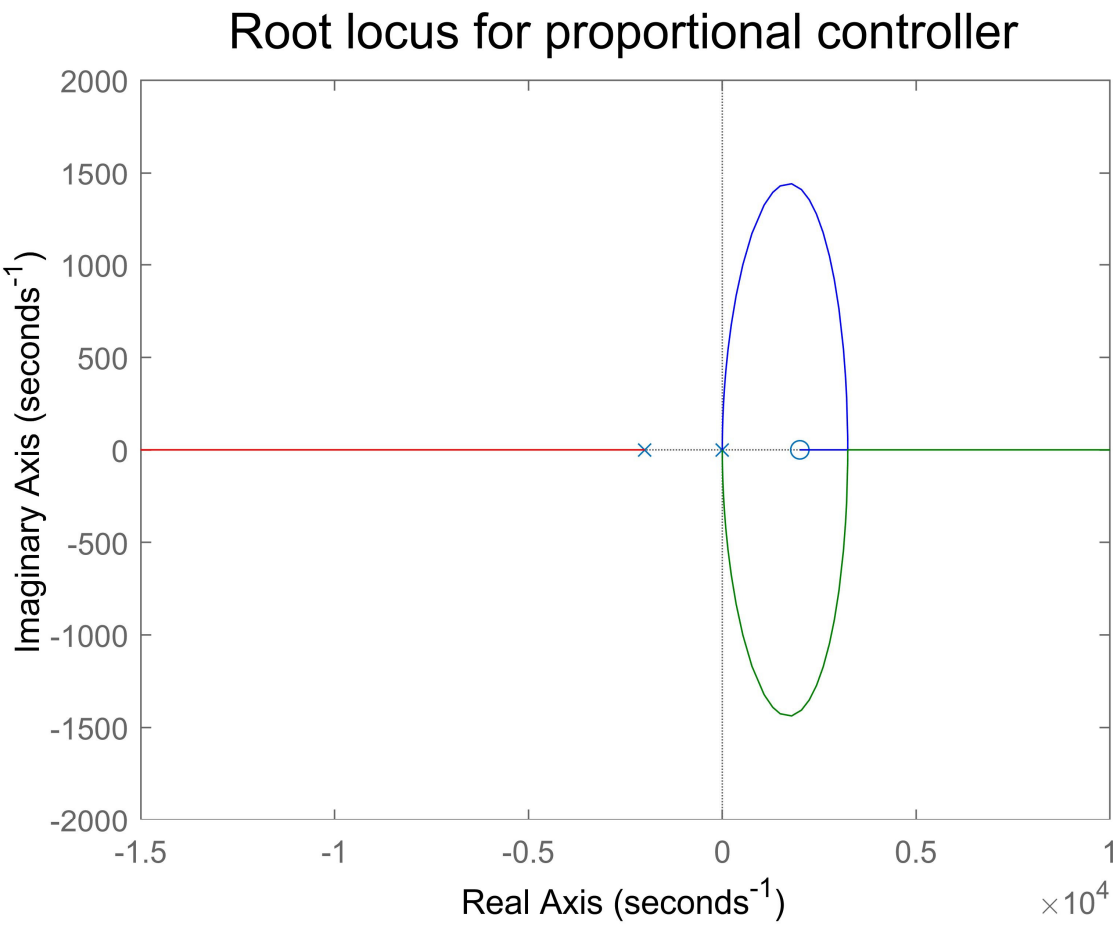
1. Unstable zeros

Code:

```
%% Unstable zeros
sys = tf([-1 2000],[0.03 60 0 0]);

% For k >= 0
rlocus(sys)
title("Root locus for proportional controller", 'Fontsize', 16)

% Find a PD Controller
[C_pd, info] = pidtune(sys, 'PD');

% Closed loop system
tf_cl = minreal(feedback(C_pd*sys, 1));

% From transfer function to state space
[A, B, C, D] = tf2ss([-0.8665 1733 998.3], [1 1999 1733 998.3]);
sys_feedback = ss(A,B,C,D);

% Get the corresponding initial states for the state space
% Using \delta z(0) = y(0) = Cx(0);
% z'(0) = y'(0) = Cx'(0) = CAx(0) + CBu(0), where u (0) = 0
% {z(0) = Cx(0); z'(0) = CAx(0)}, solve for x(0)
% It's an underdetermined system, I used pseudo inverse for one x(0)
equations_matrix = [C; C*A];
initial_values = pinv(equations_matrix)*[-1;0];

% Response with initial values
initial(sys_feedback, initial_values, 4)
title("Response to initial conditions for 4 seconds, k_p=0.015, k_d=0.026", "Fontsize", 12)

initial(sys_feedback, initial_values, 20)
title("Response to initial conditions for 20 seconds, k_p=0.015, k_d=0.026", "Fontsize", 12)
```

b) For the Proportional controller $K(s) = k_p$, it is not possible to find a gain $k_p > 0$, such that the closed-loop dynamics is asymptotically stable. This can be justified by the root locus plot: as long as $k_p > 0$, there are two closed-loop poles on the right half plane, destabilizing the system. The root locus is attached:



