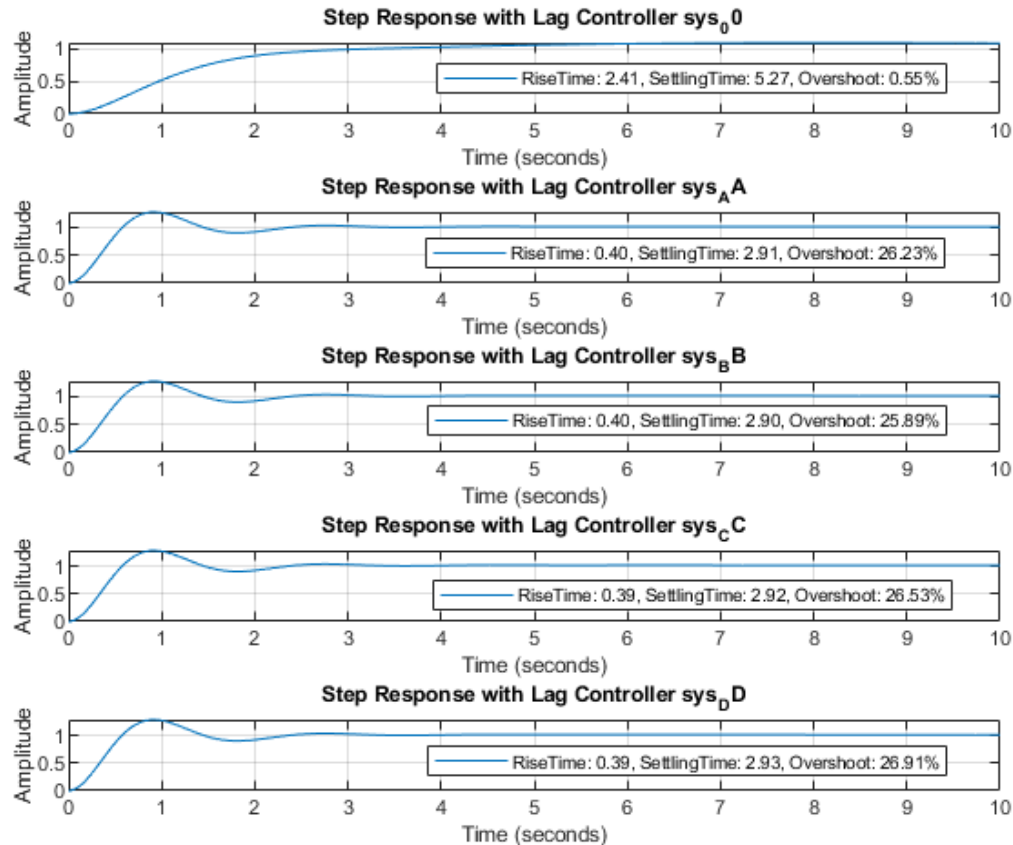
```

title('Step Response with Lag Controller sys_BB');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_BB.RiseTime, info_BB.SettlingTime, info_BB.Overshoot));

subplot(5,1,4);
plot(t_CC, y_CC);
grid on;
title('Step Response with Lag Controller sys_CC');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_CC.RiseTime, info_CC.SettlingTime, info_CC.Overshoot));

subplot(5,1,5);
plot(t_DD, y_DD);
grid on;
title('Step Response with Lag Controller sys_DD');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_DD.RiseTime, info_DD.SettlingTime, info_DD.Overshoot));

```



شکل 18 \_ پاسخ پله پس فاز ها



```
% Closed-loop transfer functions
T1 = feedback(Gc1 * G_with_delay, 1); % Closed-loop transfer function with P
controller
T2 = feedback(Gc2 * G_with_delay, 1); % Closed-loop transfer function with PI
controller
T3 = feedback(Gc3 * G_with_delay, 1); % Closed-loop transfer function with PID
controller
```

```
% Time vector for simulation
t = 0:0.01:10; % Time vector for simulation, from 0 to 10 seconds with 0.01
second steps
```

```
% Step responses
[y1, t1] = step(T1, t); % Step response of system with P controller
[y2, t2] = step(T2, t); % Step response of system with PI controller
[y3, t3] = step(T3, t); % Step response of system with PID controller
```

```
% Display step information (2% settling time criteria)
info_T1 = stepinfo(T1, 'SettlingTimeThreshold', 0.02) % Step response
information for P controller
```

```
info_T1 =
    RiseTime: 0.1033
  TransientTime: 1.6383
    SettlingTime: 1.6586
    SettlingMin: 0.6355
    SettlingMax: 1.2922
    Overshoot: 55.0678
    Undershoot: 27.4233
         Peak: 1.2922
    PeakTime: 0.4367
```

```
info_T2 = stepinfo(T2, 'SettlingTimeThreshold', 0.02) % Step response
information for PI controller
```

```
info_T2 =
    RiseTime: 0.1233
  TransientTime: 1.9139
    SettlingTime: 2.2700
    SettlingMin: 0.7542
    SettlingMax: 1.5755
    Overshoot: 57.5510
    Undershoot: 22.1497
         Peak: 1.5755
    PeakTime: 0.4882
```

```
info_T3 = stepinfo(T3, 'SettlingTimeThreshold', 0.02) % Step response
information for PID controller
```

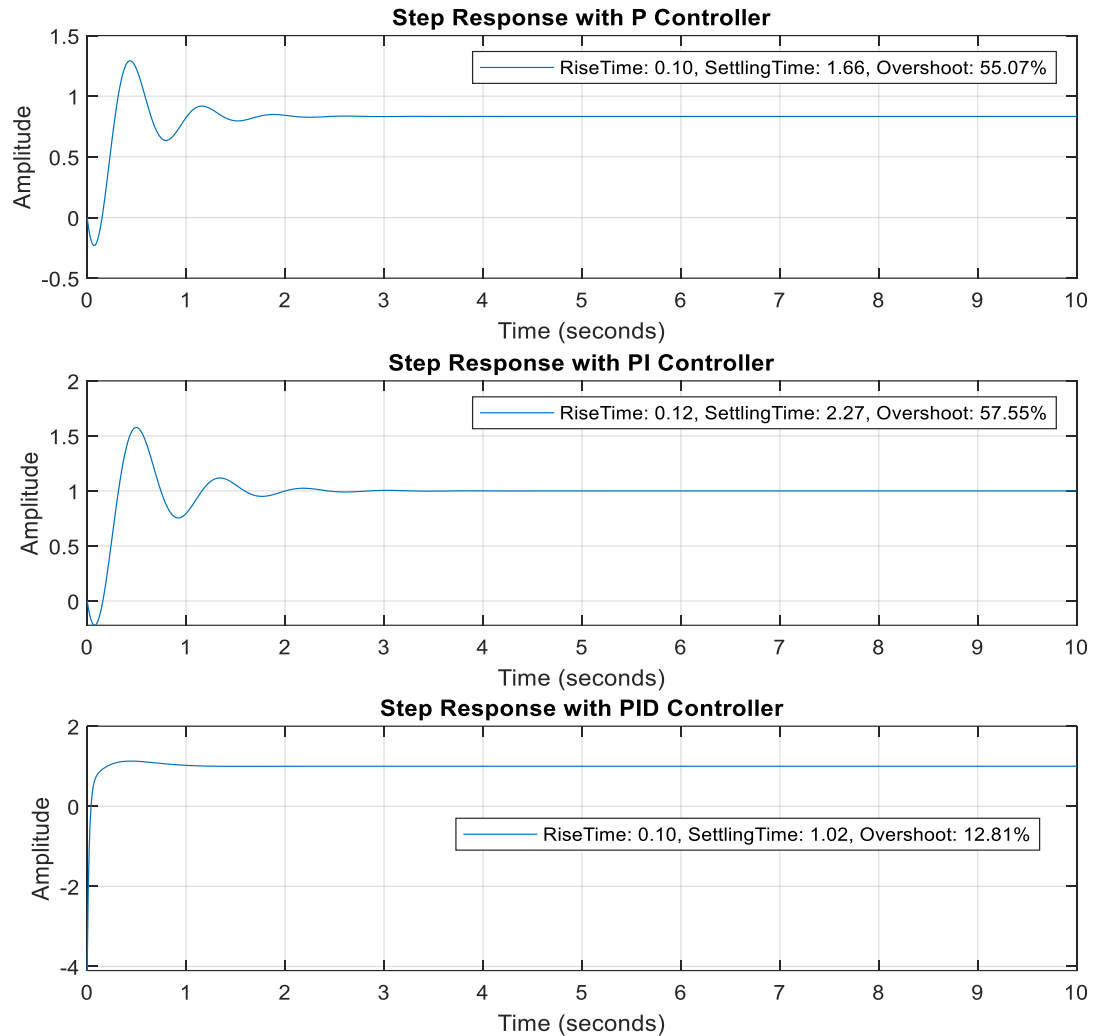
```
info_T3 =
    RiseTime: 0.1003
```

```
TransientTime: 0.6171
SettlingTime: 1.0210
SettlingMin: 0.9014
SettlingMax: 1.1281
Overshoot: 12.8098
Undershoot: 410.2041
Peak: 4.1020
PeakTime: 0
```

```
% Plot the step responses
figure;
subplot(3,1,1);
plot(t1, y1); % Plot step response of P controller
grid on;
title('Step Response with P Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T1.RiseTime, info_T1.SettlingTime, info_T1.Overshoot));

subplot(3,1,2);
plot(t2, y2); % Plot step response of PI controller
grid on;
title('Step Response with PI Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T2.RiseTime, info_T2.SettlingTime, info_T2.Overshoot));

subplot(3,1,3);
plot(t3, y3); % Plot step response of PID controller
grid on;
title('Step Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T3.RiseTime, info_T3.SettlingTime, info_T3.Overshoot));
```



