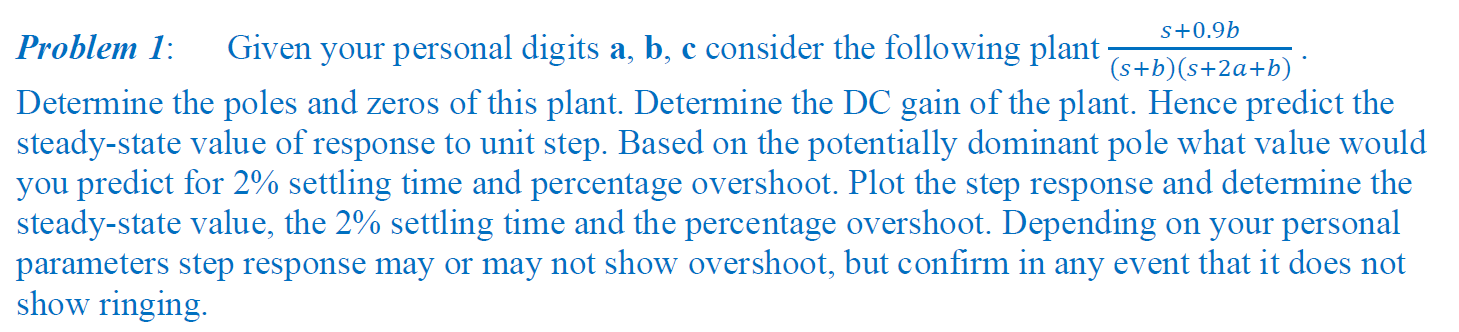
The value of this rational polynomial at *s* = 0 is called the DC gain of the plant.



CODE:

% student number = 22204523

a = 5;

b = 2;

c = 3;

N = [1 0.9\*b];

D = conv([1 b],[1 2\*a+b]);

G = tf(N,D)

DC = polyval(N,0)/polyval(D,0) % to find the DC gain (finding the value of polynomial at s=0

zeros = zero(G)

poles = pole(G)

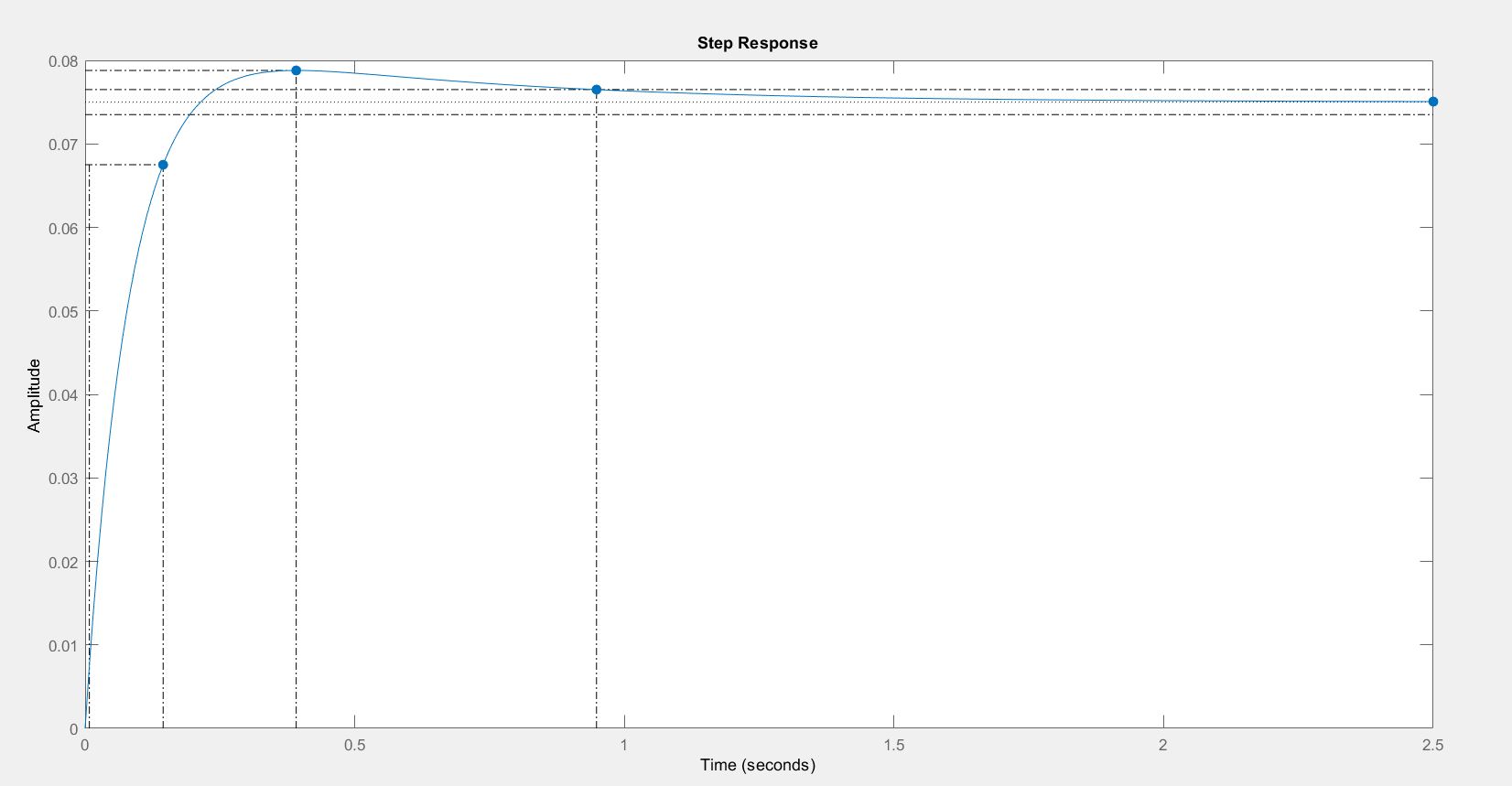
step(G)