For the PD controller, it's possible to find a pair of gain k_p, k_d that stabilize the closed-loop system. The gains are found by using the function pidtune, which are $k_p = 0.015$, $k_d = 0.026$. In order to plot the response with initial conditions specified for the output, the transfer function model needs to be converted to a state space model, using the command tf2ss. However, after the conversion, the three states of the state space model are unknown. To relate the known output initial conditions to the unknown states, the output equation $y = Cx$ is used. The initial conditions for three states satisfy:

$$\Delta z(0) = y(0) = Cx(0) = -1$$

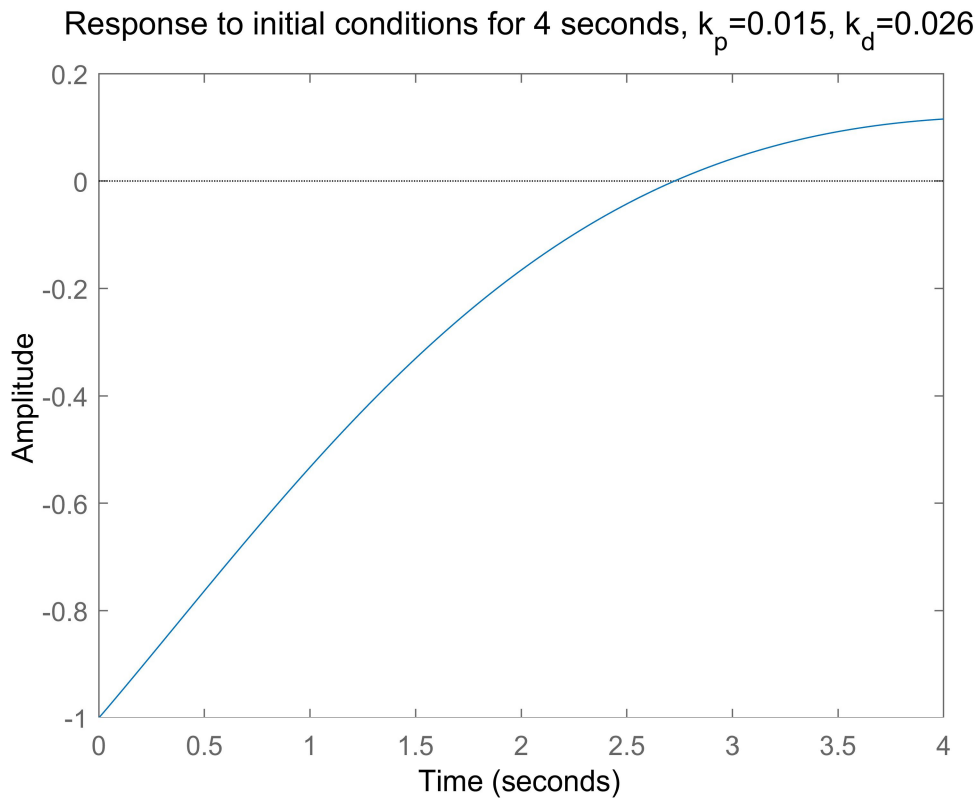
$$z'(0) = y'(0) = Cx'(0) = C\big(Ax(0) + Bu(0)\big) = CAx(0) = 0$$

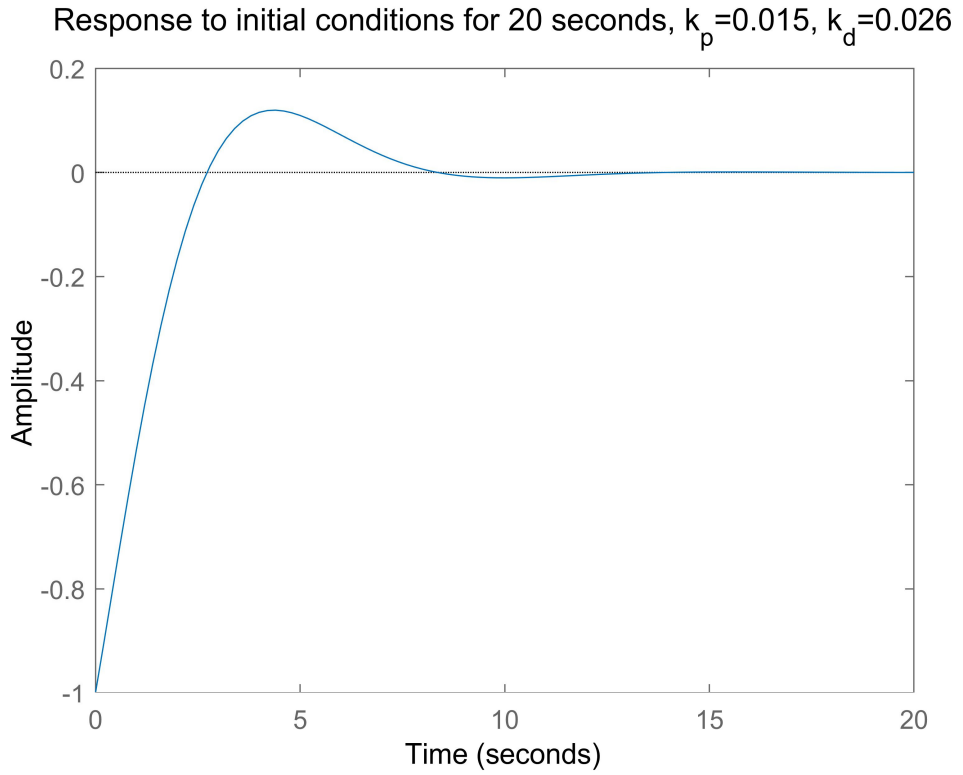
The equations for initial conditions $x(0)$:

$$Cx(0) = -1$$
$$CAx(0) = 0$$

It's an underdetermined system, where there are infinite solutions. Pseudo inverse was used to solve for one solution that satisfies the initial conditions for $\Delta z(0)$ and $z'(0)$.

