ECE 3793
Matlab Project 3 Solution

Spring 2017 Dr. Havlicek

1. (a) In text problem 9.22(d), we are given

$$X(s) = \frac{s+2}{s^2+7s+12} \qquad -4 < \mathrm{Re}\{s\} < -3.$$

The following Matlab statements determine the partial fraction expansion for $X(s)$:

```
>> numer = [1 2];
>> denom = [1 7 12];
>> [r p k] = residue(numer,denom)

r =

     2
    -1


p =

    -4
    -3


k =

     []
```

From the `[r p k]` values returned by `residue`, we can write down the partial fraction expansion as follows:

$$X(s) = \frac{2}{s+4} - \frac{1}{s+3}.$$

Now, for the overall ROC to be $-4 < \mathrm{Re}\{s\} < -3$, one of these terms must have a ROC of $\mathrm{Re}\{s\} > -4$ and the other one must have a ROC of $\mathrm{Re}\{s\} < -3$. So, the ROC's for the individual terms must be

$$X(s) = \underbrace{\frac{2}{s+4}}_{\mathrm{Re}\{s\}>-4} - \underbrace{\frac{1}{s+3}}_{\mathrm{Re}\{s\}<-3}.$$

From Table 9.2, it then follows immediately that

$$x(t) = 2e^{-4t}u(t) + e^{-3t}u(-t).$$

(b) In text problem 9.10(c), we are given

$$H_3(s) = \frac{s^2}{s^2 + 2s + 1}, \qquad \mathrm{Re}\{s\} > -1.$$

The following Matlab statements determine the partial fraction expansion for $H_3(s)$:

```
>> numer = [1 0 0];
>> denom = [1 2 1];
>> [r p k] = residue(numer,denom)

r =

    -2
     1


p =

    -1
    -1


k =

     1
```

From the `[r p k]` values returned by `residue`, we can write down the partial fraction expansion as follows:

$$H_3(s) = 1 - \frac{2}{s+1} + \frac{1}{(s+1)^2}.$$

For the overall ROC to be $\mathrm{Re}\{s\} > -1$, the ROC's for the individual terms must be

$$H_3(s) = \underbrace{1}_{\text{all } s} - \underbrace{\frac{2}{s+1}}_{\mathrm{Re}\{s\}>-1} - \underbrace{\frac{1}{(s+1)^2}}_{\mathrm{Re}\{s\}>-1}.$$

From Table 9.2, it then follows immediately that

$$h_3(t) = \delta(t) - 2e^{-t}u(t) + te^{-t}u(t).$$

2. (a) Matlab code:

```
%-----------------------------------------------------
% P2a
%
% Use the transfer function
%
%          2(s+1)(s-2)
% H(s) = --------------- , Re{s} > 1
%        (s+2)(s+3)(s-1)
%
% to illustrate the Matlab functions tf, zpk, zero,
% pole, tfdata, isstable, printsys, and pzmap.
%
clear;
z = [-1 2];            % zeros (roots of numerator)
p = [-2 -3 1];         % poles (roots of denominator)
g = 2;                 % transfer function gain
H1 = zpk(z,p,g)        % zero-pole-gain system model
H2 = tf(H1)            % convert to transfer function
                       % system model
%
% Get the numerator and denominator polynomial coefficients
%
[numer,denom] = tfdata(H1,'v')
%
% Do it again with the other model
%
[numer,denom] = tfdata(H2,'v')
%
% Compute a transfer function system model
% from the numer and denom coefficients
%
H3 = tf(numer,denom)  % H3 is the same as H1
%
% Convert to zero-pole-gain system model
%
H4 = zpk(H3)          % H4 is the same as H1
%
% Use the numer and denom coefficients to
% print the transfer function to the
% Matlab command window
%
printsys(numer,denom)
%
```

```
% Use the pole function to get the poles
% from system models H1 and H2
%
p = pole(H1)
p = pole(H2)          % same result as the line above
%
% Use the zero function to get the zeros
% from system models H1 and H2
%
z = zero(H1)
z = zero(H2)          % same result as the line above
%
% Use pzmap to get the poles and zeros
%
[p z] = pzmap(H1)
[p z] = pzmap(H2)     % same result as the line above
%
% Generate a pole-zero plot in three different ways
%
figure(1); pzmap(H1);
figure(2); pzmap(H2);
figure(3); pzmap(numer,denom);
%
% Use isstable to show that the system is unstable
%
isstable(H1)
%
% Use residue to compute a partial fraction
% expansion.  Notice that we don't
% have to multiply out the denominator by hand.
%
[r p k] = residue(numer,denom)
%
% Let's see what variables we have:
%
whos
```

Matlab Command Window Output:

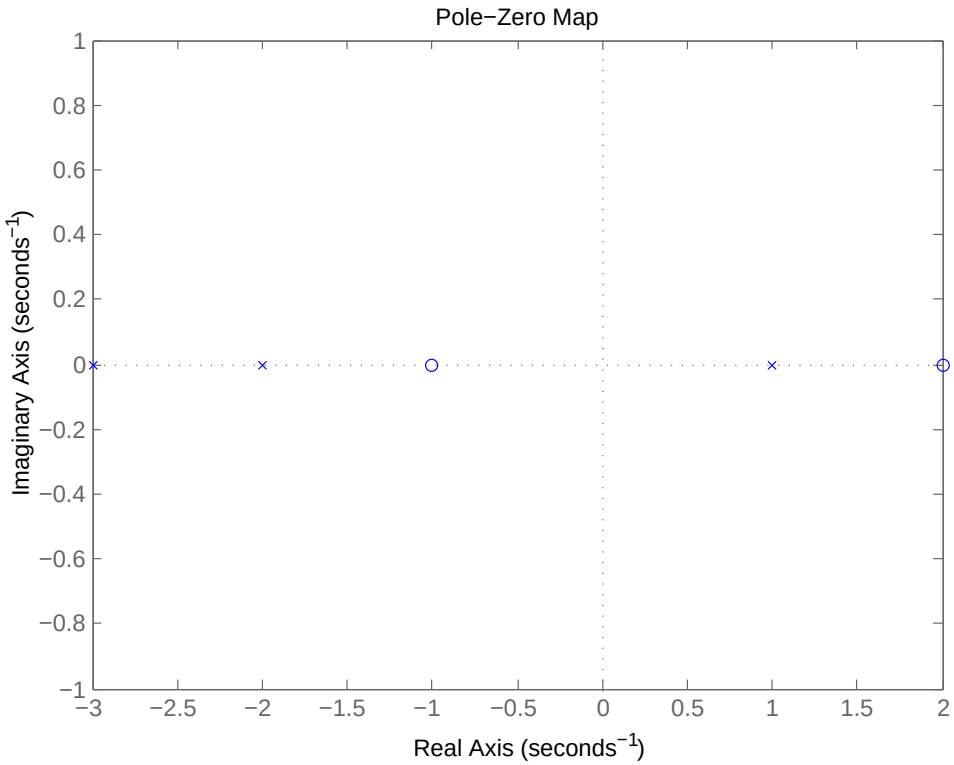
```
>> P2a

H1 =

    2 (s+1) (s-2)
  -----------------
  (s+2) (s+3) (s-1)

Continuous-time zero/pole/gain model.


H2 =

    2 s^2 - 2 s - 4
  -------------------
  s^3 + 4 s^2 + s - 6

Continuous-time transfer function.


numer =

     0     2    -2    -4


denom =

     1     4     1    -6


numer =

     0     2    -2    -4


denom =

     1     4     1    -6
```

```
H3 =

    2 s^2 - 2 s - 4
  -------------------
  s^3 + 4 s^2 + s - 6

Continuous-time transfer function.


H4 =

    2 (s-2) (s+1)
  -----------------
  (s+3) (s+2) (s-1)

Continuous-time zero/pole/gain model.


num/den =

      2 s^2 - 2 s - 4
   --------------------
   s^3 + 4 s^2 +  s - 6

p =

    -2
    -3
     1


p =

   -3.0000
   -2.0000
    1.0000


z =

    -1
     2
```

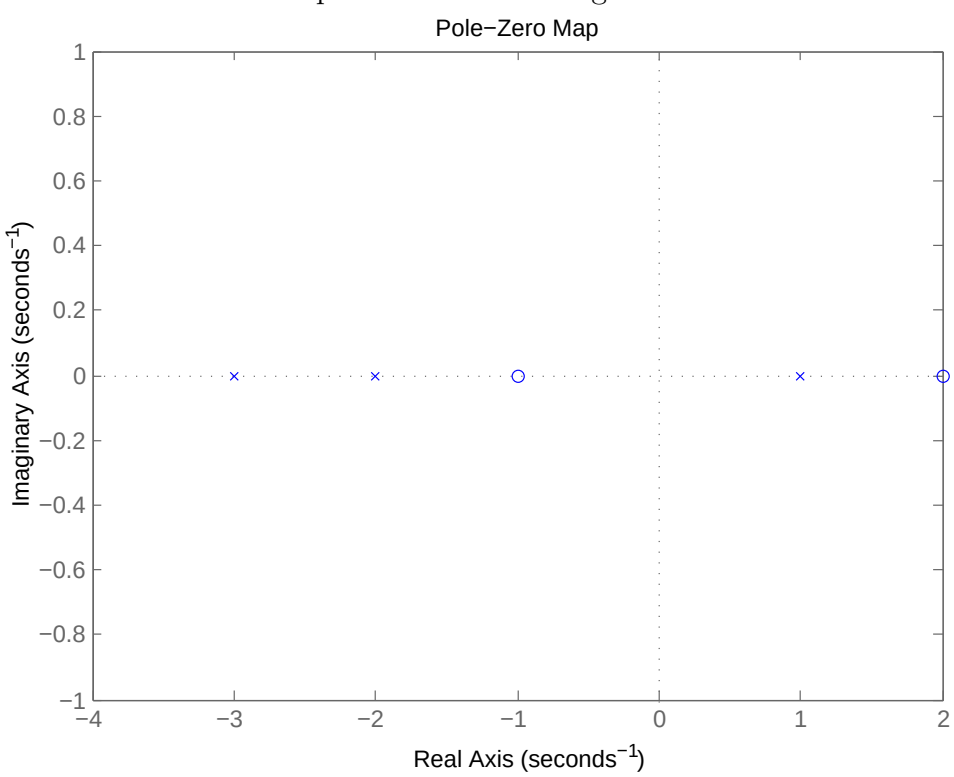
```
z =

     2
    -1


p =

    -2
    -3
     1


z =

    -1
     2


p =

   -3.0000
   -2.0000
    1.0000


z =

     2
    -1


ans =

     0


r =

    5.0000
   -2.6667
   -0.3333
```

```
p =

   -3.0000
   -2.0000
    1.0000


k =

     []

  Name       Size            Bytes  Class      Attributes

  H1         1x1              1065  zpk
  H2         1x1              1081  tf
  H3         1x1              1081  tf
  H4         1x1              1065  zpk
  ans        1x1                 1  logical
  denom      1x4                32  double
  g          1x1                 8  double
  k          0x0                 0  double
  numer      1x4                32  double
  p          3x1                24  double
  r          3x1                24  double
  z          2x1                16  double
```