stepinfo(G)

RESULTS





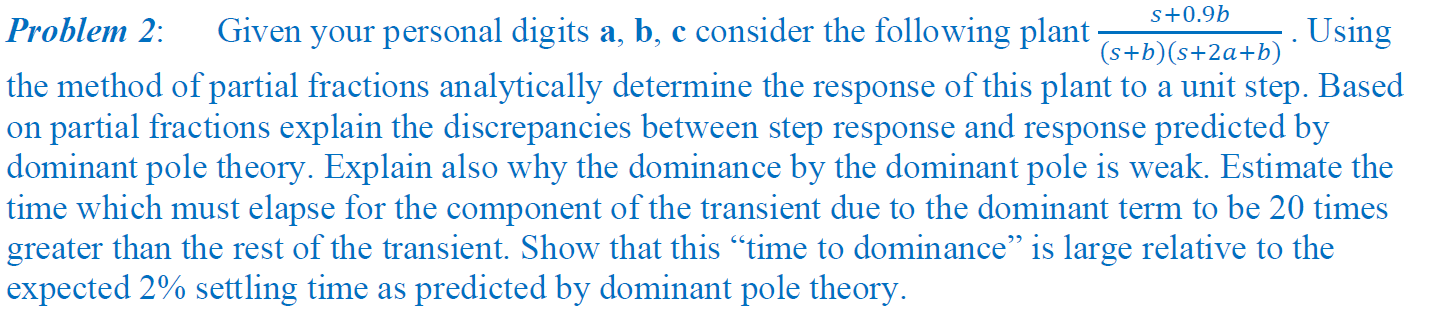




The steady-state value is understandable since the DC gain of the plant as previously noted is 0.5/3 = 0.1667. We see that the dominance of the pole at -1 is being undermined by the presence of the zero, the effect of which has been to ensure that the residue, -0.4167, associated with the dominated pole at -3 is larger in magnitude than the residue, 0.25, associated with the dominant pole at -1 and moreover that the residue 0.25 is positive. We see in lectures that a positive coefficient of would-be dominant real pole in this partial fraction expansion implies that we will see overshoot but no ringing, even though the would-be dominant pole is real. The suppression of the magnitude of coefficient 0.25 relative to magnitude of other coefficients in partial fraction expansion means that the time required for the dominance to really take effect is increased, becoming indeed equal to a good fraction of the transient time. The dominance therefore is weak and this is why the system response does not look like a first order response.