شکل 19\_ پاسخ پله سیستم ها با کنترل کننده های P, PI, PID

```
%% Confirming the steady-state value
```

```
steady_state_value = dcgain(G);
```

```
disp(['Steady-state value of the plant G: ', num2str(steady_state_value)])
```

```
Steady-state value of the plant G: 0.74627
```

```
%% Confirming the steady-state value of the controllers
```

```
steady_state_value1 = dcgain(feedback(Gc1 * G_with_delay, 1));
```

```
disp(['Steady-state value with P Controller: ', num2str(steady_state_value1)])
```

```
Steady-state value with P Controller: 0.83333
```

```
steady_state_value2 = dcgain(feedback(Gc2 * G_with_delay, 1));
```

```
disp(['Steady-state value with PI Controller: ',
```

```
num2str(steady_state_value2)])
```

```
Steady-state value with PI Controller: 1
```

```
steady_state_value3 = dcgain(feedback(Gc3 * G_with_delay, 1));
```

```
disp(['Steady-state value with PID Controller: ',
```

```
num2str(steady_state_value3)])
```

```
Steady-state value with PID Controller: 1
```

```
% Define the ramp input signal
ramp_input = t; % Ramp input signal over time
% Simulate ramp response for closed-loop sys_0
[y_r0, t_r0] = lsim(cl_sys_0, ramp_input, t); % Simulate response using lsim
figure;
subplot(6,1,1);
plot(t_r0, y_r0, 'b', t_r0, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_0');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_A
[y_rA, t_rA] = lsim(cl_sys_A, ramp_input, t); % Simulate response using lsim
subplot(6,1,2);
plot(t_rA, y_rA, 'b', t_rA, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_A');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_B
[y_rB, t_rB] = lsim(cl_sys_B, ramp_input, t); % Simulate response using lsim
subplot(6,1,3);
plot(t_rB, y_rB, 'b', t_rB, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_B');
xlabel('Time (seconds)');
ylabel('Amplitude');

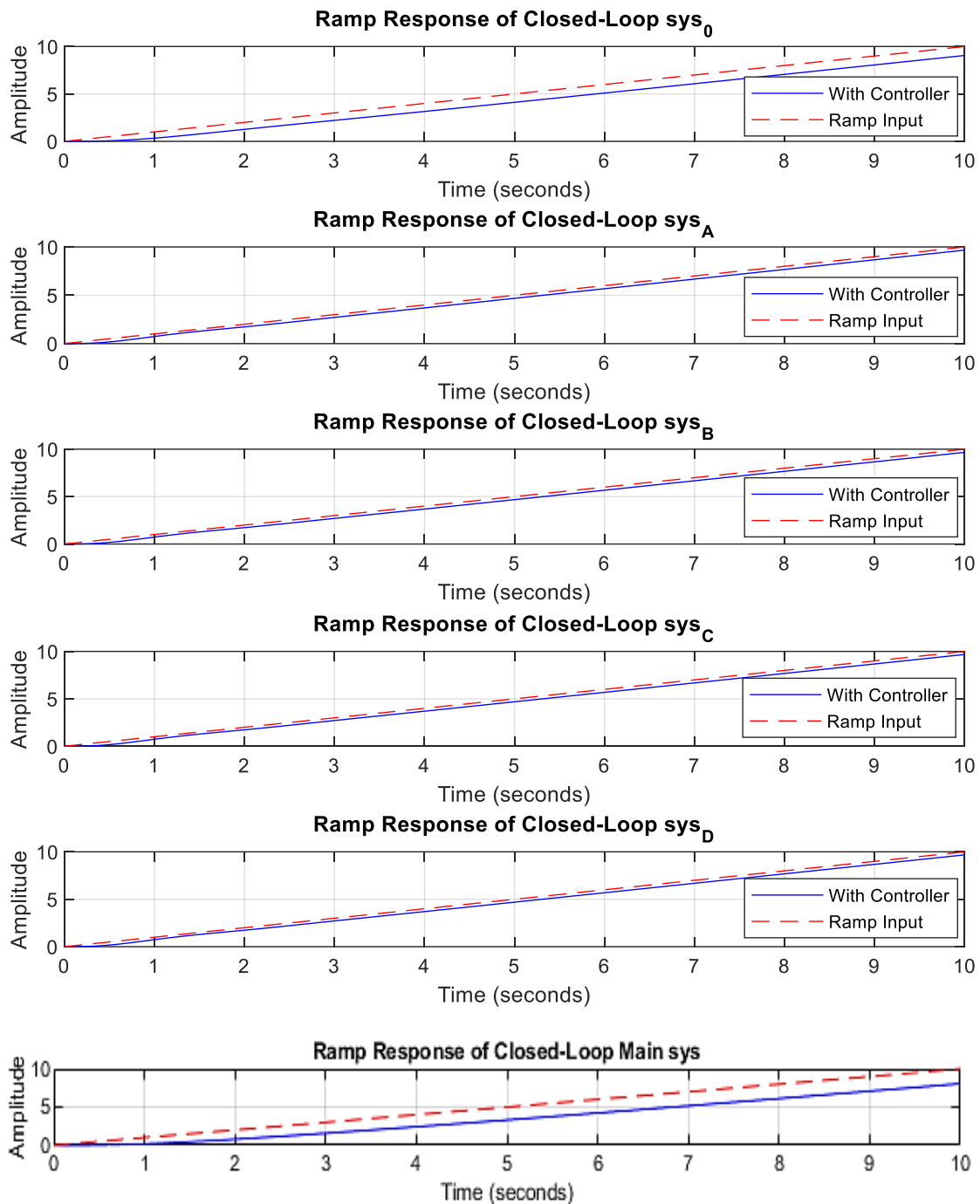
% Simulate ramp response for closed-loop sys_C
[y_rC, t_rC] = lsim(cl_sys_C, ramp_input, t); % Simulate response using lsim
subplot(6,1,4);
plot(t_rC, y_rC, 'b', t_rC, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_C');
xlabel('Time (seconds)');
```

```

ylabel('Amplitude');
% Simulate ramp response for closed-loop sys_D
[y_rD, t_rD] = lsim(cl_sys_D, ramp_input, t); % Simulate response using lsim
subplot(6,1,5);
plot(t_rD, y_rD, 'b', t_rD, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_D');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop Main sys
[y_r, t_r] = lsim(cl_sys, ramp_input, t); % Simulate response using lsim
subplot(6,1,6);
plot(t_r, y_r, 'b', t_r, ramp_input, 'r--'); % Plot system response and ramp
input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop Main sys');
xlabel('Time (seconds)');
ylabel('Amplitude');