The output response is plotted for 4 seconds. To show it converges to 0, i.e., the controller asymptotically stabilizes the system, a period of 20 seconds' output response is also attached.





2. Time Delays

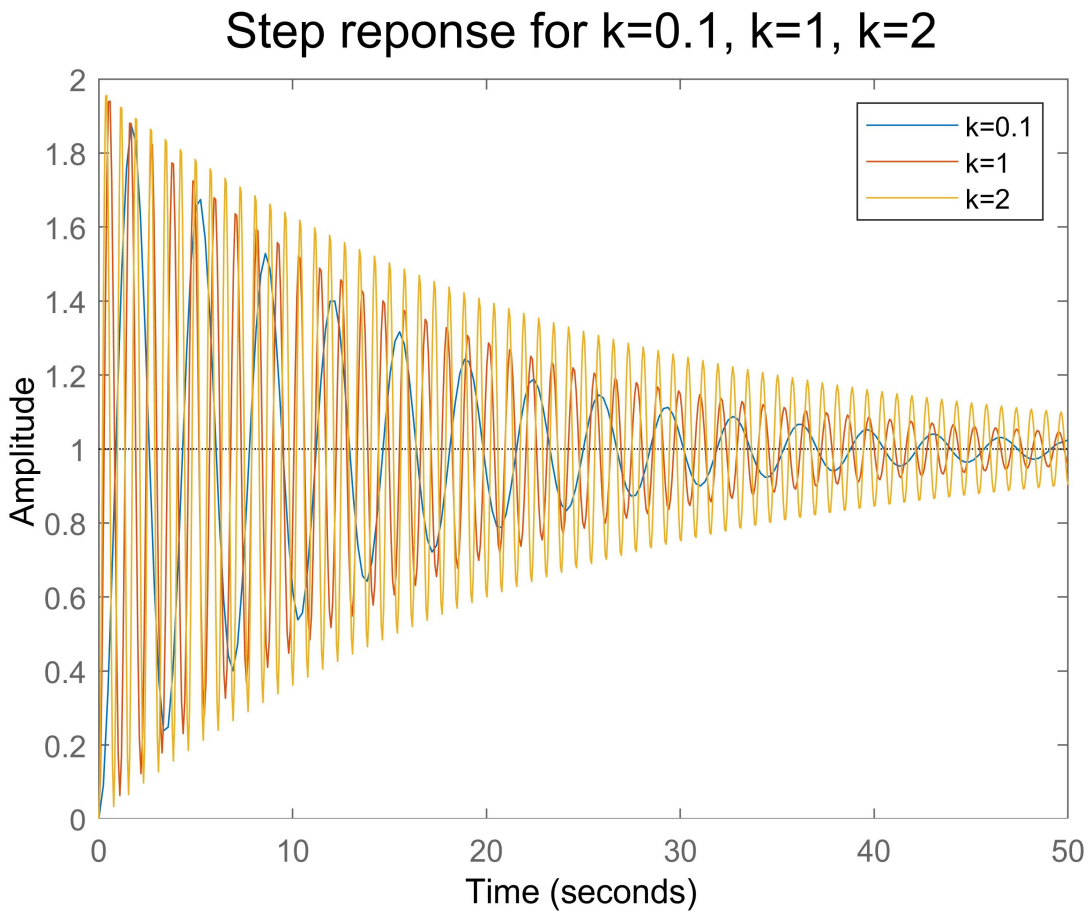
Code:

```
%% Time Delay
k1 = 0.1;
k2 = 1;
k3 = 2;

tf_ol1 = tf(100*k1/3, [1 0.15 0]);
[Gm1,Pm1,Wcg1,Wcp1] = margin(tf_ol1);
max_time_delay1 = deg2rad(Pm1)/Wcp1;

tf_ol2 = tf(100*k2/3, [1 0.15 0]);
[Gm2,Pm2,Wcg2,Wcp2] = margin(tf_ol2);
max_time_delay2 = deg2rad(Pm2)/Wcp2;

tf_ol3 = tf(100*k3/3, [1 0.15 0]);
[Gm3,Pm3,Wcg3,Wcp3] = margin(tf_ol3);
max_time_delay3 = deg2rad(Pm3)/Wcp3;

delay = 1e-3;

tf_delay = tf(100/3,[1 0.15 0], 'InputDelay', delay);
feedback1 = feedback(k1*tf_delay, 1);
feedback2 = feedback(k2*tf_delay, 1);
feedback3 = feedback(k3*tf_delay, 1);
step(feedback1, 50)
hold on
step(feedback2, 50)
hold on
step(feedback3, 50)

legend('k=0.1', 'k=1', 'k=2')
```

Maximum time delay can be calculated using $delay_{max} = \frac{phase\ margin}{crossover\ frequency}$, which are listed:

Gain	$k = 0.1$	$k = 1$	$k = 2$
Max time delay	$45ms$	$4.5ms$	$2.3ms$

The step response for $k = 0.1$, $k = 1$ and $k = 2$ with time delay of $1ms$ is plotted:



3. Nyquist Diagram

Code:

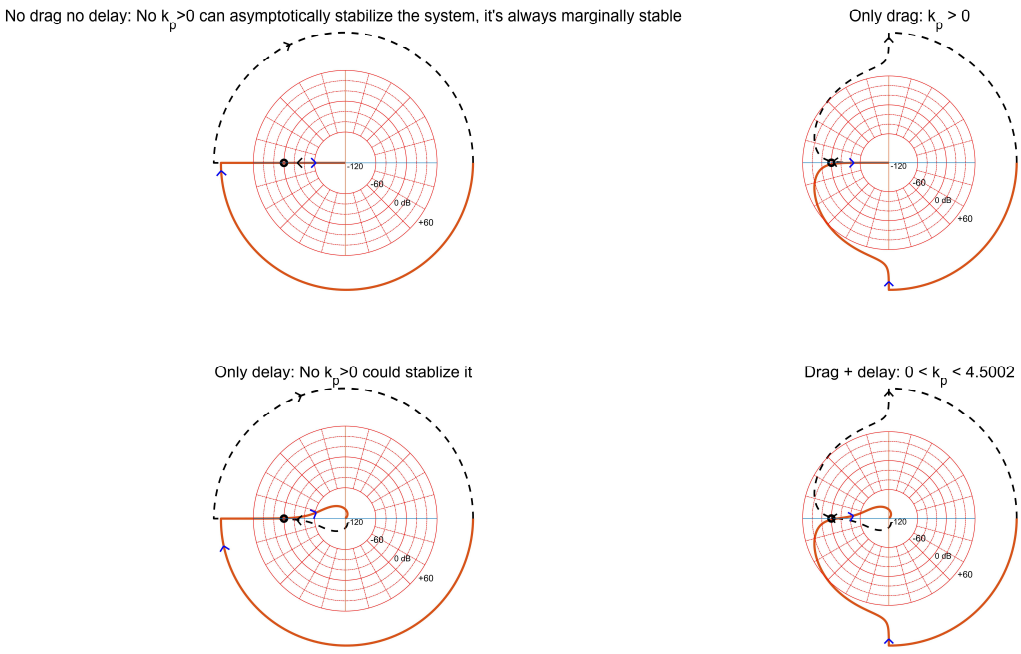
```
%% Nyquist plot
sys_ndnd = tf(100, [3 0 0]);
subplot(2,2,1)
nyqlog(sys_ndnd)
title("No drag no delay: k_p >= 0")

sys_drag = tf(100, [3 0.45 0]);
subplot(2,2,2)
nyqlog(sys_drag)
title("Only drag: k_p >= 0")

sys_delay = tf([-0.1 200],[0.003 6 0 0]);
subplot(2,2,3)
nyqlog(sys_delay)
```

```
[Gm,Pm,Wcg,Wcp] = margin(sys_delay);
title("Only delay: No proportional gain could stablize it")


sys_dd = tf([-0.1 200], [0.003 6.00045 0.9 0]);
subplot(2,2,4)
nyqlog(sys_dd)
[Gm,Pm,Wcg,Wcp] = margin(sys_dd);
title("Drag + delay: 0 <= k_p <= 4.5")
```

The range of k_p that could stabilize the closed-loop system can be identified by the gain margin and/or the closed-loop poles.

- (a): Since the closed-loop poles always lay on the imaginary axis, there is no $k_p > 0$ that could asymptotically stabilize the closed loop system. It's always marginally stable.
- (b): The Nyquist plot will never encircle $(-\frac{1}{k_p}, 0)$, where $k_p > 0$, namely its gain margin is infinite. So, the range for k_p that asymptotically stabilize the closed-loop system is $k_p > 0$.
- (c): The Nyquist plot will always encircle $(-\frac{1}{k_p}, 0)$ twice, where $k_p > 0$. As long as $k_p > 0$, there will be closed-loop pole on the right half plane, according to the Nyquist theorem. So, there is no $k_p > 0$ that could stabilize the system.
- (d): The gain margin is (about) 4.5002, meaning as long as $0 < k_p < 4.5002$, the closed-loop system is asymptotically stable.

Nyquist plots:

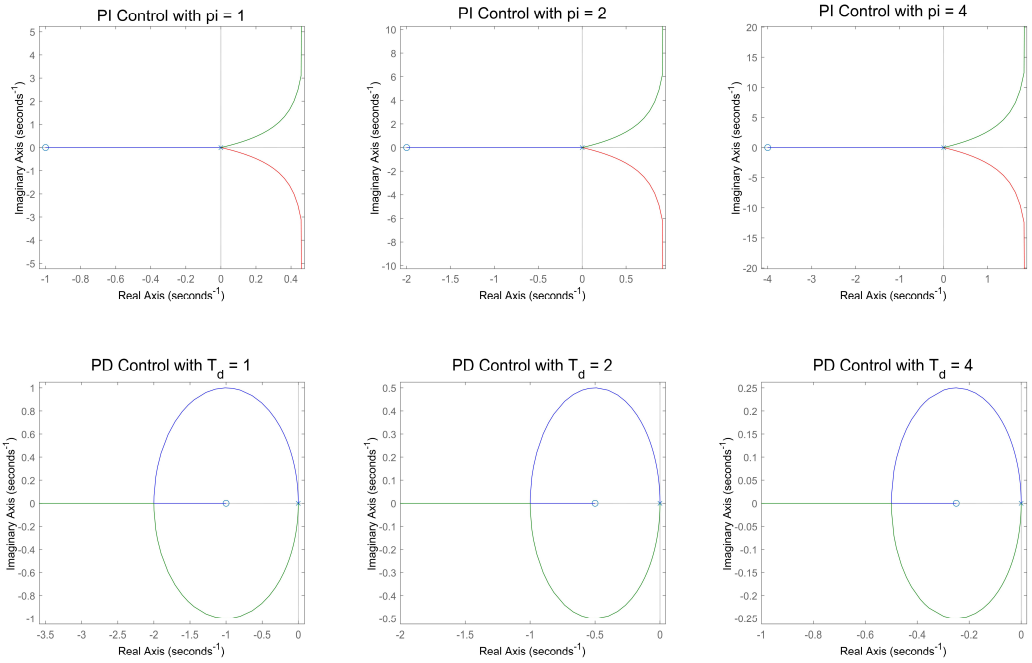


4. PID Control

Code:

```
%% PID Control
p1 = 1;
p2 = 2;
p3 = 4;

t1 = 1;
t2 = 2;
t3 = 4;

pi_ol1 = tf([1 p1], [0.03 0 0 0]);
pi_ol2 = tf([1 p2], [0.03 0 0 0]);
pi_ol3 = tf([1 p3], [0.03 0 0 0]);

subplot(2,3,1)
rlocus(pi_ol1)
title('PI Control with pi = 1')

subplot(2,3,2)
rlocus(pi_ol2)
title('PI Control with pi = 2')

subplot(2,3,3)
rlocus(pi_ol3)
title('PI Control with pi = 4')

pd_ol1 = tf([t1 1], [0.03 0 0]);
pd_ol2 = tf([t2 1], [0.03 0 0]);
pd_ol3 = tf([t3 1], [0.03 0 0]);

subplot(2,3,4)
rlocus(pd_ol1)
title('PD Control with T_d = 1')

subplot(2,3,5)
rlocus(pd_ol2)
title('PD Control with T_d = 2')

subplot(2,3,6)
rlocus(pd_ol3)
title('PD Control with T_d = 4')
```

Root locus plots:



(c) In this part, the root locus plots with indicators attached is attached below for clearer explanation of these questions.

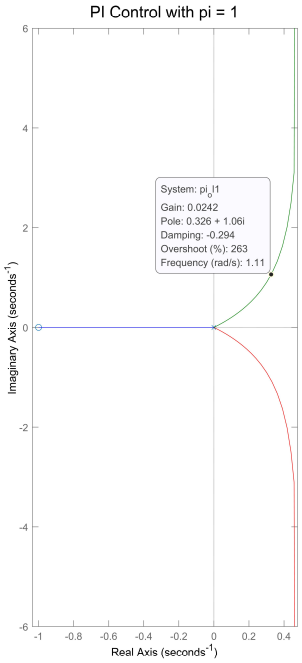
i. According to the root locus plots, the PI controller destabilizes the closed-loop system.

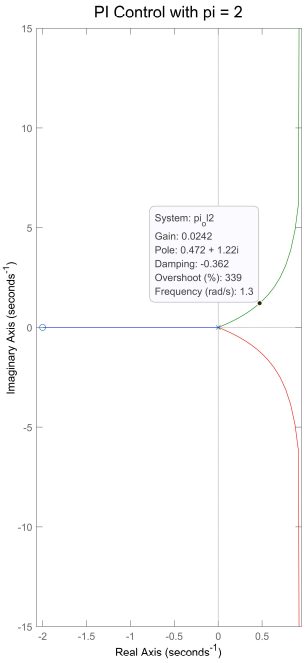
ii. As the integral gain p_i increases, with the same proportional gain, the real parts of the unstable poles of the closed-loop system become higher, as shown in the first attached plot.

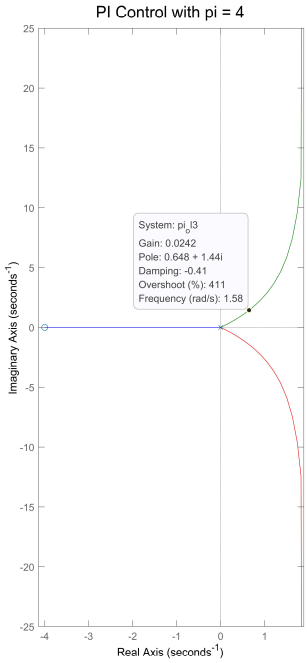
iii. PD controller asymptotically stabilize the closed-loop system. Because the closed-loop poles are on the left-half plane, as shown second attached plots.