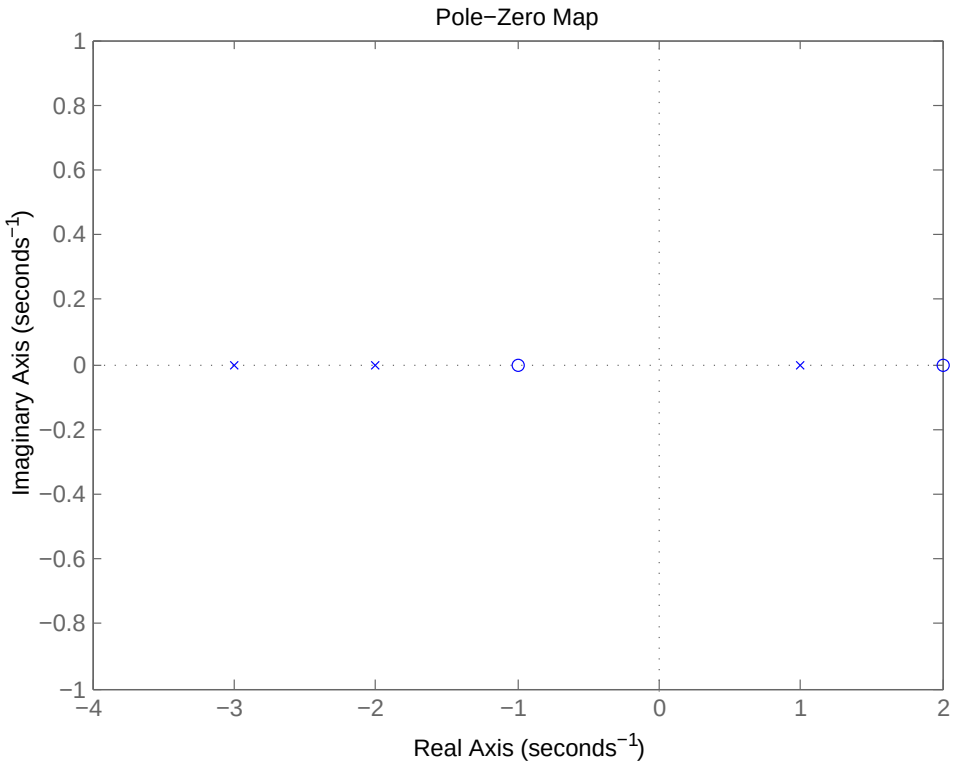
Pole-zero plot from Matlab Figure Window 1:



Pole-zero plot from Matlab Figure Window 2:



Pole-zero plot from Matlab Figure Window 3:



(b) From the `[r p k]` values returned by `residue`, we can write down the partial fraction expansion as follows:

$$H(s) = \frac{5}{s+3} - \frac{\frac{8}{3}}{s+2} - \frac{\frac{1}{3}}{s-1}.$$

For the overall ROC to be $\mathrm{Re}\{s\} > 1$, all three individual terms must have right-sided ROC's. So the individual ROC's must be

$$H(s) = \underbrace{\frac{5}{s+3}}_{\mathrm{Re}\{s\}>-3} - \underbrace{\frac{\frac{8}{3}}{s+2}}_{\mathrm{Re}\{s\}>-2} - \underbrace{\frac{\frac{1}{3}}{s-1}}_{\mathrm{Re}\{s\}>1}.$$

From Table 9.2, it then follows immediately that

$$h(t) = 5e^{-3t}u(t) - \frac{8}{3}e^{-2t}u(t) - \frac{1}{3}e^{t}u(t).$$

(c) The given input-output equation is

$$y'''(t) + 6y''(t) + 11y'(t) + 6y(t) = x(t).$$

Taking the Laplace transform on both sides, we obtain

$$\begin{aligned} s^3Y(s) + 6s^2Y(s) + 11sY(s) + 6Y(s) &= X(s) \\ \left[s^3 + 6s^2 + 11s + 6\right] Y(s) &= X(s), \end{aligned}$$

from which it follows that

$$H(s) = \frac{Y(s)}{X(s)} = \frac{1}{s^3 + 6s^2 + 11s + 6}.$$

We would have to factor the denominator to determine the ROC. However, since the problem tells us to assume that the system is causal, we know that the ROC must be the right half-plane to the right of the rightmost pole.

Matlab code:

```
%-----------------------------------------------------
% P2c
%
% For the system in text problem 9.40,
%
% - assume the system is causal
% - find the transfer function H(s) by hand
% - use tf to compute a transfer function system
%     model
% - use zpk to convert the transfer function
%     system model to a zero-pole-gain system
%     model
% - use pzmap to produce a pole-zero plot
% - use pole to find the poles
% - use isstable to determine if the system is
%     stable
% - use reside to compute a partial fraction
%     expansion and invert to find h(t)
%
clear;
numer = 1;             % numerator coefficients
denom = [1 6 11 6];    % denominator coefficients
H1 = tf(numer,denom)   % transfer function model
H2 = zpk(H1)           % convert to zero-pole-gain
pzmap(H2);             % produce pole-zero plot
p = pole(H1)           % get the poles
%
% Check for stability
%
if (isstable(H1))
  disp('The system IS stable.');
else
  disp('The system is NOT stable.');
end
%
```

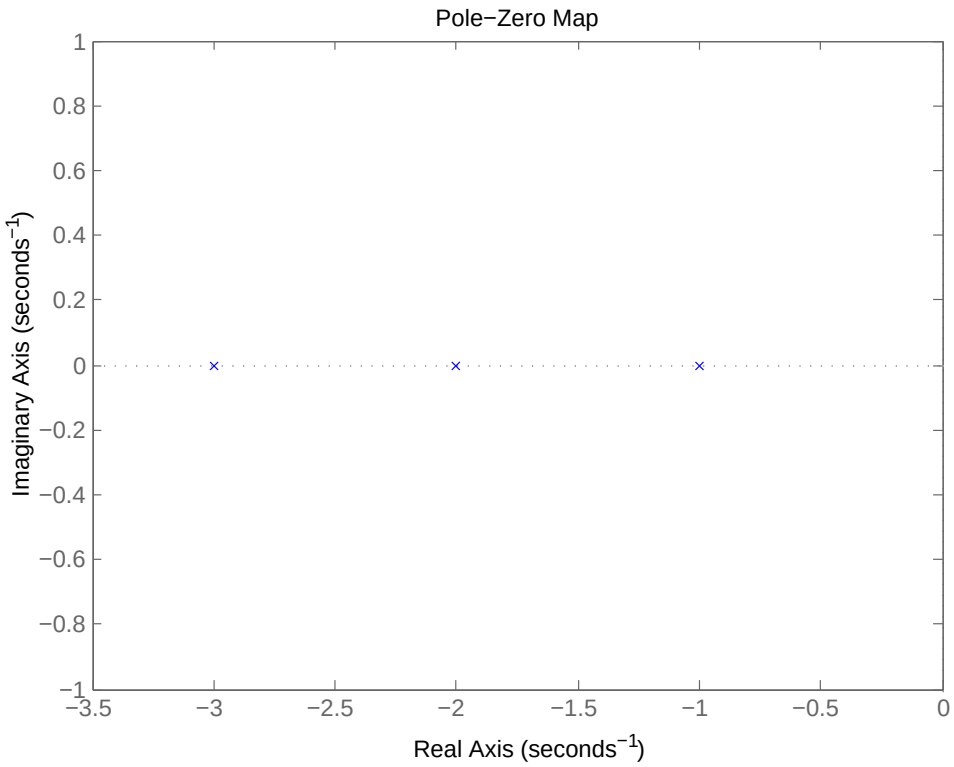
```
% Partial Fraction Expansion
%
[r p k] = residue(numer,denom)
```

Matlab Command Window Output:

```
>> P2c

H1 =

            1
  ----------------------
  s^3 + 6 s^2 + 11 s + 6

Continuous-time transfer function.


H2 =

          1
  -----------------
  (s+3) (s+2) (s+1)

Continuous-time zero/pole/gain model.


p =

   -3.0000
   -2.0000
   -1.0000

The system IS stable.

r =

    0.5000
   -1.0000
    0.5000


p =

   -3.0000
   -2.0000
```

```
   -1.0000


k =

     []
```

Pole-zero plot from Matlab Figure Window:



From the `[r p k]` values returned by `residue`, we can write down the partial fraction expansion as follows:

$$H(s) = \frac{\frac{1}{2}}{s+3} - \frac{1}{s+2} + \frac{\frac{1}{2}}{s+1}.$$

Since the system is assumed causal, the ROC must be $\mathrm{Re}\{s\} > -1$. Therefore, the ROC's for the individual terms must be

$$H(s) = \underbrace{\frac{\frac{1}{2}}{s+3}}_{\mathrm{Re}\{s\}>-3} - \underbrace{\frac{1}{s+2}}_{\mathrm{Re}\{s\}>-2} + \underbrace{\frac{\frac{1}{2}}{s+1}}_{\mathrm{Re}\{s\}>-1}.$$

From Table 9.2, it then follows immediately that

$$h(t) = \frac{1}{2}e^{-3t}u(t) - e^{-2t}u(t) + \frac{1}{2}e^{-t}u(t).$$

3. (a) Matlab code:

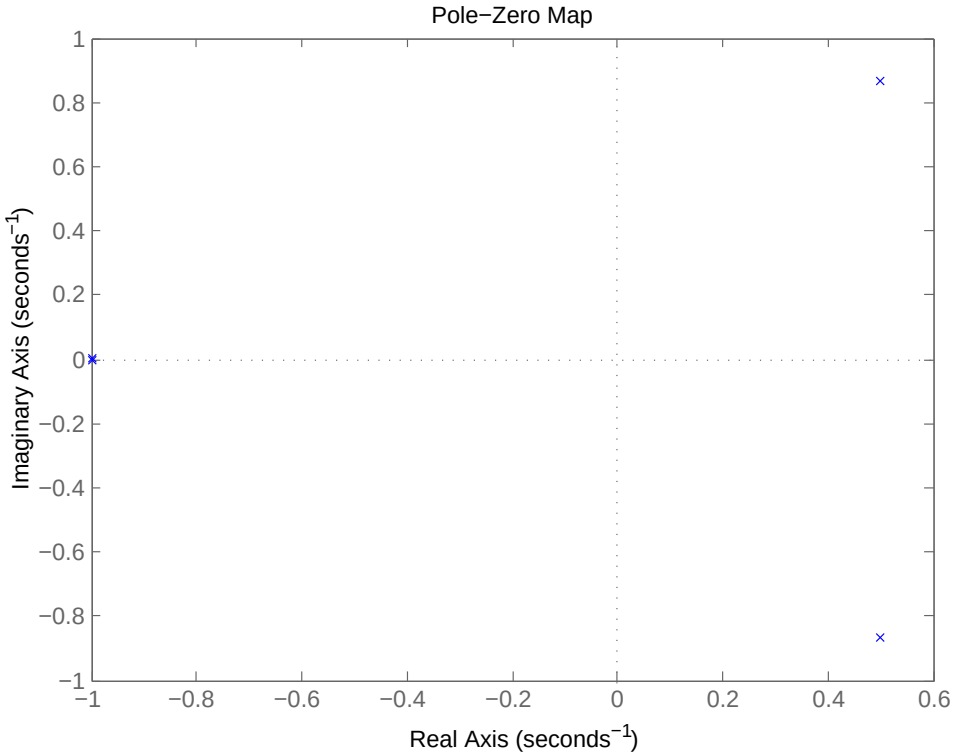
```
%-----------------------------------------------------
% P3a
%
% Use zpk, pole, zero, and pzmap to find the poles
% and zeros and generate a pole-zero plot for the
% Laplace transform H(s) in text problem 9.38.
%
clear;
s = zpk('s');
H = 1 / ((s^2 - s + 1)*(s^2 + 2*s + 1))
H2 = tf(H)
p = pole(H)
z = zero(H)
pzmap(H);
```

Matlab Command Window Output:

```
>> P3a

H =

            1
  ---------------------
  (s+1)^2 (s^2 - s + 1)

Continuous-time zero/pole/gain model.


H2 =

                  1
  ---------------------------------
  s^4 + s^3 - 2.362e-16 s^2 + s + 1

Continuous-time transfer function.


p =

   0.5000 + 0.8660i
   0.5000 - 0.8660i
  -1.0000 + 0.0000i
  -1.0000 - 0.0000i
```

```
z =

   Empty matrix: 0-by-1
```

Pole-zero plot from Matlab Figure Window:



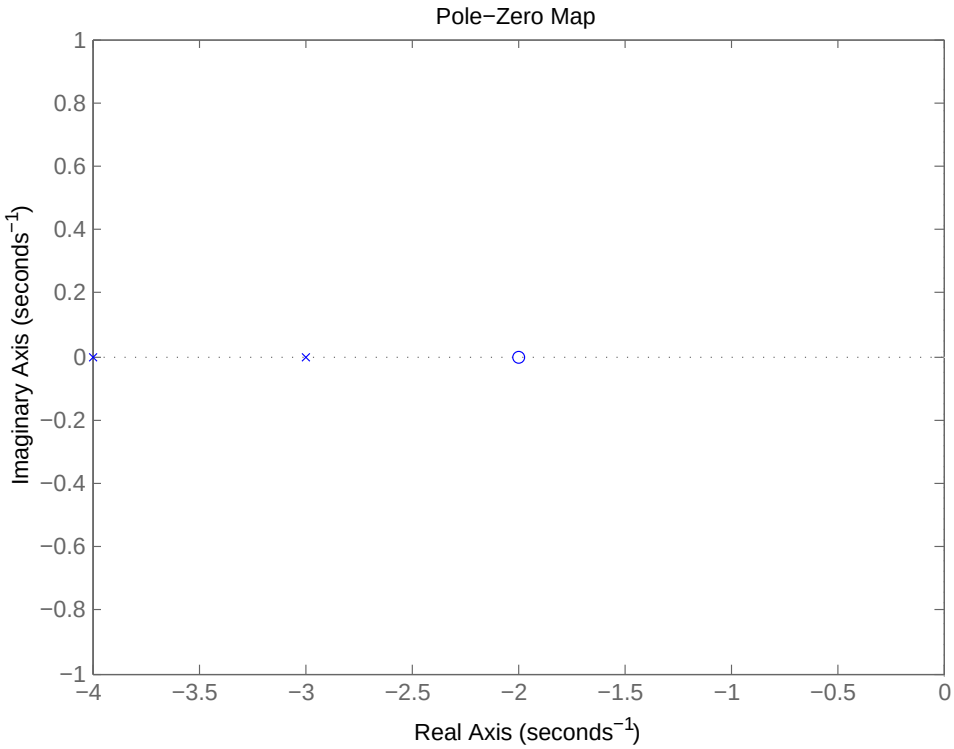
(b) Matlab code:

```
%-----------------------------------------------------
% P3b
%
% - Use tf to generate a pole-zero plot for the
%   Laplace transform X(s) in text problem 9.9.
% = Use tfdata to get the numerator and denominator
%   polynomial coefficient vectors.
% - Use residue to compute the partial fraction
%   expansion.
%
clear;
s = tf('s');
X = 2*(s+2)/(s^2 + 7*s + 12)
X2 = zpk(X)
pzmap(X);
[numer,denom] = tfdata(X,'v');
[r p k] = residue(numer,denom)
```

Matlab Command Window Output:

```
>> P3b

X =

     2 s + 4
  --------------
  s^2 + 7 s + 12

Continuous-time transfer function.


X2 =

    2 (s+2)
  -----------
  (s+4) (s+3)

Continuous-time zero/pole/gain model.


r =

     4
    -2


p =

    -4
    -3


k =

     []
```

Pole-zero plot from Matlab Figure Window:



(c) Matlab code:

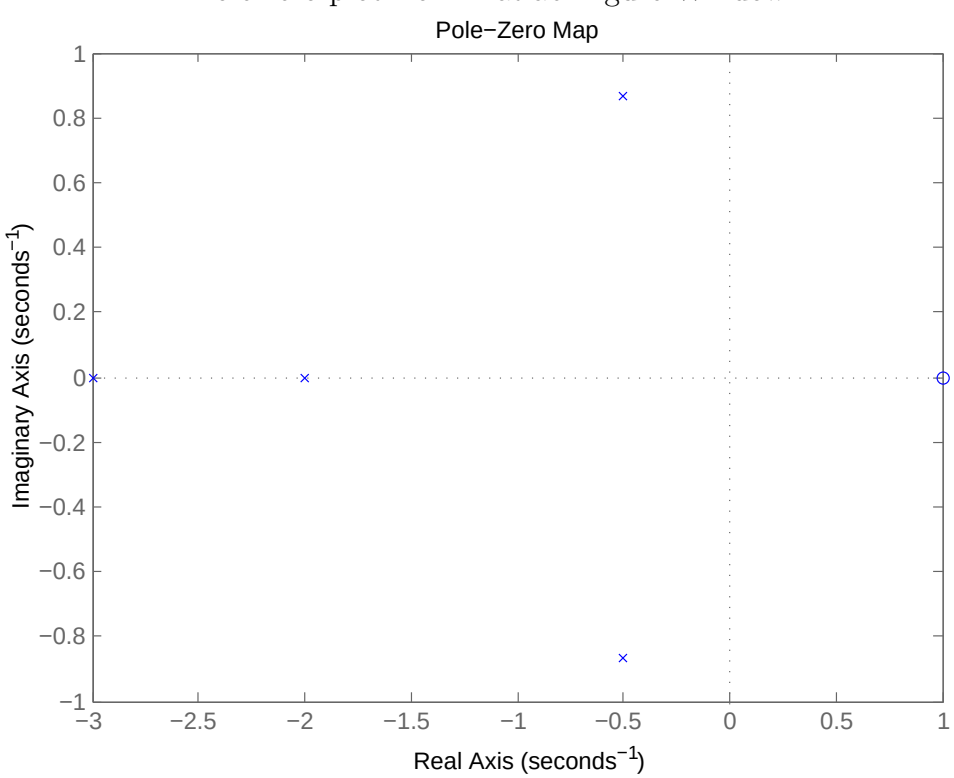
```
%-----------------------------------------------------
% P3c
%
% Generate a pole-zero plot and use it work text
% problem 9.7.
%
clear;
s = zpk('s');
H = (s-1) / ((s+2)*(s+3)*(s^2 + s + 1))
H2 = tf(H)
p = pole(H)
pzmap(H);
```

Matlab Command Window Output:

```
>> P3c

H =

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

```
Continuous-time zero/pole/gain model.