\* SettlingMin: min value of Y once the response has risen

\* SettlingMax: max value of Y once the response has risen



CODE:

% student number = 22204523

a = 5;

b = 2;

c = 3;

N = [1 0.9\*b];

D1 = conv([1 b],[1 2\*a+b]);

G = tf(N,D1)

%after step repsonse we get s multiplied by the denominator

D = [1 14 24 0];

G1 = tf(N,D)

[R P K] = residue(N,D)

RESULT:



After applying partial fraction method we get

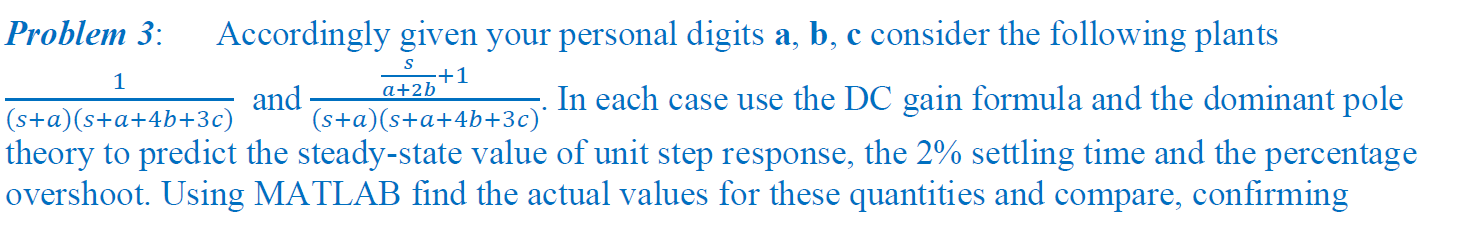
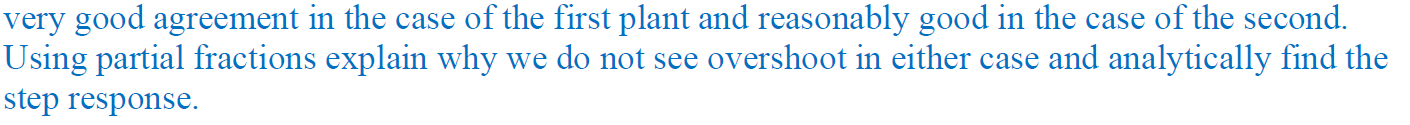
G(s) =

After inverse Laplace transform we get

g(t) = 0.075 - 0.01e-2t + 0.085e-12t

the magnitude of the dominant real pole is less than the magnitude of the dominated real pole that’s why the dominance of the dominance of the pole at -2 is undermined. The suppression of the magnitude of coefficient 0.01 relative to magnitude of other coefficient (0.085) in partial fraction expansion means that the time required for the dominance to really take effect is increased, becoming indeed equal to a good fraction of the transient time. The dominance therefore is weak and this is why the system response does not look like a first order response.

The value of A is positive which means that according to the dominant pole theory we should not have overshoot but still we are getting it because of the dominated term’s amplitude as it is affecting the response more initially

CODE:

% student number = 22204523

a = 5;

b = 2;

c = 3;

N1 = [1];

N2 = [1/9 1];

D = conv([1 a],[1 a+4\*b+3\*c]);

G1 = tf(N1,D)

G2 = tf(N2,D)

DC1 = polyval(N1,0)/polyval(D,0) % to find the DC gain

DC2 = polyval(N2,0)/polyval(D,0) % to find the DC gain

step(G1)

stepinfo(G1)