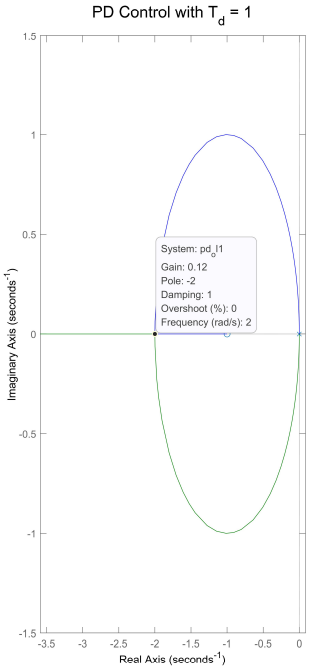
iv. As T_d increases, the proportional gain required for critically damped becomes smaller, as shown in the second attached plot.

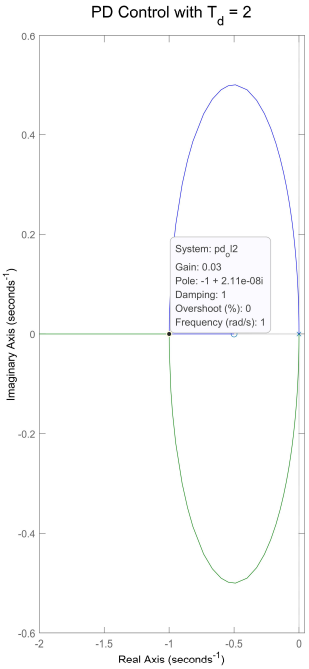
v. With gain k_p that three systems are all critically damped, the convergence rate decreases as T_d increases, as shown in the second attached plot. Because the magnitude of the real part of the two critically damped poles becomes smaller as T_d increases.

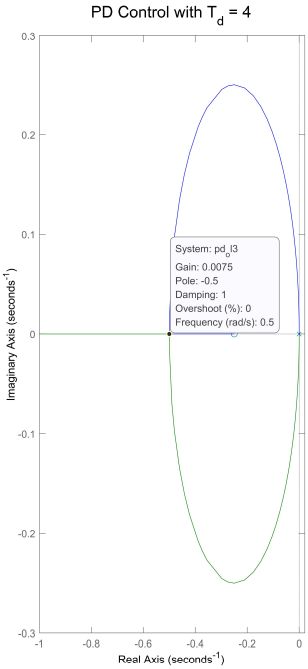

PI Control with pi = 1
System: pi_o_l1
Gain: 0.0242
Pole: 0.326 + 1.06i
Damping: -0.294
Overshoot (%): 263
Frequency (rad/s): 1.11
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)


PI Control with pi = 2
System: pi_o_l2
Gain: 0.0242
Pole: 0.472 + 1.22i
Damping: -0.362
Overshoot (%): 339
Frequency (rad/s): 1.3
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)


PI Control with pi = 4
System: pi_o_l3
Gain: 0.0242
Pole: 0.648 + 1.44i
Damping: -0.41
Overshoot (%): 411
Frequency (rad/s): 1.58
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)


PD Control with T_d = 1
System: pd_o_l1
Gain: 0.12
Pole: -2
Damping: 1
Overshoot (%): 0
Frequency (rad/s): 2
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)


PD Control with T_d = 2
System: pd_o_l2
Gain: 0.03
Pole: -1 + 2.11e-08i
Damping: 1
Overshoot (%): 0
Frequency (rad/s): 1
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)

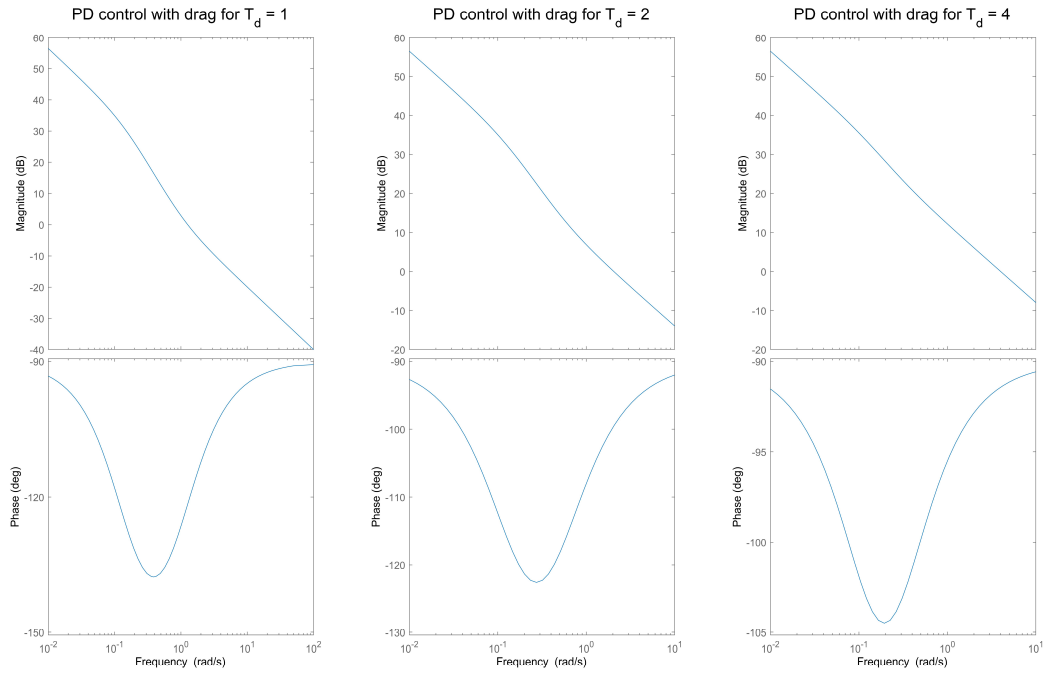

PD Control with T_d = 4
System: pd_o_l3
Gain: 0.0075
Pole: -0.5
Damping: 1
Overshoot (%): 0
Frequency (rad/s): 0.5
Imaginary Axis (seconds^-1)
Real Axis (seconds^-1)