H2 =

                s - 1
  -------------------------------
  s^4 + 6 s^3 + 12 s^2 + 11 s + 6

Continuous-time transfer function.


p =

  -2.0000 + 0.0000i
  -3.0000 + 0.0000i
  -0.5000 + 0.8660i
  -0.5000 - 0.8660i
```

Pole-zero plot from Matlab Figure Window:



From the values of the poles returned by `pole` and from the pole-zero plot, we see that there are four possible ROC's:

$$\begin{aligned} \mathrm{Re}\{s\} &< -3, \\ -3 < \mathrm{Re}\{s\} &< -2, \\ -2 < \mathrm{Re}\{s\} &< -\tfrac{1}{2}, \\ \mathrm{Re}\{s\} &> -\tfrac{1}{2}. \end{aligned}$$

Therefore, there are four different signals that have a Laplace transform that may be expressed as $X(s)$.

4. (a) Matlab code:

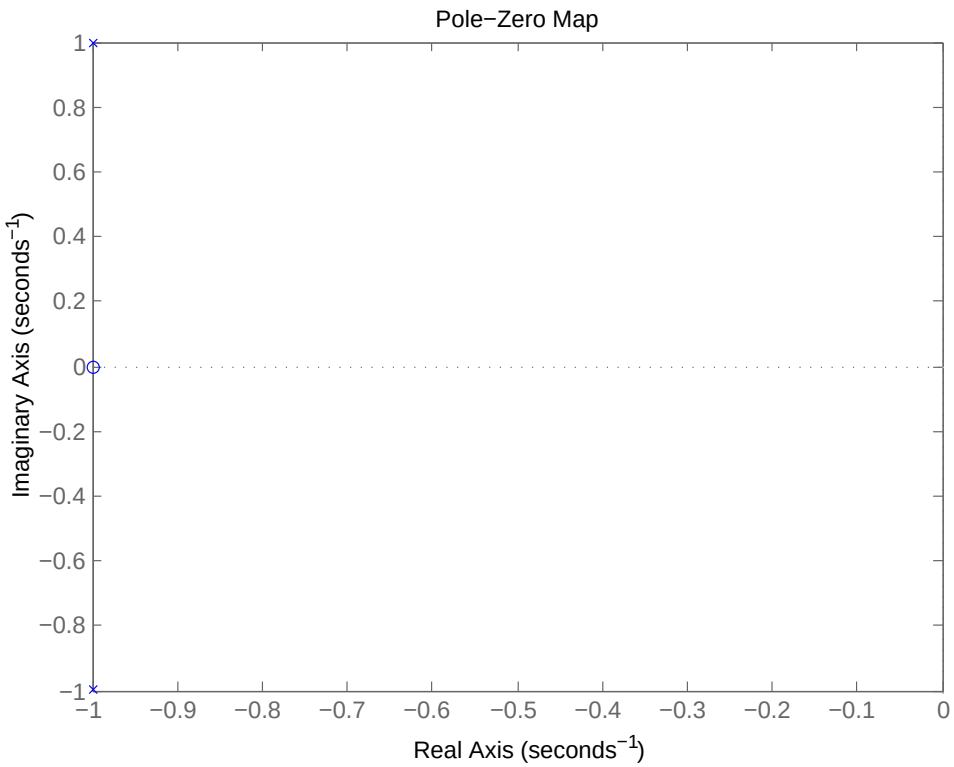
```
%-------------------------------------------------------------
% P4a
%
% For the transfer function H(s) in text problem 9.33,
% - make a pole-zero plot
% - plot the impulse response
% - plot the step response
% - plot the frequency response
%
clear;
s = tf('s');
H = (s+1)/(s^2 + 2*s + 2)
[p z] = pzmap(H)    % print poles and zeros to the cmd window
figure(1); pzmap(H);
figure(2); impulse(H);
figure(3); step(H);
figure(4); bode(H);
```

Matlab Command Window Output:

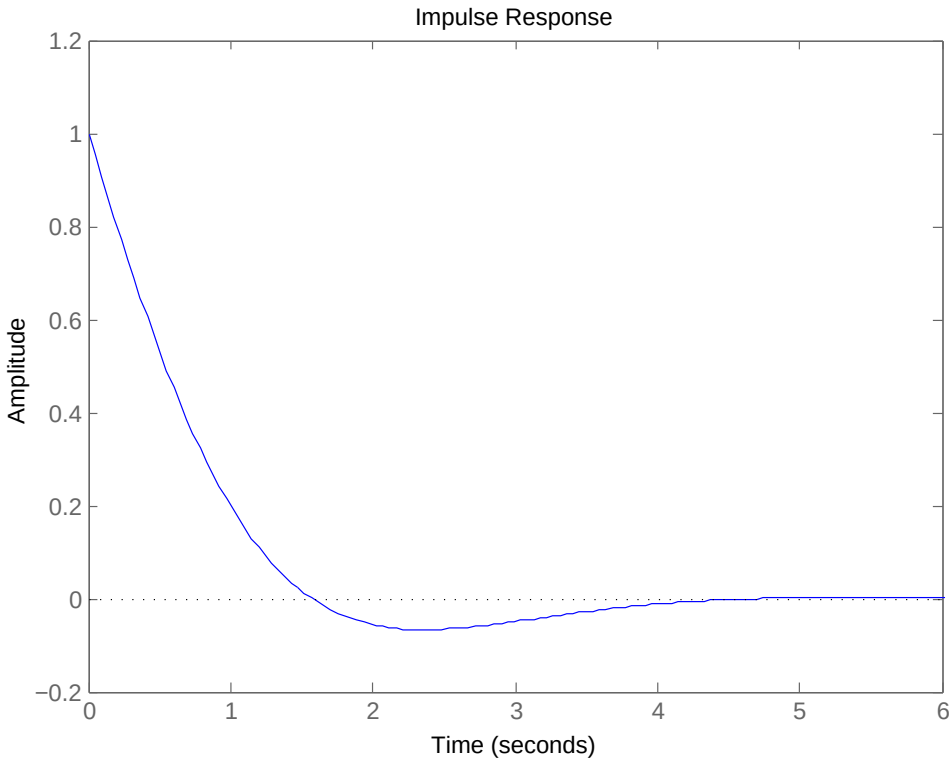
```
>> P4a

H =

      s + 1
  -------------
  s^2 + 2 s + 2

Continuous-time transfer function.


p =

  -1.0000 + 1.0000i
```

```
  -1.0000 - 1.0000i


z =

    -1
```

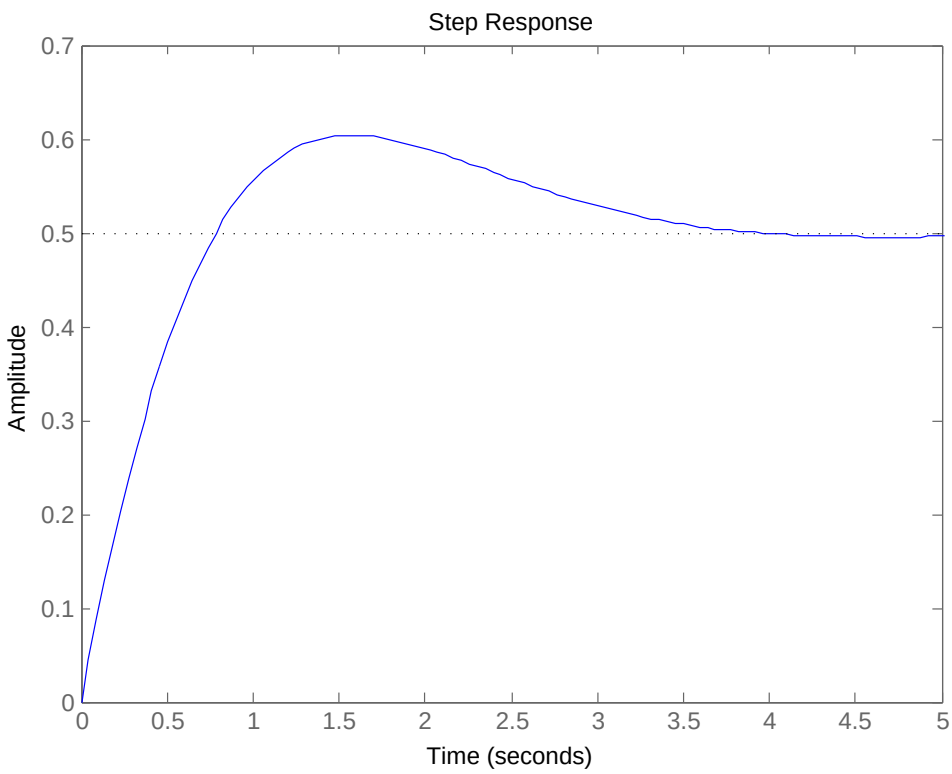
Pole-zero plot from Matlab Figure Window 1:



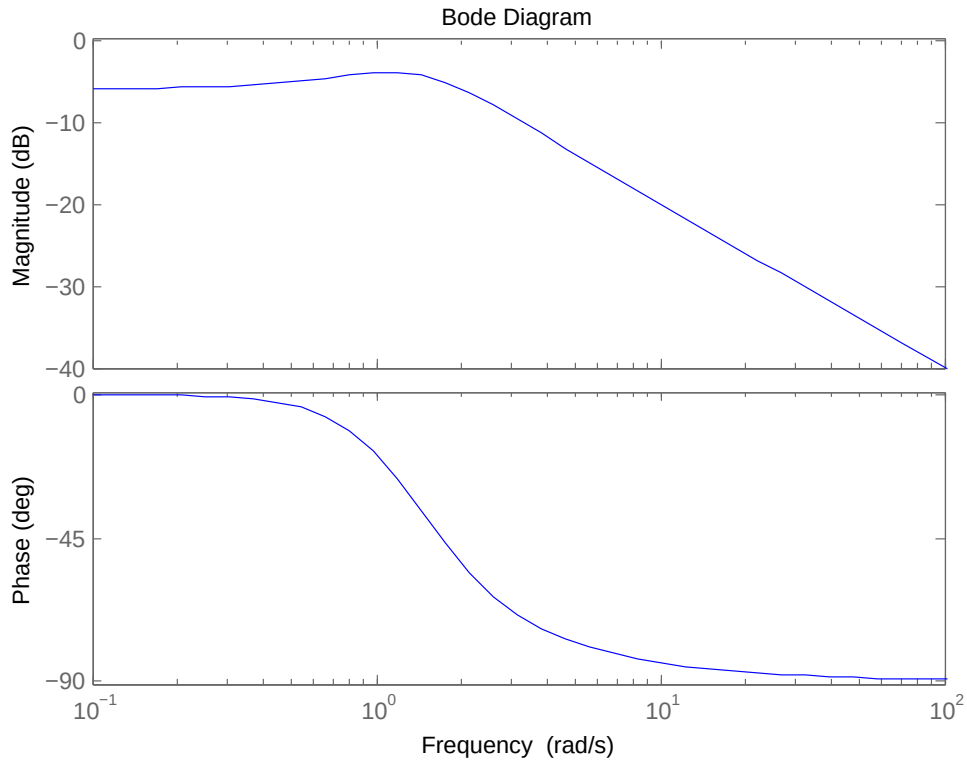
Plot of impulse response from Matlab Figure Window 2:



Plot of step response from Matlab Figure Window 3:



Plot of frequency response from Matlab Figure Window 4:



(b) Matlab code:

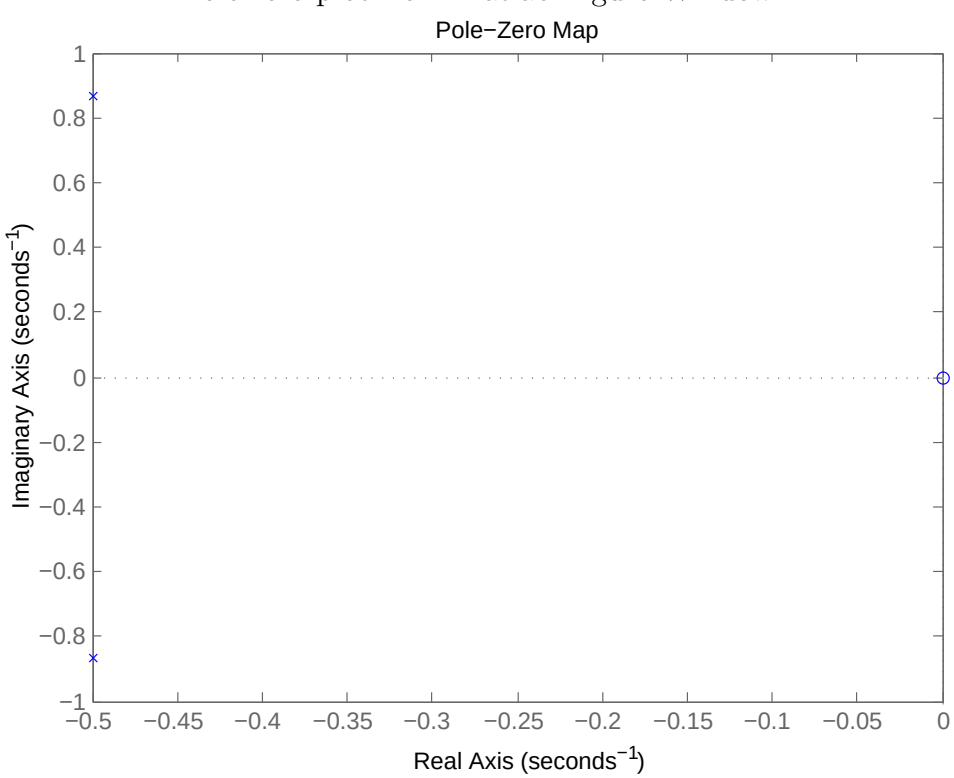
```
%-------------------------------------------------------------
% P4b
%
% For the transfer function H_2(s) in text problem 9.10(b),
% - make a pole-zero plot
% - plot the impulse response
% - plot the step response
% - plot the frequency response
%
clear;
s = tf('s');
H = s/(s^2 + s + 1)
[p z] = pzmap(H)
figure(1); pzmap(H);
figure(2); impulse(H);
figure(3); step(H);
figure(4); bode(H);
```

Matlab Command Window Output:

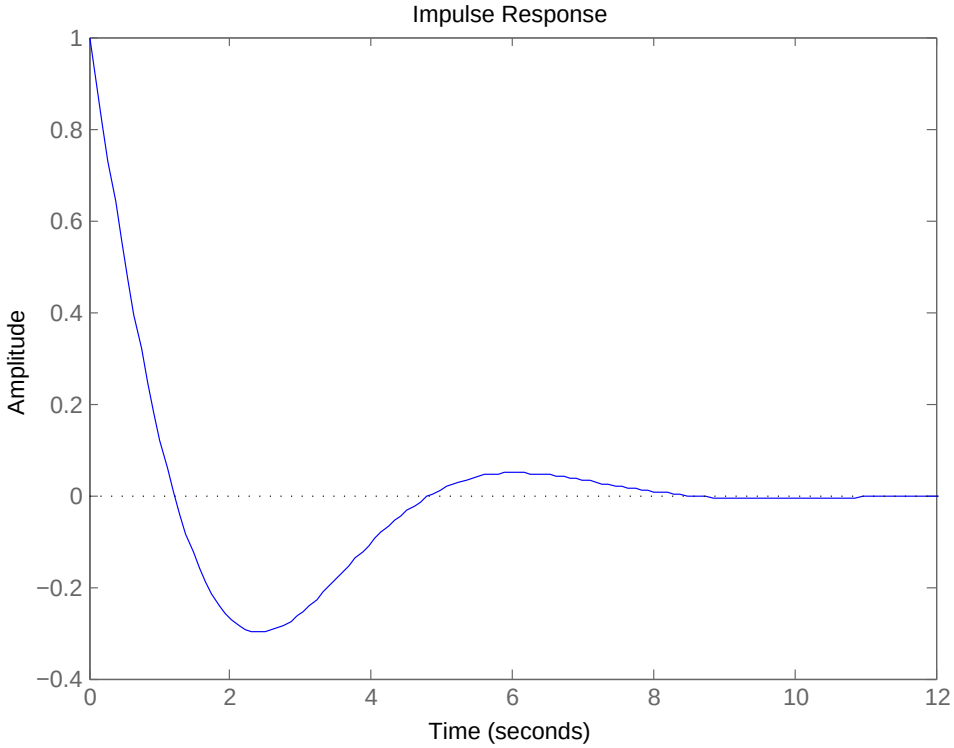
```
>> P4b
```