```



شکل ۲۰\_ پاسخ به ورودی شیب سیستم بدون کنترل کننده، با کنترل کننده پیش‌فاز، با گین ۱/۲

```

% Time vector for simulation
t = 0:0.01:10;
ramp_input = t; % Ramp input signal over time 't'

% Ramp response for cl_sys_00 (Lag Controller)
[y_r_00, t_r_00] = lsim(cl_sys_00, ramp_input, t);
[y_r_00_sys, t_r_00_sys] = lsim(sys_0, ramp_input, t);
figure;
subplot(5,1,1);
plot(t_r_00, y_r_00, 'b', t_r_00_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_00
legend('With Lag Controller 00', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller 00');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_AA (Lag Controller)
[y_r_AA, t_r_AA] = lsim(cl_sys_AA, ramp_input, t);
[y_r_AA_sys, t_r_AA_sys] = lsim(sys_A, ramp_input, t);
subplot(5,1,2);
plot(t_r_AA, y_r_AA, 'b', t_r_AA_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_AA
legend('With Lag Controller AA', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller AA');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_BB (Lag Controller)
[y_r_BB, t_r_BB] = lsim(cl_sys_BB, ramp_input, t);
[y_r_BB_sys, t_r_BB_sys] = lsim(sys_B, ramp_input, t);
subplot(5,1,3);
plot(t_r_BB, y_r_BB, 'b', t_r_BB_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_BB
legend('With Lag Controller BB', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller BB');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_CC (Lag Controller)
[y_r_CC, t_r_CC] = lsim(cl_sys_CC, ramp_input, t);
[y_r_CC_sys, t_r_CC_sys] = lsim(sys_C, ramp_input, t);
subplot(5,1,4);
plot(t_r_CC, y_r_CC, 'b', t_r_CC_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_CC
legend('With Lag Controller CC', 'Without Controller');

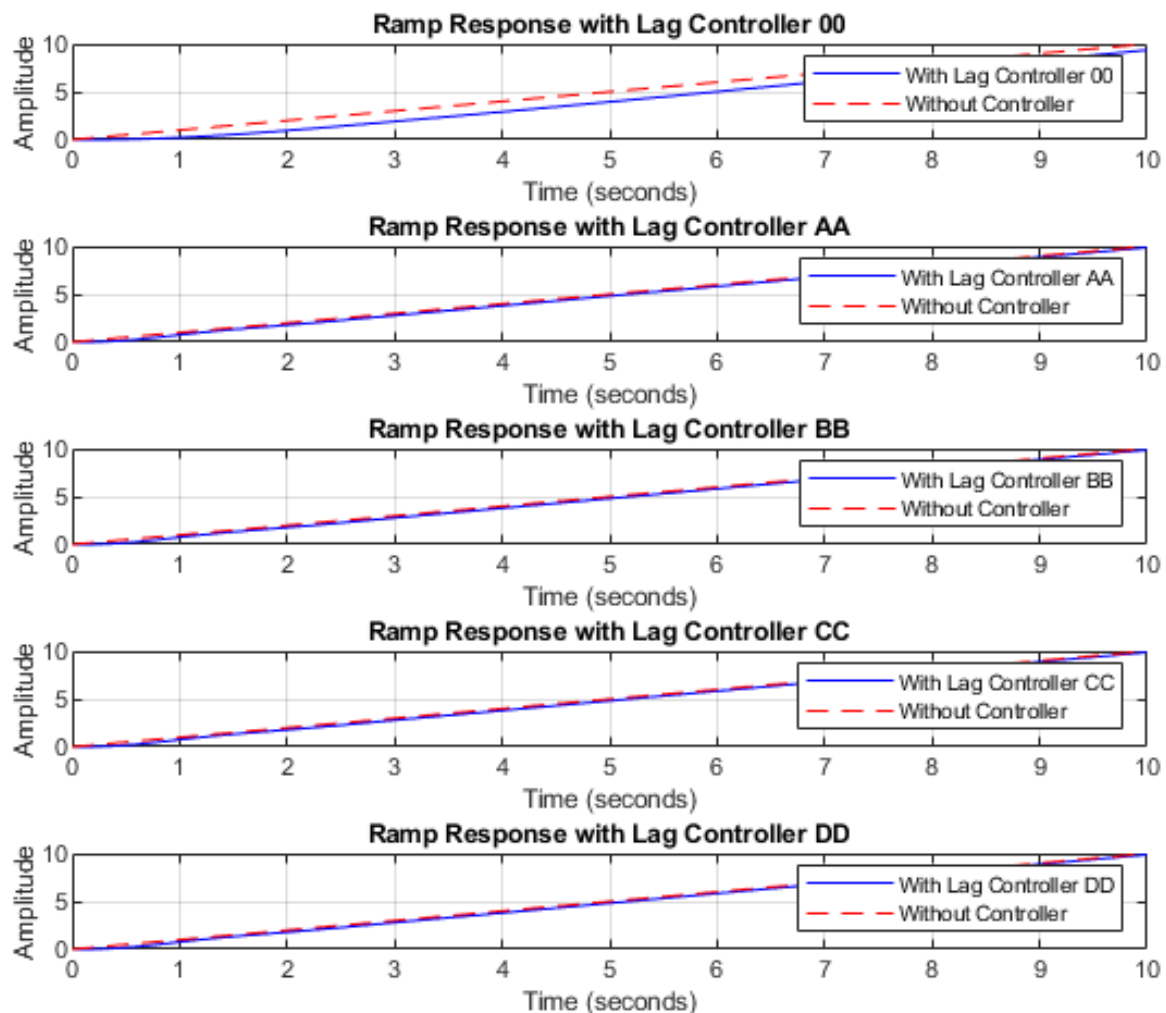
```

```

grid on;
title('Ramp Response with Lag Controller CC');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_DD (Lag Controller)
[y_r_DD, t_r_DD] = lsim(cl_sys_DD, ramp_input, t);
[y_r_DD_sys, t_r_DD_sys] = lsim(sys_D, ramp_input, t);
subplot(5,1,5);
plot(t_r_DD, y_r_DD, 'b', t_r_DD_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_DD
legend('With Lag Controller DD', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller DD');
xlabel('Time (seconds)');
ylabel('Amplitude');