figure

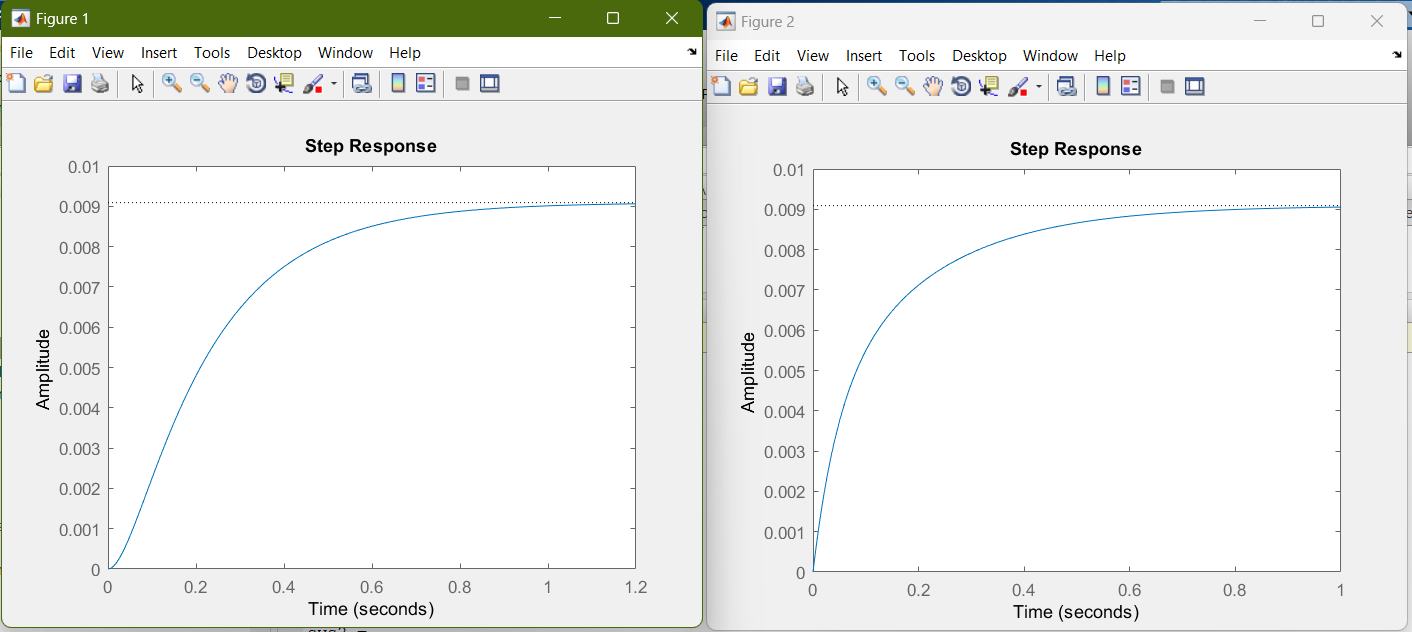
step(G2)

stepinfo(G2)

Results:







System 1 System 2

The ts2% of both the systems was calculated as 4/5 = 0.8 Seconds.

The ts2% calculated for system 1 = 0.8340 Sec (very good agreement with the calculated time).

The ts2% calculated for system 2 = 0.6718 Sec (reasonably good agreement with the calculated time).

After residue function the partial fractions were found as:

For system 1:

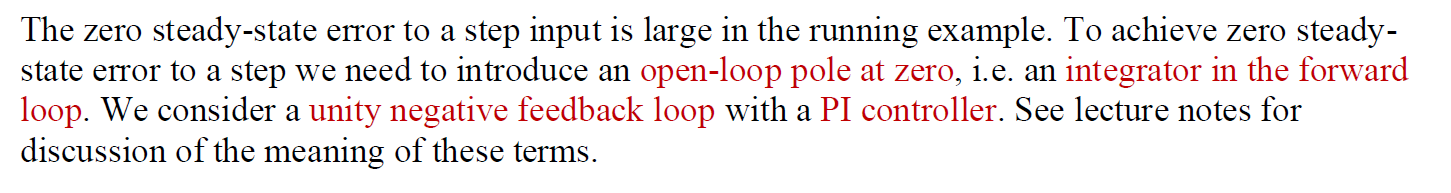
G(s)