5. Bode Diagrams

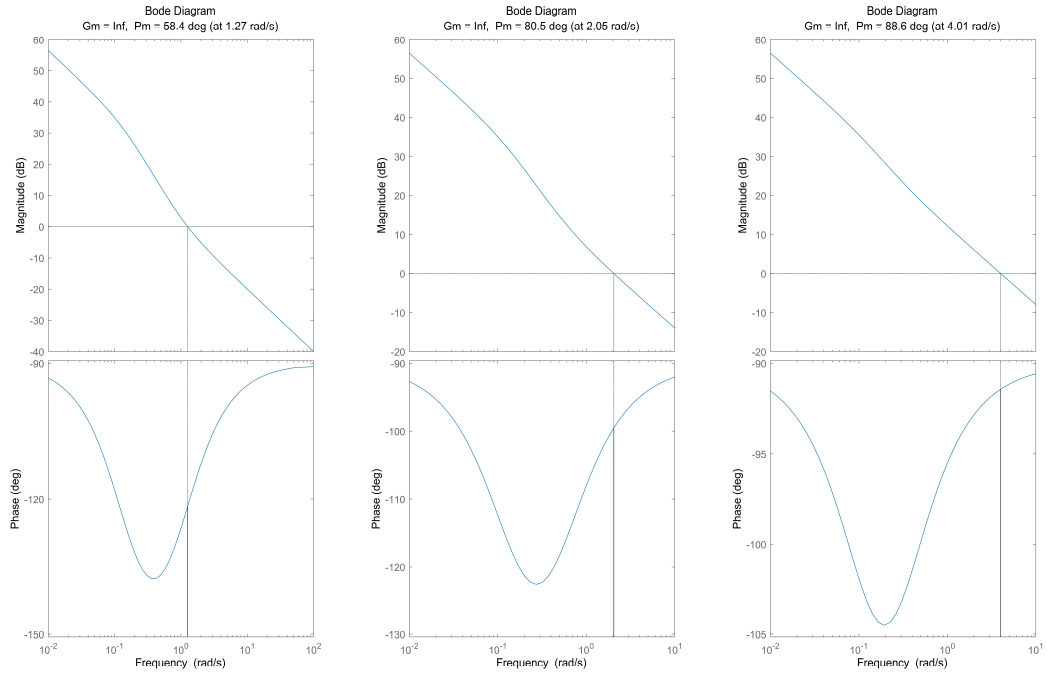
Code:

```
%% Bode Plot
t1 = 1;
t2 = 2;
t3 = 4;

pd_ol_drag1 = tf([t1 1],[1 0.15 0]);
pd_ol_drag2 = tf([t2 1],[1 0.15 0]);
pd_ol_drag3 = tf([t3 1],[1 0.15 0]);

subplot(1,3,1)
bode(pd_ol_drag1)
title('PD control with drag for T_d = 1','fontsize',16)
[Gm1,Pm1,Wcg1,Wcp1] = margin(pd_ol_drag1);

subplot(1,3,2)
bode(pd_ol_drag2)
title('PD control with drag for T_d = 2','fontsize',16)
[Gm2,Pm2,Wcg2,Wcp2] = margin(pd_ol_drag2);

subplot(1,3,3)
bode(pd_ol_drag3)
title('PD control with drag for T_d = 4','fontsize',16)
[Gm3,Pm3,Wcg3,Wcp3] = margin(pd_ol_drag3);

subplot(1,3,1)
margin(pd_ol_drag1);
subplot(1,3,2)
margin(pd_ol_drag2);
subplot(1,3,3)
margin(pd_ol_drag3);
```

Bode Diagrams:



(c) The gain margin is all infinity, regardless of the differential gain T_d. The phase margin increases as T_d increases. The results can be shown in the bode plot with margins indicated, which is attached below.