```



شکل 21 \_ پاسخ به ورودی شیب پس فاز ها



```

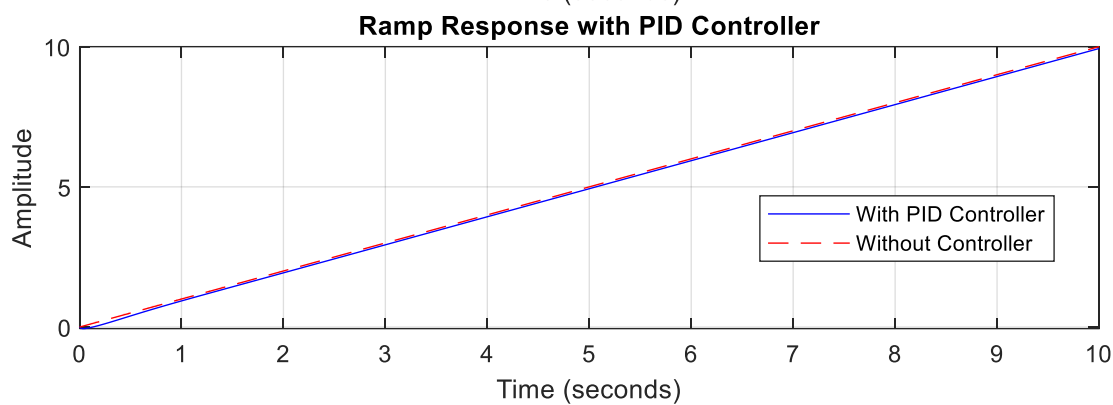
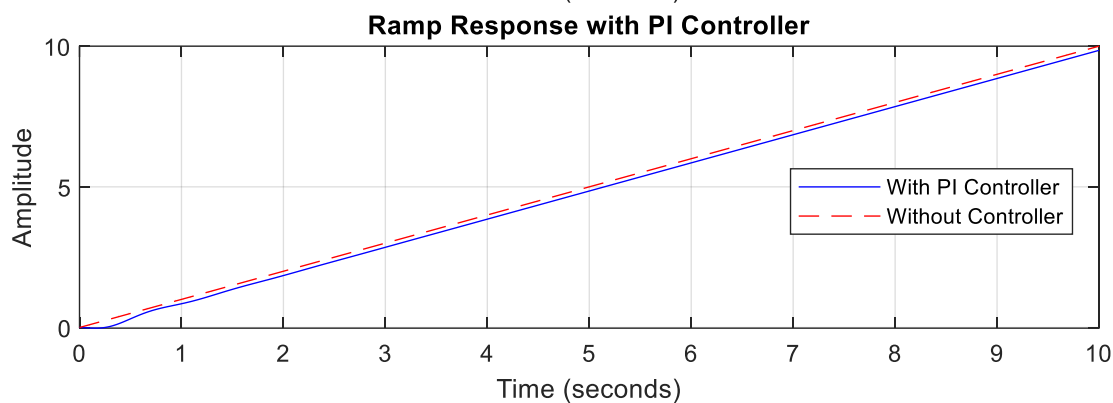
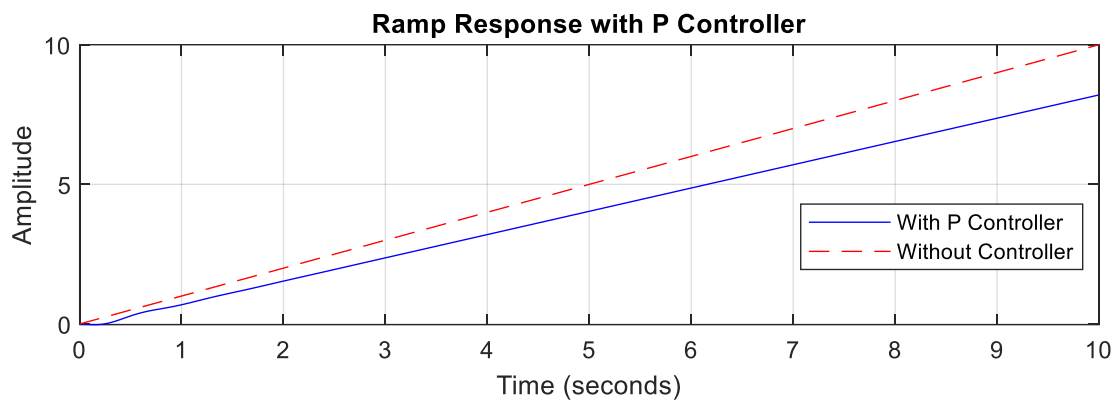
ramp_input = t; % Ramp input signal over time 't'

% Ramp response for T1 (P Controller)
[y_r1, t_r1] = lsim(T1, ramp_input, t); % Simulate ramp response with T1 (P
Controller)
[y_r1_sys, t_r1_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
figure;
subplot(3,1,1);
plot(t_r1, y_r1, 'b', t_r1_sys, ramp_input, 'r--'); % Plot ramp response for
T1
legend('With P Controller', 'Without Controller');
grid on;
title('Ramp Response with P Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for T2 (PI Controller)
[y_r2, t_r2] = lsim(T2, ramp_input, t); % Simulate ramp response with T2 (PI
Controller)
[y_r2_sys, t_r2_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
subplot(3,1,2);
plot(t_r2, y_r2, 'b', t_r2_sys, ramp_input, 'r--'); % Plot ramp response for
T2
legend('With PI Controller', 'Without Controller');
grid on;
title('Ramp Response with PI Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for T3 (PID Controller)
[y_r3, t_r3] = lsim(T3, ramp_input, t); % Simulate ramp response with T3 (PID
Controller)
[y_r3_sys, t_r3_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
subplot(3,1,3);
plot(t_r3, y_r3, 'b', t_r3_sys, ramp_input, 'r--'); % Plot ramp response for
T3
legend('With PID Controller', 'Without Controller');
grid on;
title('Ramp Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

```



شکل ۲۲\_ پاسخ به ورودی شیب سیستم ها با کنترل کننده های  $P, PI, PID$

انجام محاسبات در متلب:

```
% Display poles and zeros of the system
```

```
pole(sys)
```

```
ans = 3×1
      0
     -2
     -1
```

```
zero(sys)
```

```
ans = -0.5000
```

```
% Calculate damping ratio (zeta) and natural frequency (Wn)
```

```
Mp = log(0.3); % Desired percentage overshoot in logarithm form
```

```
zeta = -Mp/sqrt(Mp^2 + pi^2) % Damping ratio calculation
```

```
zeta = 0.3579
```

```
tp = 3; % Desired settling time
```

```
Wn = 4/(tp*zeta) % Natural frequency calculation
```

```
Wn = 3.7259
```

```
beta = acos(zeta); % Angle beta calculation
```

```
betaa = rad2deg(beta); % Convert angle beta to degrees
```

```
betaa = 69.0313
```

```
% Calculate gains for different compensators
```

```
s = -1.33 + 3.47j; % Point in the complex plane for compensator calculations
```

```
num0 = (2*s+1); % Numerator of the original transfer function at point s
```

```
den0 = s*(s+1)*(s+2); % Denominator of the original transfer function at point s
```

```
% Calculating gains K for different compensators
```

```
num_A = num0 * (s+2);
```

```
den_A = den0 * (s+2.13);
```

```
K_A = abs(den_A / num_A)
```

```
K_A = 6.4642
```

```
num_B = num0 * (s+1.5);
```

```
den_B = den0 * (s+1.62);
```

```
K_B = abs(den_B / num_B);
```

```
K_B = 6.4299
```

```
num_C = num0 * (s+2.5);
```

```
den_C = den0 * (s+2.64);
```

```
K_C = abs(den_C / num_C)
```

```
K_C = 6.4978
```

```
num_D = num0 * (s+3.63);
```

```
den_D = den0 * (s+3.81);
```

```
K_D = abs(den_D / num_D)
```

```
K_D = 6.5726
```

```
% Create open-loop systems with compensators
```

```
sys_0 = tf(1.2*num, den) % Open-loop system with initial gain adjustment
```

```
sys_0 =
```

$$\frac{2.4 s + 1.2}{s^3 + 3 s^2 + 2 s}$$

```
Continuous-time transfer function.
```

```
Model Properties
```

```
sys_A = tf(K_A*[2 1], [1 3.13 2.13 0]); % Open-loop system with compensator A
```

```
sys_A =
```

$$\frac{12.93 s + 6.464}{s^3 + 3.13 s^2 + 2.13 s}$$

```
Continuous-time transfer function.
```

```
Model Properties
```

```
sys_B = tf(K_B*[2 4 1.5], [1 4.62 6.86 3.24 0]); % Open-loop system with compensator B
```

```
sys_B =
```

$$\frac{12.86 s^2 + 25.72 s + 9.645}{s^4 + 4.62 s^3 + 6.86 s^2 + 3.24 s}$$

```
Continuous-time transfer function.
```

```
Model Properties
```

```
sys_C = tf(K_C*[2 6 2.5], [1 5.64 9.92 5.28 0]); % Open-loop system with compensator C
```

```
sys_C =
```

$$\frac{13 s^2 + 38.99 s + 16.24}{s^4 + 5.64 s^3 + 9.92 s^2 + 5.28 s}$$

```
Continuous-time transfer function.
```

```
Model Properties
```

```
sys_D = tf(K_D*[2 8.26 3.63], [1 6.81 13.43 7.62 0]); % Open-loop system with compensator D
```

```
sys_D =
```

$$\frac{13.15 s^2 + 54.29 s + 23.86}{s^4 + 6.81 s^3 + 13.43 s^2 + 7.62 s}$$

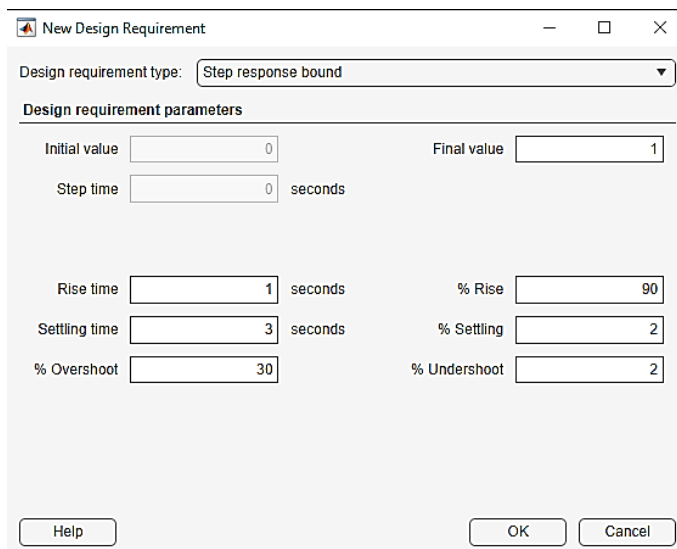
Continuous-time transfer function.