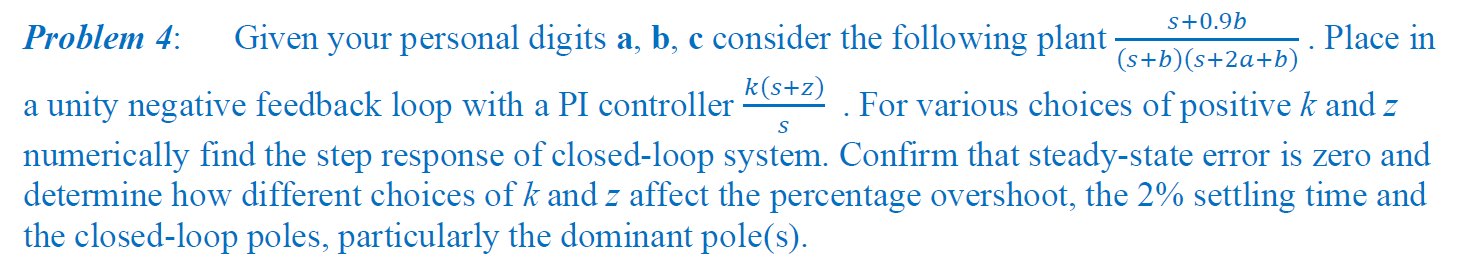
For system 2:

G(s)

The amplitude of the dominant term and dominated term is very small as compared to unity Dc gain value that’s why we don’t see any overshoot in both the systems.

There is some non-linearity in the 1st system because of the -ve dominated term and then it settled and dominance was provided to the dominant pole.





CODE:

kp = 10;

ki = 20;

k = kp

z = ki/kp

N = [1 1.8];

D = conv([1 2],[1 12])

Gp = tf(N,D)

Nc = [1 z];

Np = [1 0];

Gc = tf(k\*Nc,Np)

Go = series(Gp,Gc)

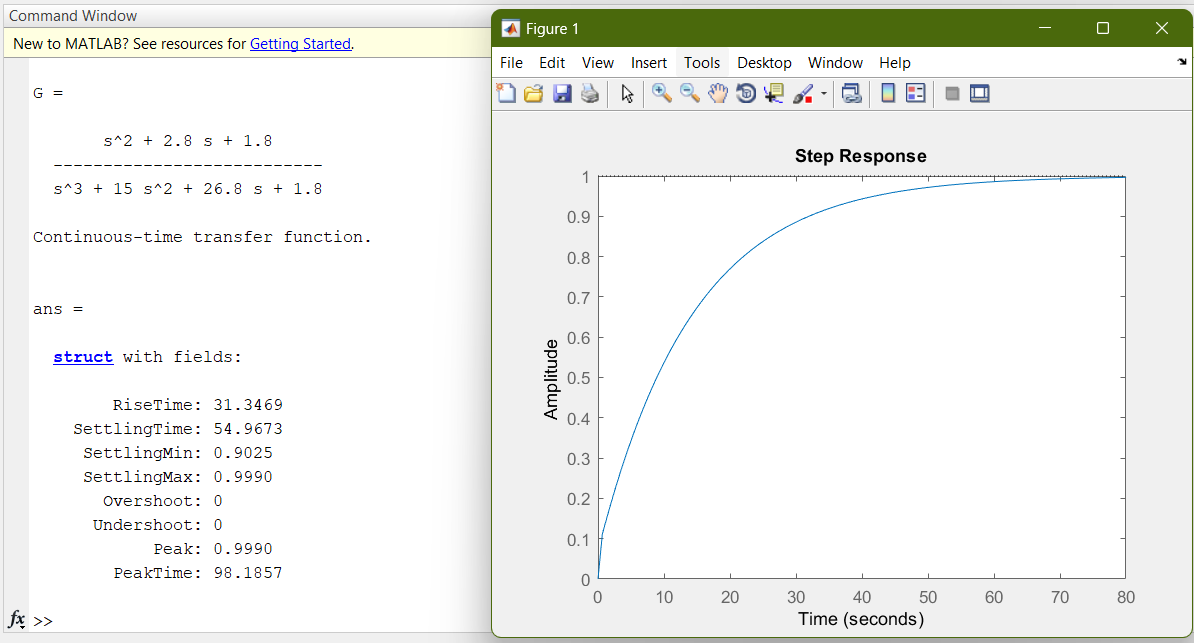
Gi = tf([1],[1]) % unity feedback

G = feedback(Go,Gi)

step(G)

stepinfo(G)

RESULT



At kp = 1 and ki = 1 the response was free from overshoots it was very slow

The kp = 10 and ki = 20 was taken to make the system faster

Here k = kp = 10 and z = ki/kp = 2