```
H =

      s
  -----------
  s^2 + s + 1

Continuous-time transfer function.


p =

  -0.5000 + 0.8660i
  -0.5000 - 0.8660i


z =

     0
```

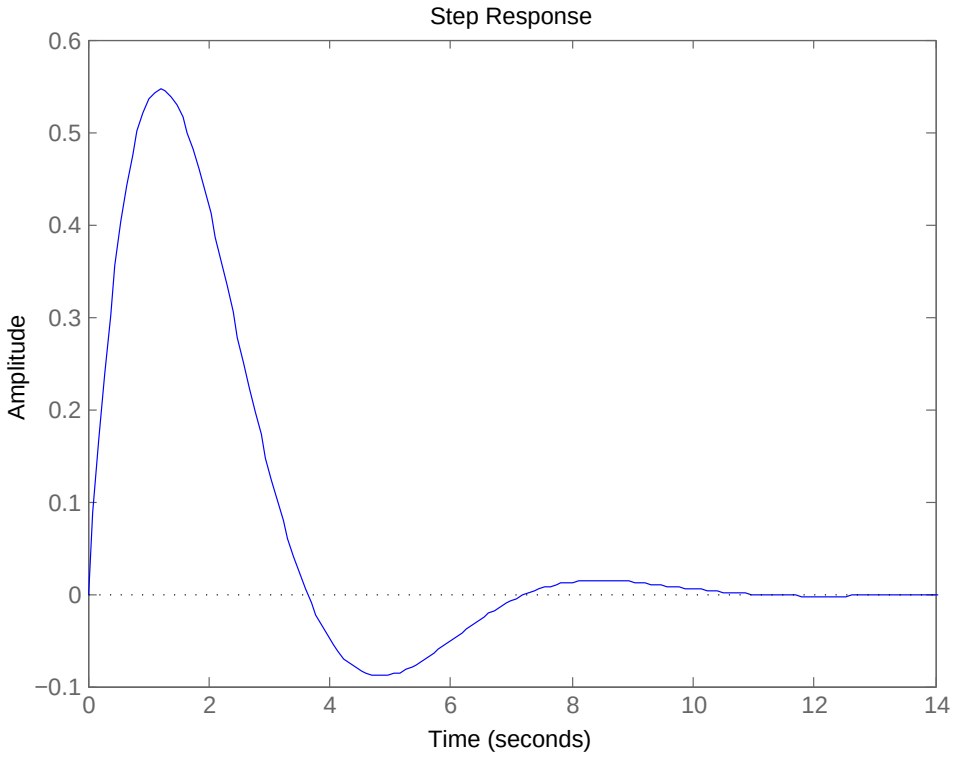
Pole-zero plot from Matlab Figure Window 1:



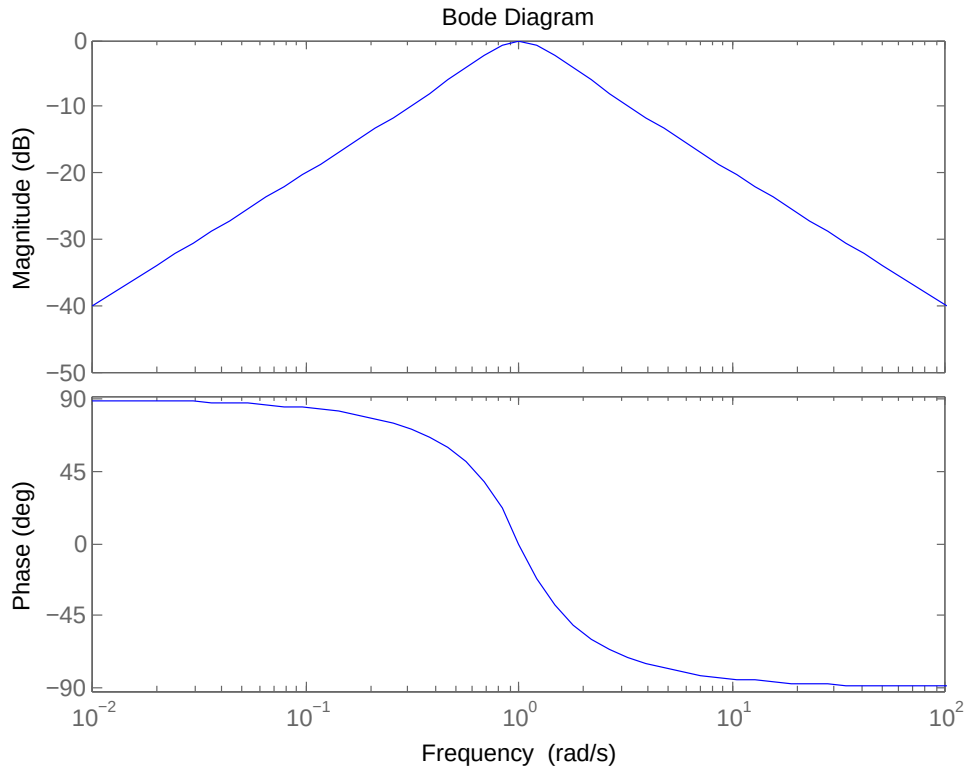
Plot of impulse response from Matlab Figure Window 2:



Plot of step response from Matlab Figure Window 3:



Plot of frequency response from Matlab Figure Window 4:



5. (a) Matlab code:

```
%-----------------------------------------------------------------
% P5a
%
% Work text problem 9.17.
%   - the system block diagram is given in Fig. P9.17 on p. 724.
%   - The figure involves four LTI system blocks.  They will
%       be called H1 through H4, from top to bottom.
%   - The overall transfer function will be called H.
%   - Note: because the feedback paths enter the summing nodes
%       without a "-" sign, we will have to use the "+1" form
%       of the Matlab feedback command.
%   - The last statement of this code will print out the
%       final transfer function H(s) to the Matlab command
%       window.
%       - From this, the differential equation relating the input
%         x(t) and output y(t) can simply be written down.
%
clear;
s = tf('s');
H1 = 2/s;
numerH2 = -4;  % vector syntax not needed because it is a scalar
```