Model Properties

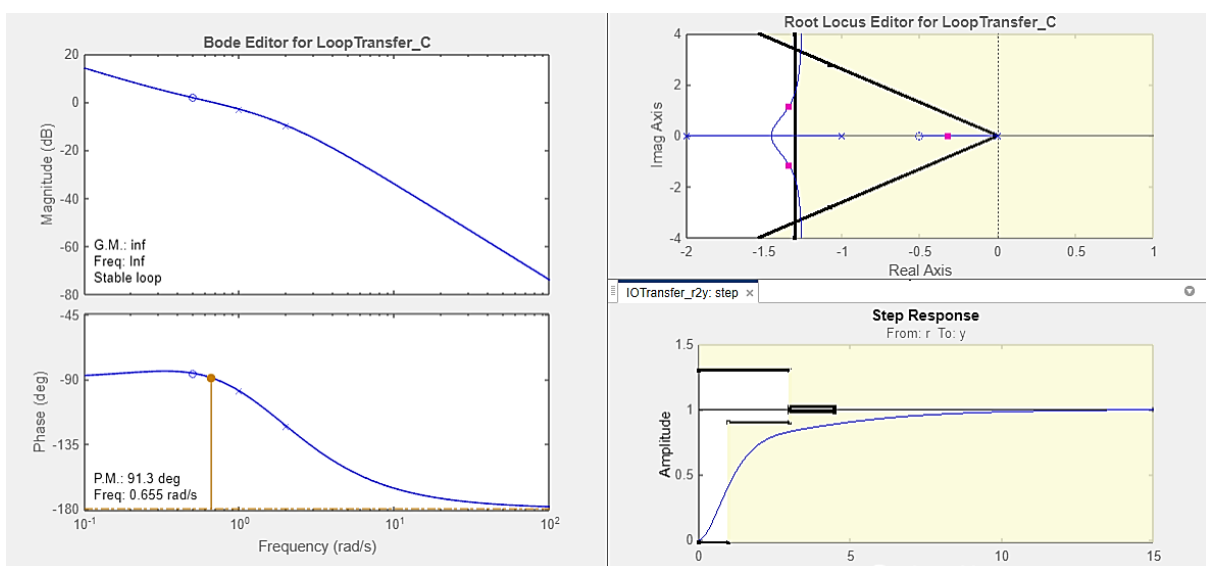
## استفاده از sisotool برای طراحی کنترل کننده

برای طراحی و تنظیم کنترل کننده‌ها، از ابزار SISOtool در MATLAB استفاده شده است. این ابزار امکان تنظیم پارامترهای کنترل کننده و مشاهده تاثیرات آن بر پاسخ سیستم را فراهم می‌کند.

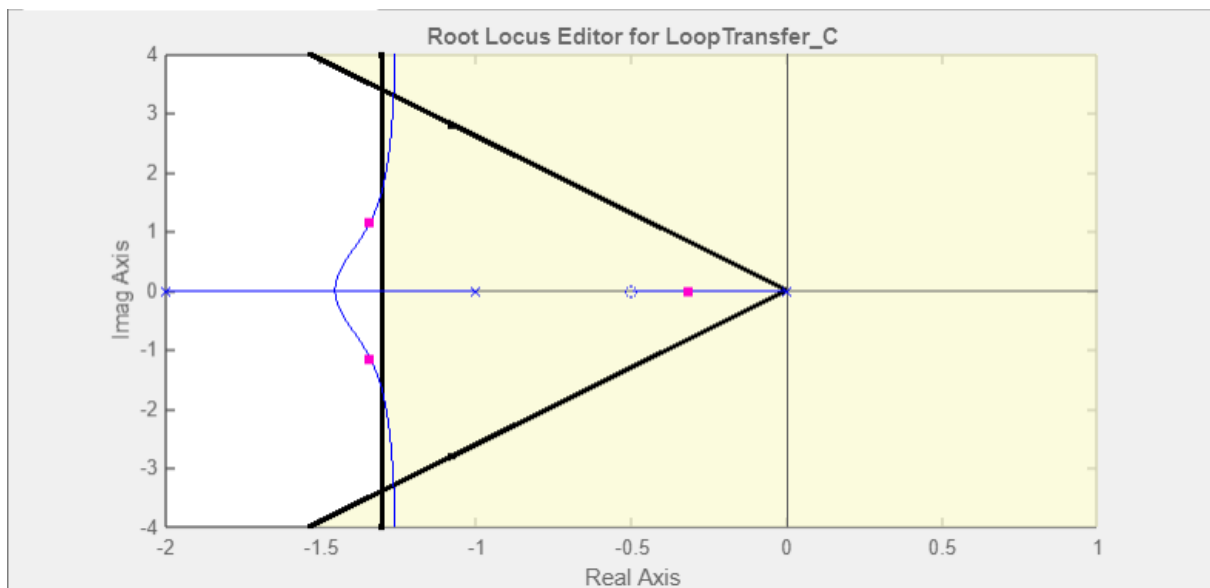
برای استفاده از این ابزار، ابتدا تابع انتقال سیستم را در MATLAB تعریف کرده و سپس با اجرای دستور `'sisotool(sys)'` محیط گرافیکی SISOtool باز می‌شود. در این محیط می‌توان پارامترهای مختلف کنترل کننده‌ها را تنظیم و نتایج را به صورت نموداری مشاهده کرد.



شکل ۲۳\_ پارامتر های SISOtool



شکل ۲۴\_ نتایج نمودار های حاصل از SISOtool



شکل ۲۵ \_ Root Locus حاصل از SISOTool

نقاط برخورد دو خط مشکی همان نقاط مطلوب برای طراحی کنترل کننده ما می باشند.

## تحلیل کلی پاسخ ها

### سیستم بدون کنترل کننده

- **پایداری:** سیستم بدون کنترل کننده پایدار است زیرا همه قطب ها (مکان هندسی ریشه ها) در ناحیه چپ صفحه  $s$  قرار دارند.

- **مشخصات پاسخ پله:**

- زمان نشست 10 ثانیه
- زمان صعود 4.56 ثانیه
- بالازدگی ندارد

این سیستم دارای زمان نشست طولانی، پاسخ کند و بدون بالازدگی است.

### کنترل کننده با $K = 1.2$

- **پایداری:** سیستم با گین ۱,۲ پایدار است و نسبت به سیستم بدون کنترل کننده، پاسخ سریع تری دارد.
- **مشخصات پاسخ پله:**

- a. زمان نشست 8.73 ثانیه
- b. زمان صعود 3.68 ثانیه
- c. بالازدگی ندارد

- **مشخصات:**

در حالت کلی تمام این کنترل کننده باعث بهبود در زمان نشست و زمان صعود شده اند و تاثیری در میزان بالازدگی ندارد.

### کنترل کننده پیش فاز (Lead Controller)

- **مشخصات پاسخ پله:**

- ❖ **پیش فاز اول:**

- زمان نشست 3.65 ثانیه
- زمان صعود 0.4 ثانیه
- بالازدگی 23.34%

- ❖ **پیش فاز دوم:**

- زمان نشست 3.7 ثانیه



○ زمان صعود 0.4 ثانیه

○ بالازدگی 23.04%

❖ پیش فاز سوم:

○ زمان نشست 3.61 ثانیه

○ زمان صعود 0.4 ثانیه

○ بالازدگی 23.64%

❖ پیش فاز با روش هندسی:

○ زمان نشست 2.38 ثانیه

○ زمان صعود 0.4 ثانیه

○ بالازدگی 24.03%

- پایداری: سیستم با کنترل کننده های پیش فاز پایدار است و نسبت به سیستم بدون کنترل کننده، پاسخ سریع تری دارد.
- مشخصات:

در حالت کلی تمام این کنترل کننده ها باعث بهبود در زمان نشست و زمان صعود شده اند. دقت شود در تقریبا تمام حالات زمان نشست و بالازدگی تقریبا به یک مقدار می باشند و این تفاوت به دلیل تقریبی بودن یکسری از پارامترهای محاسبه شده می باشد اما بهترین پاسخ بدست آمده که باعث شده زمان نشست کمی کمتر و بالازدگی آن کمی بیشتر از حالات دیگر شود حالت هندسی می باشد که پاسخ نسبتا سریع تر و مطلوب تری را به ما می دهد.