Full MATLAB code:

```
%% Unstable zeros
sys = tf([-1 2000],[0.03 60 0 0]);

% For k >= 0
rlocus(sys)
title("Root locus for proportional controller", 'Fontsize', 16)

% Find a PD Controller
[C_pd, info] = pidtune(sys, 'PD');
%C_pd = tf([0.15 0.22],[1]);

% Closed loop system
tf_cl = minreal(feedback(C_pd*sys, 1));

% From transfer function to state space
[A, B, C, D] = tf2ss([-0.8665 1733 998.3], [1 1999 1733 998.3]);
sys_feedback = ss(A,B,C,D);

% Get the corresponding initial states for the state space
% Using \delta z(0) = y(0) = Cx(0);
% z'(0) = y'(0) = Cx'(0) = CAx(0) + CBu(0), where u (0) = 0
% {z(0) = Cx(0); z'(0) = CAx(0)}, solve for x(0)
% It's an underdetermined system, I used pseudo inverse for one x(0)
equations_matrix = [C*A; C];
initial_values = pinv(equations_matrix)*[0;-1];

% Response with initial values
initial(sys_feedback, initial_values, 4)
title("Response to initial conditions for 4 seconds, k_p=0.015, k_d=0.026", "Fontsize", 12)

initial(sys_feedback, initial_values, 4)
title("Response to initial conditions for 20 seconds, k_p=0.015, k_d=0.026", "Fontsize", 12)
%% Time Delay
k1 = 0.1;
k2 = 1;
k3 = 2;

tf_ol1 = tf(100*k1/3, [1 0.15 0]);
[Gm1,Pm1,Wcg1,Wcp1] = margin(tf_ol1);
max_time_delay1 = deg2rad(Pm1)/Wcp1;

tf_ol2 = tf(100*k2/3, [1 0.15 0]);
[Gm2,Pm2,Wcg2,Wcp2] = margin(tf_ol2);
```

```
max_time_delay2 = deg2rad(Pm2)/Wcp2;


tf_ol3 = tf(100*k3/3, [1 0.15 0]);
[Gm3,Pm3,Wcg3,Wcp3] = margin(tf_ol3);
max_time_delay3 = deg2rad(Pm3)/Wcp3;



delay = 1e-3;

tf_delay = tf(100/3,[1 0.15 0], 'InputDelay', delay);
feedback1 = feedback(k1*tf_delay, 1);
feedback2 = feedback(k2*tf_delay, 1);
feedback3 = feedback(k3*tf_delay, 1);
step(feedback1, 50)
hold on
step(feedback2, 50)
hold on
step(feedback3, 50)
title("Step reponse for k=0.1, k=1, k=2","Fontsize",16)

legend('k=0.1', 'k=1', 'k=2')

%% Nyquist plot
sys_ndnd = tf(100, [3 0 0]);
subplot(2,2,1)
nyqlog(sys_ndnd)
title("No drag no delay: No k_p>0 can asymptotically stabilize the system, it's always marginally
stable",'fontsize',16)
[Gm1,Pm1,Wcg1,Wcp1] = margin(sys_ndnd);

sys_drag = tf(100, [3 0.45 0]);
subplot(2,2,2)
nyqlog(sys_drag)
title("Only drag: k_p > 0",'fontsize',16)
[Gm2,Pm2,Wcg2,Wcp2] = margin(sys_drag);
Gm2

sys_delay = tf([-0.1 200],[0.003 6 0 0]);
subplot(2,2,3)
nyqlog(sys_delay)
[Gm3,Pm3,Wcg3,Wcp3] = margin(sys_delay);
title("Only delay: No k_p>0 could stablize it",'fontsize',16)
Gm3
```

```
sys_dd = tf([-0.1 200], [0.003 6.00045 0.9 0]);
subplot(2,2,4)
nyqlog(sys_dd)
[Gm4,Pm4,Wcg4,Wcp4] = margin(sys_dd);
title("Drag + delay: 0 < k_p < 4.5002",'fontsize',16)
Gm4
%% PID Control
p1 = 1;
p2 = 2;
p3 = 4;

t1 = 1;
t2 = 2;
t3 = 4;

pi_ol1 = tf([1 p1], [0.03 0 0 0]);
pi_ol2 = tf([1 p2], [0.03 0 0 0]);
pi_ol3 = tf([1 p3], [0.03 0 0 0]);

% subplot(1,3,1)
% rlocus(pi_ol1)
% title('PI Control with pi = 1','fontsize',16)
%
% subplot(1,3,2)
% rlocus(pi_ol2)
% title('PI Control with pi = 2','fontsize',16)
%
% subplot(1,3,3)
% rlocus(pi_ol3)
% title('PI Control with pi = 4','fontsize',16)

pd_ol1 = tf([t1 1], [0.03 0 0]);
pd_ol2 = tf([t2 1], [0.03 0 0]);
pd_ol3 = tf([t3 1], [0.03 0 0]);

subplot(1,3,1)
rlocus(pd_ol1)
title('PD Control with T_d = 1','fontsize',16)

subplot(1,3,2)
rlocus(pd_ol2)
title('PD Control with T_d = 2','fontsize',16)

subplot(1,3,3)
```

```
rlocus(pd_ol3)
title('PD Control with T_d = 4','fontsize',16)

%% Bode Plot
t1 = 1;
t2 = 2;
t3 = 4;

pd_ol_drag1 = tf([t1 1],[1 0.15 0]);
pd_ol_drag2 = tf([t2 1],[1 0.15 0]);
pd_ol_drag3 = tf([t3 1],[1 0.15 0]);

subplot(1,3,1)
bode(pd_ol_drag1)
title('PD control with drag for T_d = 1','fontsize',16)
[Gm1,Pm1,Wcg1,Wcp1] = margin(pd_ol_drag1);

subplot(1,3,2)
bode(pd_ol_drag2)
title('PD control with drag for T_d = 2','fontsize',16)
[Gm2,Pm2,Wcg2,Wcp2] = margin(pd_ol_drag2);

subplot(1,3,3)
bode(pd_ol_drag3)
title('PD control with drag for T_d = 4','fontsize',16)
[Gm3,Pm3,Wcg3,Wcp3] = margin(pd_ol_drag3);

subplot(1,3,1)
margin(pd_ol_drag1);
subplot(1,3,2)
margin(pd_ol_drag2);
subplot(1,3,3)
margin(pd_ol_drag3);
```