```
denomH2 =  1;  % but to call tf we do need a denominator
H2 = tf(numerH2,denomH2);
H3 = tf(1/s);
numerH4 = -2;
denomH4 =  1;
H4 = tf(numerH4,denomH4);

H = parallel(feedback(H1,H2,+1),feedback(H3,H4,+1))
```

Matlab Command Window Output:

```
>> P5a

H =

     3 s + 12
  ---------------
  s^2 + 10 s + 16

Continuous-time transfer function.
```

From the command window output of the last Matlab statement, we have that

$$H(s) = \frac{3s + 12}{s^2 + 10s + 16} = \frac{Y(s)}{X(s)}.$$

Cross multiplying, we obtain

$$s^2Y(s) + 10sY(s) + 16Y(s) = 3sX(s) + 12X(s).$$

The differential equation relating the input $x(t)$ to the output $y(t)$ is then obtained by applying the inverse Laplace transform to both sides:

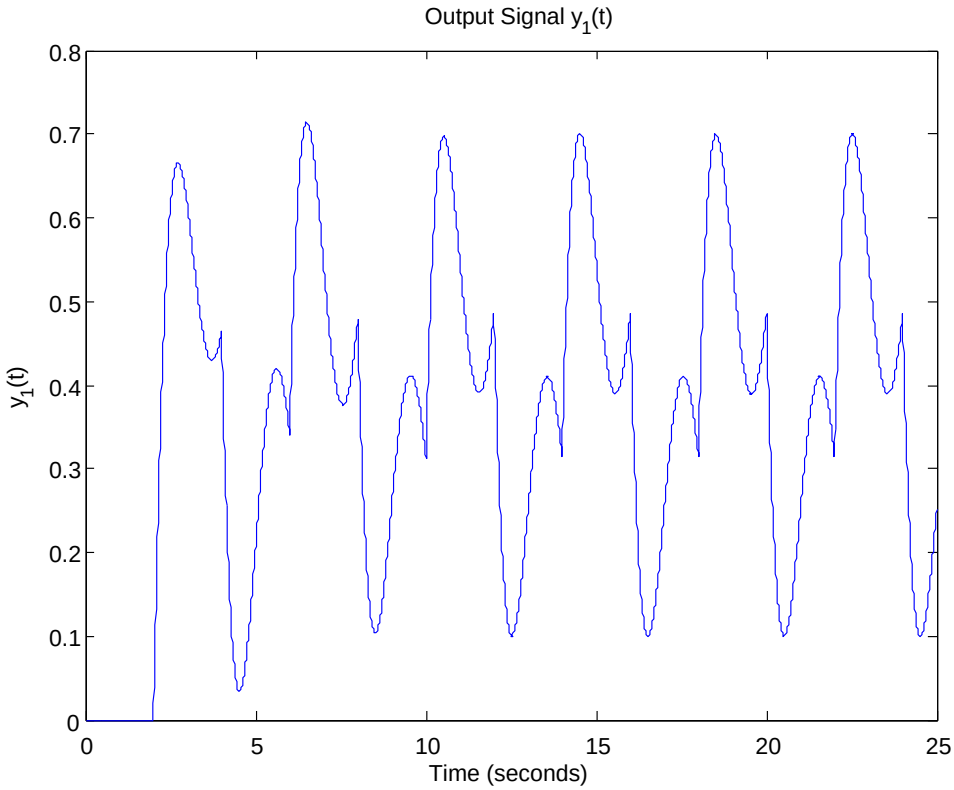
$$y''(t) + 10y'(t) + 16y(t) = 3x'(t) + 12x(t).$$

6. (a) Matlab code:

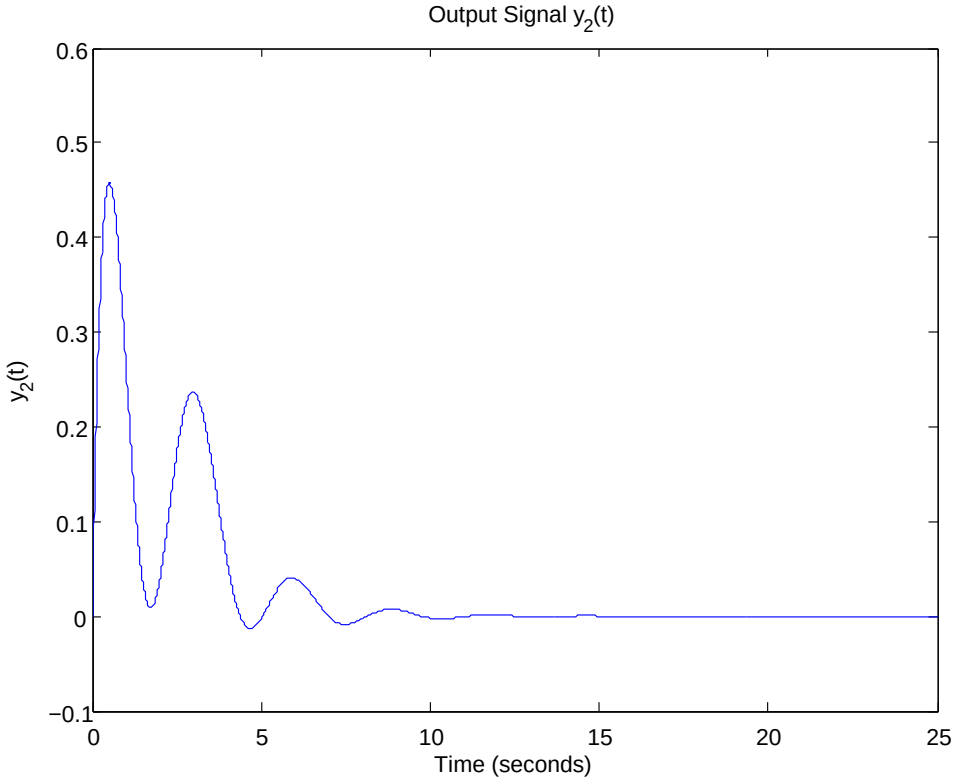
```
%-----------------------------------------------------------------
% P6a
%
% Use lsim to simulate an LTI system for two different input
% signals.
%
close all;                    % close any open figure windows
clear;
%
% Make the system model
```

```
%
numer = [2 1 4];
denom = [1 2 6 5];
H = tf(numer,denom);
%
% Make input x1 a periodic square wave
%
[x1,t1] = gensig('square',4,25,0.01);
%
% Simulate system and plot the output signal
%
y1 = lsim(H,x1,t1);
figure(1);
plot(t1,y1);
title('Output Signal y_1(t)');
xlabel('Time (seconds)');
ylabel('y_1(t)');
%
% Make input x2 a one-sided decaying exponential
%
t2 = 0:0.01:25;
x2 = exp(-t2);
%
% Simulate system and plot the output signal
%
y2 = lsim(H,x2,t2);
figure(2);
plot(t2,y2);
title('Output Signal y_2(t)');
xlabel('Time (seconds)');
ylabel('y_2(t)');
```

Output signal $y_1(t)$ from Matlab Figure Window 1:



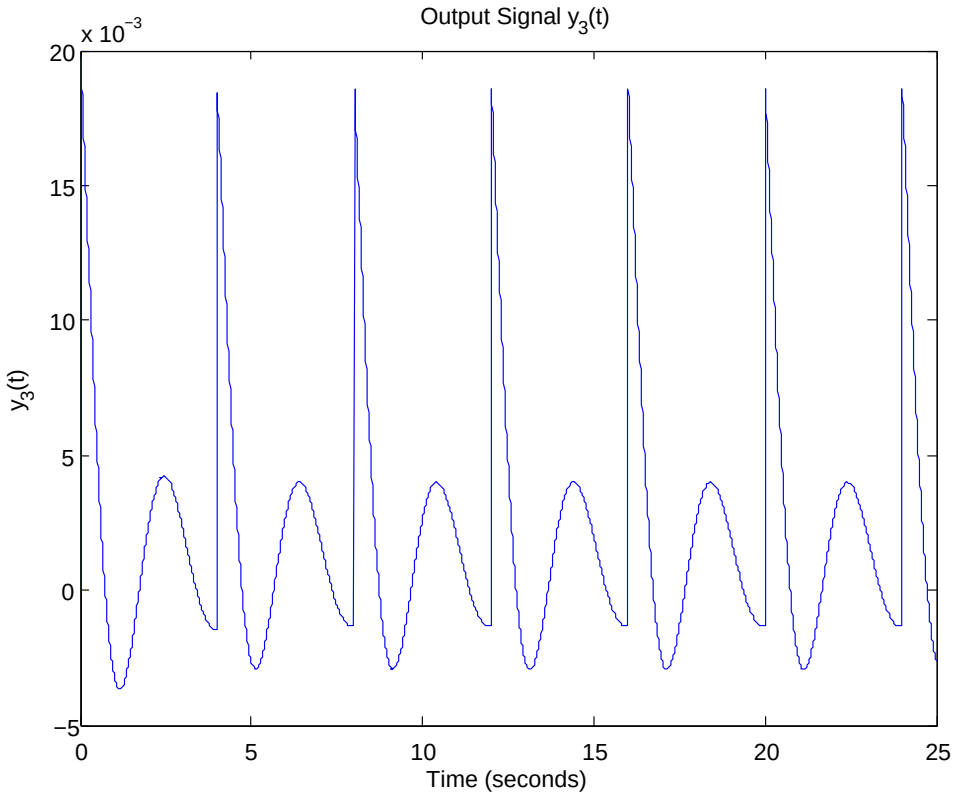
Output signal $y_2(t)$ from Matlab Figure Window 2:



(b) Matlab code:

```
%-----------------------------------------------------------------
% P6b
%
% Use lsim to simulate the LTI system from Problem 6(a) with a
% periodic pulse input signal x3(t).
%
close all;                    % close any open figure windows
clear;
%
% Make the system model
%
numer = [2 1 4];
denom = [1 2 6 5];
H = tf(numer,denom);
%
% Make input x3 a periodic pulse
%
[x3,t3] = gensig('pulse',4,25,0.01);
%
% Simulate system and plot the output signal
%
y3 = lsim(H,x3,t3);
figure(1);
plot(t3,y3);
title('Output Signal y_3(t)');
xlabel('Time (seconds)');
ylabel('y_3(t)');
```

Output signal $y_3(t)$ from Matlab Figure Window 1:



7. Matlab code:

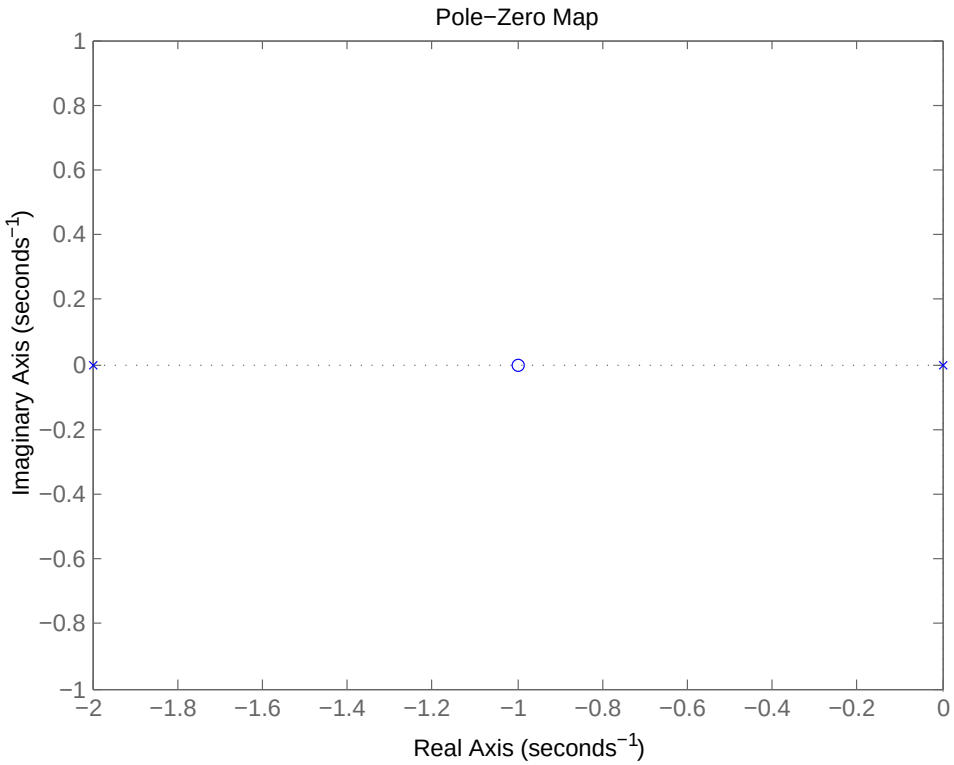
```
%-----------------------------------------------------------------
% P7
%
% Analyze causal LTI open loop system.
%

close all;                    % close any open figure windows
clear;

s = tf('s');
G1 = (s+1)/(s+2);
G2 = 1/(500*s^2);
H = series(G1,G2)

p = pole(H)
figure(1);
pzmap(H);
if (isstable(H))
  disp('The system IS stable.');
else
```

```
  disp('The system is NOT stable.');
end
figure(2);
impulse(H);

t = 0:0.001:5;
x = exp(-t).*cos(5*pi*t);
y = lsim(H,x,t);
figure(3);
plot(t,y);
title('Output Signal y(t)');
xlabel('Time (seconds)');
ylabel('y(t)');
```

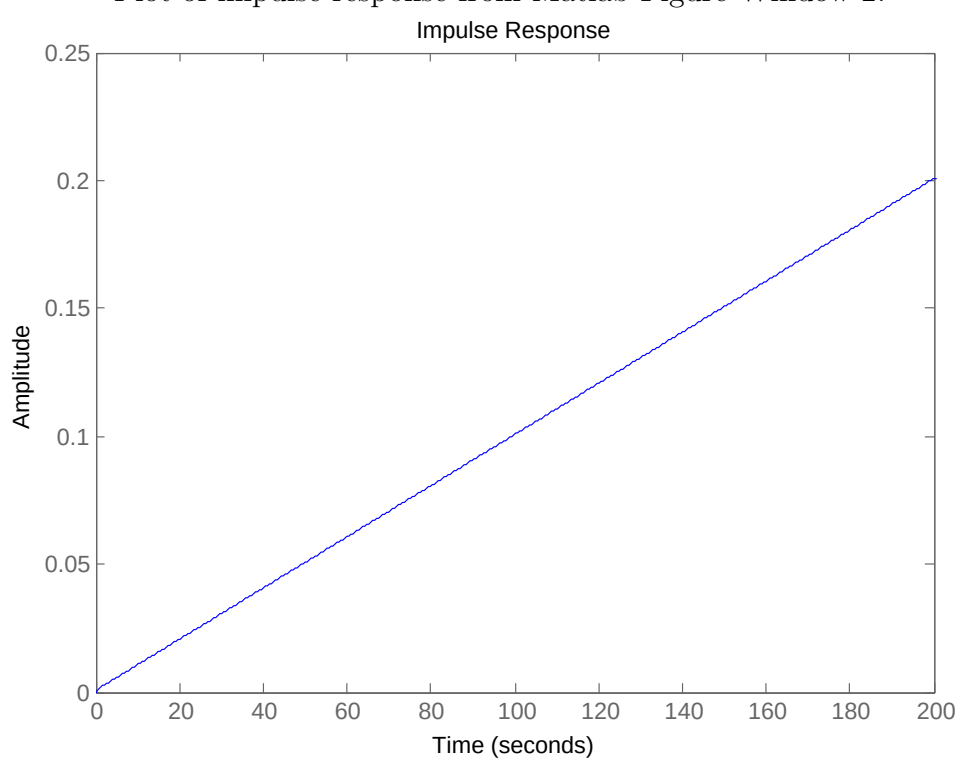
Matlab Command Window Output:

```
>> P7

H =

        s + 1
  ------------------
  500 s^3 + 1000 s^2

Continuous-time transfer function.


p =

     0
     0
    -2

The system is NOT stable.
```

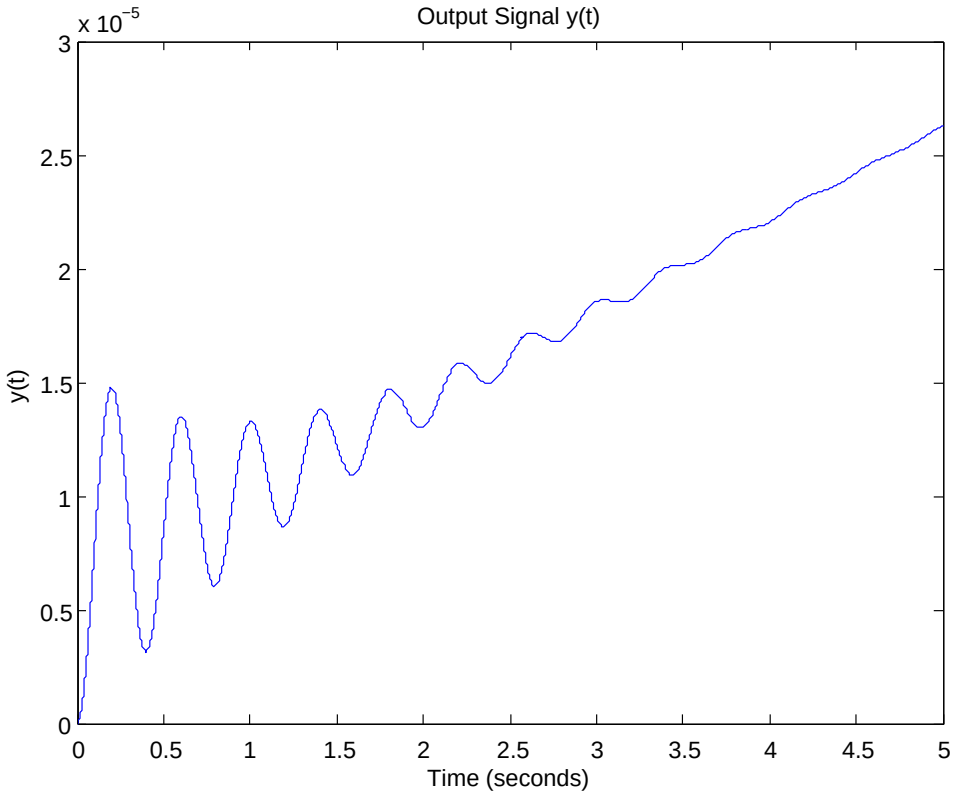
Pole-zero plot from Matlab Figure Window 1:



Plot of impulse response from Matlab Figure Window 2:



Plot of output signal $y(t)$ from Matlab Figure Window 3:



8. Matlab code:

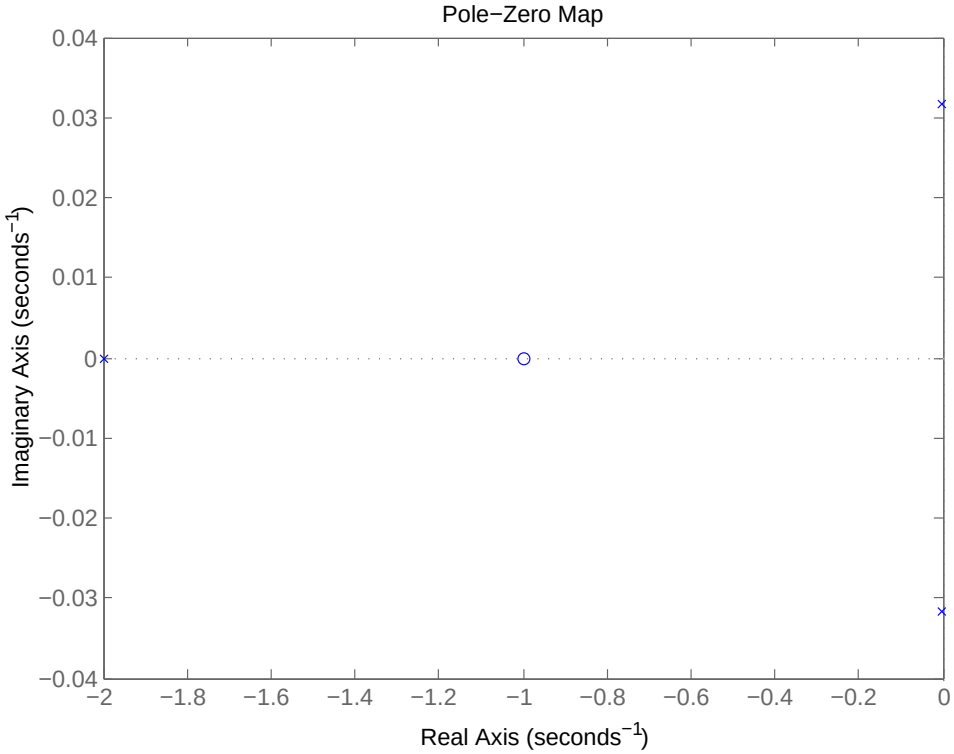
```
%-----------------------------------------------------------------
% P8
%
% Analyze causal LTI closed loop system.
%

close all;                      % close any open figure windows
clear;

s = tf('s');
G1 = (s+1)/(s+2);
G2 = 1/(500*s^2);
G3 = tf(1,1);
H = feedback(series(G1,G2),G3)

p = pole(H)
figure(1);
pzmap(H);
if (isstable(H))
  disp('The system IS stable.');
```