*کنترل کننده پس فاز (Lag Controller)*

- مشخصات پاسخ پله:

❖ پس فاز برای گین 1.2

○ زمان نشست 5.27 ثانیه

○ زمان صعود 2.41 ثانیه

○ بالازدگی 0.55%

❖ پس فاز اول:

○ زمان نشست 2.91 ثانیه

○ زمان صعود 0.4 ثانیه

○ بالازدگی 26.23%

❖ پس فاز دوم:

○ زمان نشست 2.9 ثانیه

○ زمان صعود 0.4 ثانیه

○ بالازدگی 25.89%

❖ پس فاز سوم:

○ زمان نشست 2.92 ثانیه

○ زمان صعود 0.39 ثانیه

○ بالازدگی 26.53%

❖ پس فاز برای روش هندسی:

○ زمان نشست 2.93 ثانیه

○ زمان صعود 0.39 ثانیه

○ بالازدگی 26.91%

- پایداری: سیستم با کنترل کننده پس فاز نیز پایدار است و پاسخ آن نسبت به سیستم بدون کنترل کننده بهبود یافته است.
- مشخصات:

با اعمال این کنترل کننده ها به سیستمی که برای آن کنترل کننده پیش فاز طراحی شده سعی در کاهش خطای حالت دائم این سیستم ها داشتیم اما انتظار می رود، با اعمال این کنترل کننده ها به سیستم، مکان هندسی ریشه جابه جا شود که همین اتفاق می افتد حال برای آنکه دوباره سیستم در محل قبلی خود قرار گیرد میتوانیم پس از بررسی مجدد سیستم، کنترل کننده پیش فاز جدیدی طراحی کنیم.

در شکل ۲۱ مشاهده میشود تمام پس فاز ها خطای حالت دائم به ورودی شیب را صفر کرده اند به جز پس فاز برای گین ۱,۲ که به این دلیل نامناسب است چون گین ۱,۲ از اول پاسخ پله مناسبی به ما نداد.

#### کنترل کننده PID (روش زیگلر-نیکولز)

از انجایی که طراحی با این روش دقیق نیست و برای تقریبی از سیستم نوشته شده پس انتظار می رود پاسخ دقیقی را در واقعیت به ما ندهد.

• مشخصات پاسخ پله:

○ زمان نشست 1.02 ثانیه

○ زمان صعود 0.1 ثانیه

○ بالازدگی 12.81%

سیستم با استفاده از کنترل کننده PID توانسته است زمان نشست و زمان صعود را کاهش دهد اما با بالازدگی بیشتری مواجه شده است. دقت شود که پاسخ سیستم بهتر از حد انتظار ما (۳ ثانیه) است.

### کنترل کننده PI (روش زیگلر-نیکولز)

- مشخصات پاسخ:

- زمان نشست 2.27 ثانیه
- زمان صعود 0.12 ثانیه
- بالازدگی 57.55%

با استفاده از کنترل کننده PI، زمان نشست و زمان صعود بهبود یافته و بالازدگی نیز در حدود قابل قبولی قرار دارد.

### کنترل کننده P (روش زیگلر-نیکولز)

- مشخصات پاسخ پله:

- زمان نشست 1.66 ثانیه
- زمان صعود 0.1 ثانیه
- بالازدگی 55.07%

این کنترل کننده توانسته است زمان نشست و زمان صعود را بهبود بخشد و بالازدگی را نیز کنترل کند.

### نتیجه گیری

- **بهترین عملکرد:** کنترل کننده PID (روش زیگلر-نیکولز) از نظر زمان نشست و زمان صعود عملکرد بهتری دارد، اما با بالازدگی بیشتری همراه است.
- **عملکرد مناسب:** کنترل کننده PI و تناسبی (روش زیگلر-نیکولز) از نظر زمان نشست و زمان صعود بهبود یافته و بالازدگی قابل قبولی دارند. در میان سایر کنترل کننده ها پیش فاز با روش هندسی مطلوب تر است.
- **پیش فاز و پس فاز:** کنترل کننده های پیش فاز باعث بهبود سریعتر زمان نشست و زمان صعود می شوند، اما با بالازدگی بیشتری همراه هستند. کنترل کننده های پس فاز زمان نشست و زمان صعود طولانی تری دارند اما بالازدگی کمتری دارند همچنین باعث بهبود خطای حالت دائم به ورودی شیب می شوند. در صورتی کنترل کننده های پس فاز بهتر هستند که، یک کنترل کننده پیش فاز دیگر نیز طراحی شود تا مکان هندسی ریشه ها دوباره روی نقاط مطلوب قرار گیرد.
- **بدون کنترل کننده:** سیستم پایدار است ولی پاسخ کند دارد.

این تحلیل ها نشان می دهند که انتخاب کنترل کننده مناسب بستگی به اولویت های سیستم دارد. اگر زمان نشست و زمان صعود مهمتر است، کنترل کننده های PID و PI مناسب هستند. اگر بالازدگی و خطای حالت دائم کمتر اهمیت دارد، کنترل کننده های پس فاز (پس از طراحی پیش فاز ها) بهتر عمل می کنند.

## کل کد متلب جهت ران کردن در Matlab Live Editor

```
% Define the numerator and denominator of the transfer function
num = [2 1]; % Numerator coefficients of '2s+1'
den = conv([1 0], conv([1 1], [1 2])); % Denominator coefficients of
's(s+1)(s+2)'

% Create the transfer function model
sys = tf(num, den)

% Generate and plot the root locus
figure;
rlocus(sys);
hold on
points = [-1.33+3.47j, -1.33-3.47j]; % Points to mark on the root locus plot
plot(points, '*'); % Plotting points on the root locus plot
title('Root Locus of G(s) = (2s+1) / (s(s+1)(s+2))');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Display poles and zeros of the system
pole(sys)
zero(sys)

% Calculate damping ratio (zeta) and natural frequency (Wn)
Mp = log(0.3); % Desired percentage overshoot in logarithm form
zeta = -Mp/sqrt(Mp^2 + pi^2) % Damping ratio calculation
tp = 3; % Desired settling time
Wn = 4/(tp*zeta) % Natural frequency calculation
beta = acos(zeta); % Angle beta calculation
betaa = rad2deg(beta) % Convert angle beta to degrees

% Calculate gains for different compensators
s = -1.33 + 3.47j; % Point in the complex plane for compensator calculations
num0 = (2*s+1); % Numerator of the original transfer function at point s
den0 = s*(s+1)*(s+2); % Denominator of the original transfer function at point
s

% Calculating gains K for different compensators
num_A = num0 * (s+2);
den_A = den0 * (s+2.13);
K_A = abs(den_A / num_A)

num_B = num0 * (s+1.5);
den_B = den0 * (s+1.62);
K_B = abs(den_B / num_B)
```

```

num_C = num0 * (s+2.5);
den_C = den0 * (s+2.64);
K_C = abs(den_C / num_C)

num_D = num0 * (s+3.63);
den_D = den0 * (s+3.81);
K_D = abs(den_D / num_D)

% Create open-loop systems with compensators
sys_0 = tf(1.2*num, den) % Open-loop system with initial gain adjustment
sys_A = tf(K_A*[2 1], [1 3.13 2.13 0]) % Open-loop system with compensator A
sys_B = tf(K_B*[2 4 1.5], [1 4.62 6.86 3.24 0]) % Open-loop system with
compensator B
sys_C = tf(K_C*[2 6 2.5], [1 5.64 9.92 5.28 0]) % Open-loop system with
compensator C
sys_D = tf(K_D*[2 8.26 3.63], [1 6.81 13.43 7.62 0]) % Open-loop system with
compensator D

% Create closed-loop systems with unit feedback
cl_sys_0 = feedback(sys_0, 1); % Closed-loop system with initial gain
adjustment
cl_sys_A = feedback(sys_A, 1); % Closed-loop system with compensator A
cl_sys_B = feedback(sys_B, 1); % Closed-loop system with compensator B
cl_sys_C = feedback(sys_C, 1); % Closed-loop system with compensator C
cl_sys_D = feedback(sys_D, 1); % Closed-loop system with compensator D
cl_sys = feedback(sys, 1); % Closed-loop Main system

% Time vector for simulation
t = 0:0.01:10;

%% Step responses
[y_0, t_0] = step(cl_sys_0, t); % Step response of closed-loop system with
initial gain adjustment
[y_A, t_A] = step(cl_sys_A, t); % Step response of closed-loop system with
compensator A
[y_B, t_B] = step(cl_sys_B, t); % Step response of closed-loop system with
compensator B
[y_C, t_C] = step(cl_sys_C, t); % Step response of closed-loop system with
compensator C
[y_D, t_D] = step(cl_sys_D, t); % Step response of closed-loop system with
compensator D
[y, t] = step(cl_sys, t); % Step response of closed-loop Main system

% Display step information
info_0 = stepinfo(cl_sys_0, 'SettlingTimeThreshold', 0.02) % Step information
of closed-loop system with initial gain adjustment

```

```

info_A = stepinfo(cl_sys_A) % Step information of closed-loop system with
compensator A
info_B = stepinfo(cl_sys_B) % Step information of closed-loop system with
compensator B
info_C = stepinfo(cl_sys_C) % Step information of closed-loop system with
compensator C
info_D = stepinfo(cl_sys_D) % Step information of closed-loop system with
compensator D
info = stepinfo(cl_sys) % Step information of closed-loop Main system

% Display step response plots
figure;
subplot(6,1,1);
plot(t_0, y_0);
grid on;
title('Step Response of Closed-Loop sys_0');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_0.RiseTime, info_0.SettlingTime, info_0.Overshoot));

subplot(6,1,2);
plot(t_A, y_A);
grid on;
title('Step Response of Closed-Loop sys_A');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_A.RiseTime, info_A.SettlingTime, info_A.Overshoot));

subplot(6,1,3);
plot(t_B, y_B);
grid on;
title('Step Response of Closed-Loop sys_B');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_B.RiseTime, info_B.SettlingTime, info_B.Overshoot));

subplot(6,1,4);
plot(t_C, y_C);
grid on;
title('Step Response of Closed-Loop sys_C');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_C.RiseTime, info_C.SettlingTime, info_C.Overshoot));

```

```

subplot(6,1,5);
plot(t_D, y_D);
grid on;
title('Step Response of Closed-Loop sys_D');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_D.RiseTime, info_D.SettlingTime, info_D.Overshoot));

subplot(6,1,6);
plot(t, y);
grid on;
title('Step Response of Closed-Loop Main sys');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info.RiseTime, info.SettlingTime, info.Overshoot));

%% Ramp input response

% Define the ramp input signal
ramp_input = t; % Ramp input signal over time

% Simulate ramp response for closed-loop sys_0
[y_r0, t_r0] = lsim(cl_sys_0, ramp_input, t); % Simulate response using lsim
figure;
subplot(6,1,1);
plot(t_r0, y_r0, 'b', t_r0, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_0');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_A
[y_rA, t_rA] = lsim(cl_sys_A, ramp_input, t); % Simulate response using lsim
subplot(6,1,2);
plot(t_rA, y_rA, 'b', t_rA, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_A');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_B
[y_rB, t_rB] = lsim(cl_sys_B, ramp_input, t); % Simulate response using lsim

```

```

subplot(6,1,3);
plot(t_rB, y_rB, 'b', t_rB, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_B');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_C
[y_rC, t_rC] = lsim(cl_sys_C, ramp_input, t); % Simulate response using lsim
subplot(6,1,4);
plot(t_rC, y_rC, 'b', t_rC, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_C');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop sys_D
[y_rD, t_rD] = lsim(cl_sys_D, ramp_input, t); % Simulate response using lsim
subplot(6,1,5);
plot(t_rD, y_rD, 'b', t_rD, ramp_input, 'r--'); % Plot system response and
ramp input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop sys_D');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Simulate ramp response for closed-loop Main sys
[y_r, t_r] = lsim(cl_sys, ramp_input, t); % Simulate response using lsim
subplot(6,1,6);
plot(t_r, y_r, 'b', t_r, ramp_input, 'r--'); % Plot system response and ramp
input
legend('With Controller', 'Ramp Input');
grid on;
title('Ramp Response of Closed-Loop Main sys');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Define the transfer function  $G(s) = 1 / (1 + 1.34s)$ 
numerator = 1; % Numerator of the transfer function
denominator = [1 1.34]; % Denominator of the transfer function
G = tf(numerator, denominator); % Create transfer function G(s)

% Define the delay

```



```

delay = 0.2; % Time delay in seconds

% Pade approximation for the delay
[num_delay, den_delay] = pade(delay, 1); % 1st-order Pade approximation
Delay_approx = tf(num_delay, den_delay); % Create transfer function for delay approximation

% Combine transfer function with delay approximation
G_with_delay = series(G, Delay_approx); % Series combination of G(s) and delay

% Define controllers
% P Controller
Kp1 = 1.34 / 0.2; % Proportional gain for P controller
Gc1 = Kp1; % Transfer function of P controller

% PI Controller
Kp2 = 0.9 * Kp1; % Proportional gain for PI controller
Ti2 = 0.2 / 0.3; % Integral time constant for PI controller
Gc2 = Kp2 * (1 + tf(1, [Ti2 0])); % Transfer function of PI controller

% PID Controller
Kp3 = 1.2 * Kp1; % Proportional gain for PID controller
Ti3 = 2 * 0.2; % Integral time constant for PID controller
Td3 = 0.5 * 0.2; % Derivative time constant for PID controller
Gc3 = Kp3 * (1 + tf([Td3 0], 1) + tf(1, [Ti3 0])); % Transfer function of PID controller

% Closed-loop transfer functions
T1 = feedback(Gc1 * G_with_delay, 1); % Closed-loop transfer function with P controller
T2 = feedback(Gc2 * G_with_delay, 1); % Closed-loop transfer function with PI controller
T3 = feedback(Gc3 * G_with_delay, 1); % Closed-loop transfer function with PID controller

% Time vector for simulation
t = 0:0.01:10; % Time vector for simulation, from 0 to 10 seconds with 0.01 second steps

% Step responses
[y1, t1] = step(T1, t); % Step response of system with P controller
[y2, t2] = step(T2, t); % Step response of system with PI controller
[y3, t3] = step(T3, t); % Step response of system with PID controller

% Display step information (2% settling time criteria)
info_T1 = stepinfo(T1, 'SettlingTimeThreshold', 0.02) % Step response information for P controller

```

```

info_T2 = stepinfo(T2, 'SettlingTimeThreshold', 0.02) % Step response
information for PI controller
info_T3 = stepinfo(T3, 'SettlingTimeThreshold', 0.02) % Step response
information for PID controller

% Plot the step responses
figure;
subplot(3,1,1);
plot(t1, y1); % Plot step response of P controller
grid on;
title('Step Response with P Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T1.RiseTime, info_T1.SettlingTime, info_T1.Overshoot));

subplot(3,1,2);
plot(t2, y2); % Plot step response of PI controller
grid on;
title('Step Response with PI Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T2.RiseTime, info_T2.SettlingTime, info_T2.Overshoot));

subplot(3,1,3);
plot(t3, y3); % Plot step response of PID controller
grid on;
title('Step Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_T3.RiseTime, info_T3.SettlingTime, info_T3.Overshoot));

%% Confirming the steady-state value
steady_state_value = dcgain(G);
disp(['Steady-state value of the plant G: ', num2str(steady_state_value)]);

%% Confirming the steady-state value of the controllers
steady_state_value1 = dcgain(feedback(Gc1 * G_with_delay, 1));
disp(['Steady-state value with P Controller: ',
num2str(steady_state_value1)]);

steady_state_value2 = dcgain(feedback(Gc2 * G_with_delay, 1));
disp(['Steady-state value with PI Controller: ',
num2str(steady_state_value2)]);

steady_state_value3 = dcgain(feedback(Gc3 * G_with_delay, 1));