```
else
  disp('The system is NOT stable.');
end
figure(2);
impulse(H);
figure(3);
bode(H);

t = 0:0.01:20000;
x = exp(-t/10000).*cos(pi/500*t);
y = lsim(H,x,t);
figure(4);
plot(t,y);
title('Output Signal y(t)');
xlabel('Time (seconds)');
ylabel('y(t)');
```

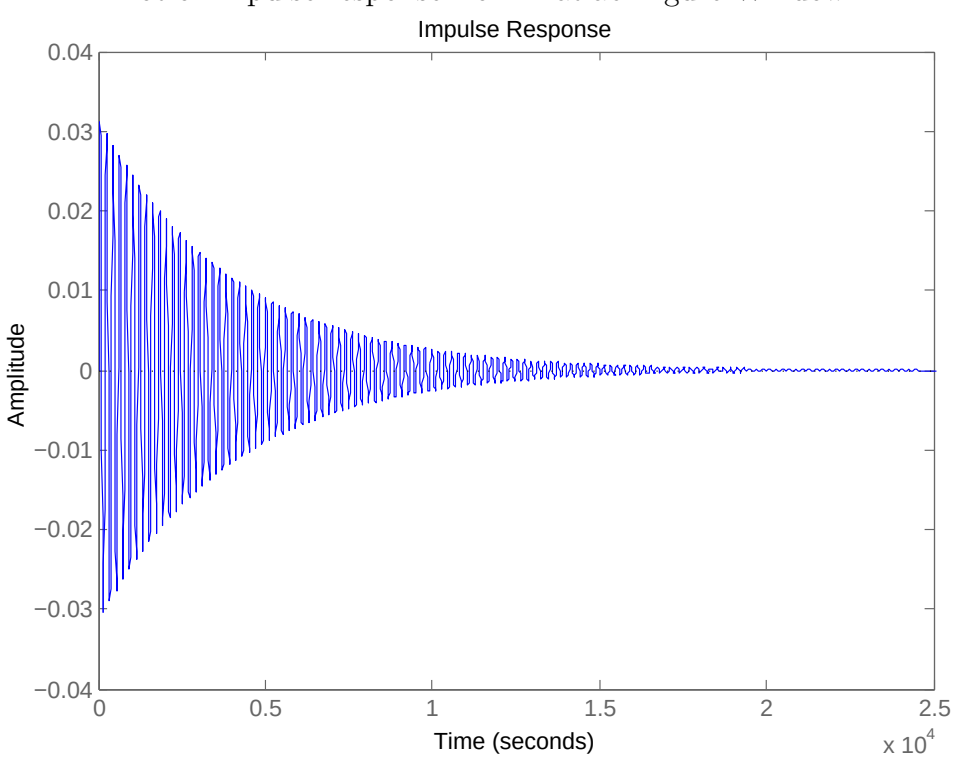
Matlab Command Window Output:

```
>> P8

H =

            s + 1
  --------------------------
  500 s^3 + 1000 s^2 + s + 1

Continuous-time transfer function.


p =

  -1.9995 + 0.0000i
  -0.0002 + 0.0316i
  -0.0002 - 0.0316i

The system IS stable.
```

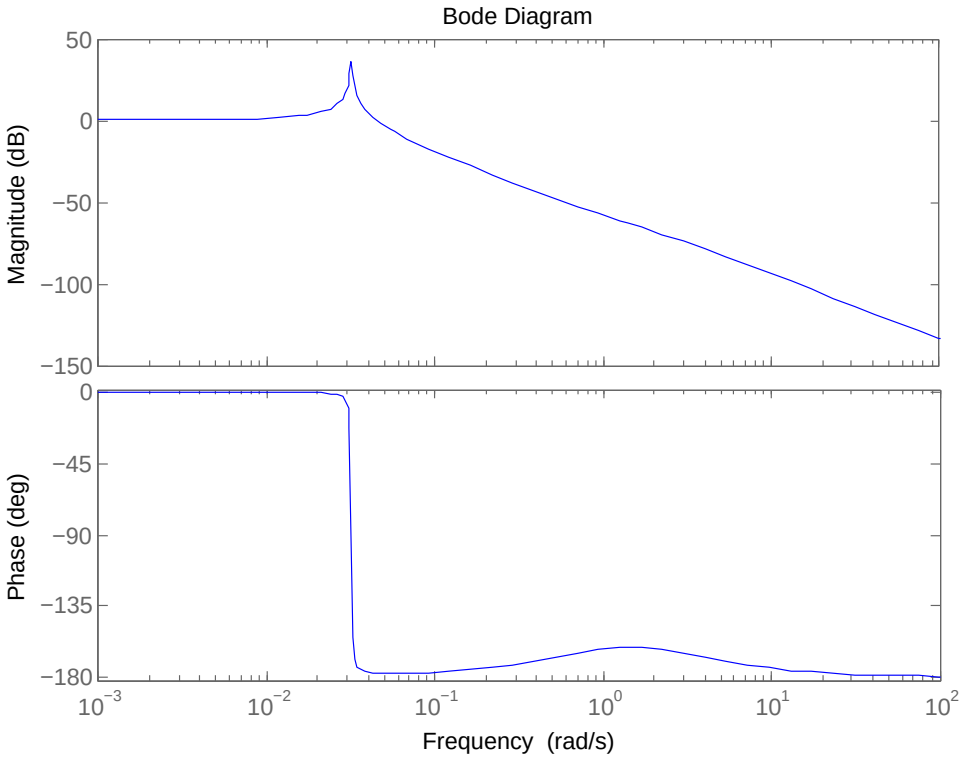
Pole-zero plot from Matlab Figure Window 1:



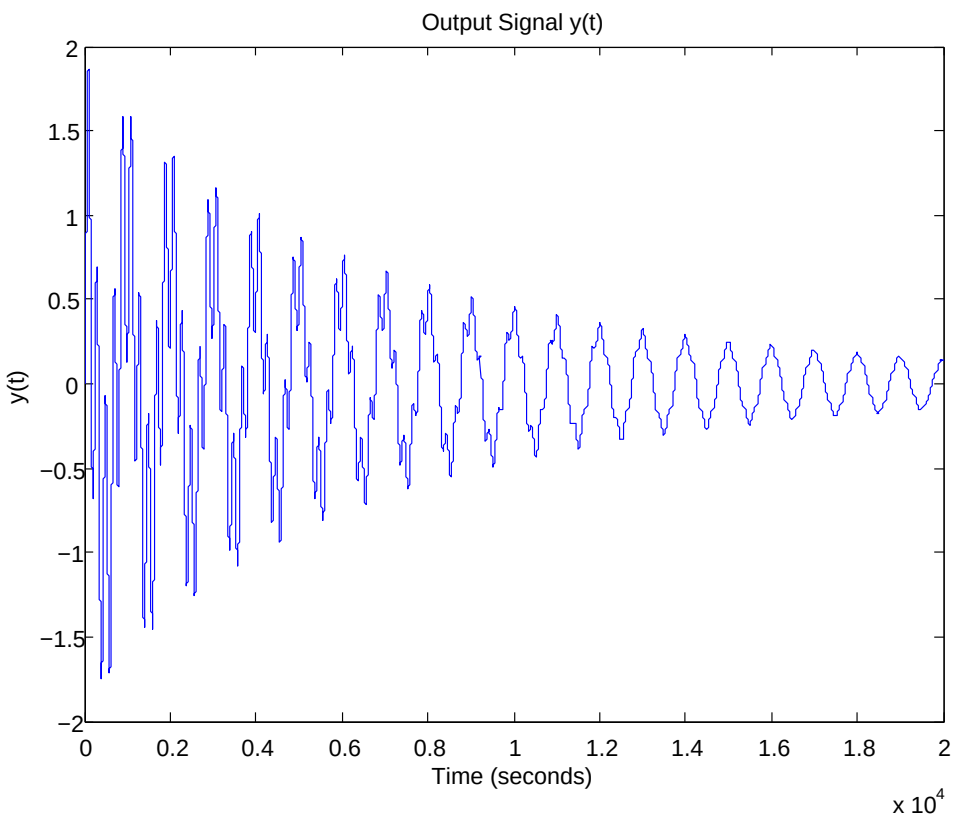
Plot of impulse response from Matlab Figure Window 2:



Plot of frequency response from Matlab Figure Window 3:



Plot of output signal $y(t)$ from Matlab Figure Window 4:



9. (a)
```
>> syms s t
>> xt = exp(-t)*sin(2*t)*heaviside(t);
>> Xs = laplace(xt)

Xs =

2/((s + 1)^2 + 4)
```

(b)
```
>> syms s t
>> xt = exp(-4*t)*heaviside(t) + exp(-5*t)*sin(5*t)*heaviside(t);
>> Xs = laplace(xt)

Xs =

1/(s + 4) + 5/((s + 5)^2 + 25)
```

(c)
```
>>  syms s t
>> H1s = 1/( (s+1)*(s+3) );
>> h1t = ilaplace(H1s)

h1t =

exp(-t)/2 - exp(-3*t)/2
```

(d)
```
>> syms s t
>> H2s = s/(s^2 + s + 1);
>> h2t = ilaplace(H2s)

h2t =

exp(-t/2)*(cos((3^(1/2)*t)/2) - (3^(1/2)*sin((3^(1/2)*t)/2))/3)
```

(e)
```
>> syms s t
>> H3s = s^2/(s^2 + 2*s +1);
>> h3t = ilaplace(H3s)

h3t =

dirac(t) - 2*exp(-t) + t*exp(-t)
```

(f)
```
>> syms s t
>> Xs = 1/(s^2 + 9);
>> xt = ilaplace(Xs)

xt =

sin(3*t)/3
```