```

```

disp(['Steady-state value with PID Controller: ',
num2str(steady_state_value3)]);
%% Ramp input response
ramp_input = t; % Ramp input signal over time 't'

% Ramp response for T1 (P Controller)
[y_r1, t_r1] = lsim(T1, ramp_input, t); % Simulate ramp response with T1 (P
Controller)
[y_r1_sys, t_r1_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
figure;
subplot(3,1,1);
plot(t_r1, y_r1, 'b', t_r1_sys, ramp_input, 'r--'); % Plot ramp response for
T1
legend('With P Controller', 'Without Controller');
grid on;
title('Ramp Response with P Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for T2 (PI Controller)
[y_r2, t_r2] = lsim(T2, ramp_input, t); % Simulate ramp response with T2 (PI
Controller)
[y_r2_sys, t_r2_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
subplot(3,1,2);
plot(t_r2, y_r2, 'b', t_r2_sys, ramp_input, 'r--'); % Plot ramp response for
T2
legend('With PI Controller', 'Without Controller');
grid on;
title('Ramp Response with PI Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for T3 (PID Controller)
[y_r3, t_r3] = lsim(T3, ramp_input, t); % Simulate ramp response with T3 (PID
Controller)
[y_r3_sys, t_r3_sys] = lsim(G_with_delay, ramp_input, t); % Simulate ramp
response without controller
subplot(3,1,3);
plot(t_r3, y_r3, 'b', t_r3_sys, ramp_input, 'r--'); % Plot ramp response for
T3
legend('With PID Controller', 'Without Controller');
grid on;
title('Ramp Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Amplitude');

```

```

%% Root locus plots

% Root locus plot for sys_0 (Controller with K = 1.2)
figure;
rlocus(sys_0);
hold on
points = [-1.33+3.47j, -1.33-3.47j]; % Points to mark on the root locus plot
plot(points, '*');
title('Root Locus of sys K = 1.2');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plots for systems with phase-lead compensators
% Root locus plot for sys_A
figure;
rlocus(sys_A);
hold on
plot(points, '*');
title('Root Locus of sys A');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_B
figure;
rlocus(sys_B);
hold on
plot(points, '*');
title('Root Locus of sys B');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_C
figure;
rlocus(sys_C);
hold on
plot(points, '*');
title('Root Locus of sys C');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

```

```

% Root locus plot for sys_D
figure;
rlocus(sys_D);
hold on
plot(points, '*');
title('Root Locus of sys D');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

%% Add phase-lag controllers and their root locus plots

% Define systems with added phase-lag controllers
sys_00 = series(tf([1 0.1333], [1 0.0024]), sys_0);
sys_AA = series(tf([1 0.1333], [1 0.0119]), sys_A);
sys_BB = series(tf([1 0.1333], [1 0.0117]), sys_B);
sys_CC = series(tf([1 0.1333], [1 0.0121]), sys_C);
sys_DD = series(tf([1 0.1333], [1 0.0123]), sys_D);

% Root locus plot for sys_00
figure;
rlocus(sys_00);
hold on
plot(points, '*');
title('Root Locus of sys K = 1.2 with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_AA
figure;
rlocus(sys_AA);
hold on
plot(points, '*');
title('Root Locus of sys A with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_BB
figure;
rlocus(sys_BB);
hold on
plot(points, '*');
title('Root Locus of sys B with Phase-Lead Controller');

```

```

xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_CC
figure;
rlocus(sys_CC);
hold on
plot(points, '*');
title('Root Locus of sys C with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Root locus plot for sys_DD
figure;
rlocus(sys_DD);
hold on
plot(points, '*');
title('Root Locus of sys D with Phase-Lag Controller');
xlabel('Real Axis');
ylabel('Imaginary Axis');
grid on;
hold off

% Create closed-loop systems with unit feedback
cl_sys_00 = feedback(sys_00, 1); % Closed-loop system with initial gain
adjustment
cl_sys_AA = feedback(sys_AA, 1); % Closed-loop system with compensator A
cl_sys_BB = feedback(sys_BB, 1); % Closed-loop system with compensator B
cl_sys_CC = feedback(sys_CC, 1); % Closed-loop system with compensator C
cl_sys_DD = feedback(sys_DD, 1); % Closed-loop system with compensator D

% Time vector for simulation
t = 0:0.01:10;

% Compute step responses
[y_00, t_00] = step(cl_sys_00, t);
[y_AA, t_AA] = step(cl_sys_AA, t);
[y_BB, t_BB] = step(cl_sys_BB, t);
[y_CC, t_CC] = step(cl_sys_CC, t);
[y_DD, t_DD] = step(cl_sys_DD, t);

% Compute step response characteristics
info_00 = stepinfo(y_00, t_00);
info_AA = stepinfo(y_AA, t_AA);

```

```

info_BB = stepinfo(y_BB, t_BB);
info_CC = stepinfo(y_CC, t_CC);
info_DD = stepinfo(y_DD, t_DD);

% Plot step responses
figure;
subplot(5,1,1);
plot(t_00, y_00);
grid on;
title('Step Response with Lag Controller sys_00');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_00.RiseTime, info_00.SettlingTime, info_00.Overshoot));

subplot(5,1,2);
plot(t_AA, y_AA);
grid on;
title('Step Response with Lag Controller sys_AA');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_AA.RiseTime, info_AA.SettlingTime, info_AA.Overshoot));

subplot(5,1,3);
plot(t_BB, y_BB);
grid on;
title('Step Response with Lag Controller sys_BB');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_BB.RiseTime, info_BB.SettlingTime, info_BB.Overshoot));

subplot(5,1,4);
plot(t_CC, y_CC);
grid on;
title('Step Response with Lag Controller sys_CC');
xlabel('Time (seconds)');
ylabel('Amplitude');
legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_CC.RiseTime, info_CC.SettlingTime, info_CC.Overshoot));

subplot(5,1,5);
plot(t_DD, y_DD);
grid on;
title('Step Response with Lag Controller sys_DD');
xlabel('Time (seconds)');
ylabel('Amplitude');

```

```

legend(sprintf('RiseTime: %.2f, SettlingTime: %.2f, Overshoot: %.2f%%',
info_DD.RiseTime, info_DD.SettlingTime, info_DD.Overshoot));

% Time vector for simulation
t = 0:0.01:10;
ramp_input = t; % Ramp input signal over time 't'

% Ramp response for cl_sys_00 (Lag Controller)
[y_r_00, t_r_00] = lsim(cl_sys_00, ramp_input, t);
[y_r_00_sys, t_r_00_sys] = lsim(sys_0, ramp_input, t);
figure;
subplot(5,1,1);
plot(t_r_00, y_r_00, 'b', t_r_00_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_00
legend('With Lag Controller 00', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller 00');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_AA (Lag Controller)
[y_r_AA, t_r_AA] = lsim(cl_sys_AA, ramp_input, t);
[y_r_AA_sys, t_r_AA_sys] = lsim(sys_A, ramp_input, t);
subplot(5,1,2);
plot(t_r_AA, y_r_AA, 'b', t_r_AA_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_AA
legend('With Lag Controller AA', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller AA');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_BB (Lag Controller)
[y_r_BB, t_r_BB] = lsim(cl_sys_BB, ramp_input, t);
[y_r_BB_sys, t_r_BB_sys] = lsim(sys_B, ramp_input, t);
subplot(5,1,3);
plot(t_r_BB, y_r_BB, 'b', t_r_BB_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_BB
legend('With Lag Controller BB', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller BB');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_CC (Lag Controller)
[y_r_CC, t_r_CC] = lsim(cl_sys_CC, ramp_input, t);
[y_r_CC_sys, t_r_CC_sys] = lsim(sys_C, ramp_input, t);
subplot(5,1,4);

```



```

plot(t_r_CC, y_r_CC, 'b', t_r_CC_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_CC
legend('With Lag Controller CC', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller CC');
xlabel('Time (seconds)');
ylabel('Amplitude');

% Ramp response for cl_sys_DD (Lag Controller)
[y_r_DD, t_r_DD] = lsim(cl_sys_DD, ramp_input, t);
[y_r_DD_sys, t_r_DD_sys] = lsim(sys_D, ramp_input, t);
subplot(5,1,5);
plot(t_r_DD, y_r_DD, 'b', t_r_DD_sys, ramp_input, 'r--'); % Plot ramp response
for cl_sys_DD
legend('With Lag Controller DD', 'Without Controller');
grid on;
title('Ramp Response with Lag Controller DD');
xlabel('Time (seconds)');
ylabel('Amplitude');

```