Table of Contents

Mayank Vanjani

```
Lab 5: Frequency Responses
11/30/18
```

```
clear;
clc;

step = @(n, t) n >= t;
delta = @(n,t) n == t;
ramp = @(n,t) (n-t).*(n >= t);
```

1 Transfer Function Derivation

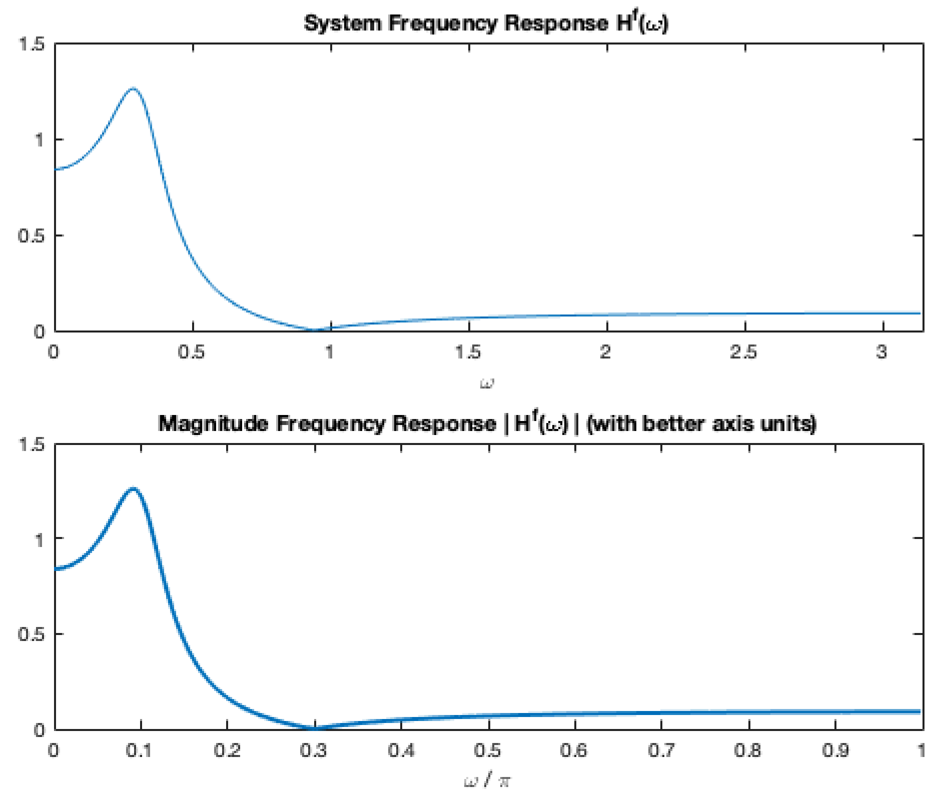
```
y(n) = 0.1x(n) - 0.1176x(n-1) + 0.1x(n-2) + 1.7119y(n-1) - 0.81y(n-2)
y(n) - 1.7119y(n-1) + 0.81y(n-2) = 0.1x(n) - 0.1176x(n-1) + 0.1x(n-2)
H(z) = Y(s)/X(s) = (0.1 - 0.1176s + 0.1s^2) / (1 - 1.7119s + 0.81s^2)
```

2 Magnitude of Frequency Response

```
H(w) = H(z) at z = e^(jw) is discrete Fourier Transform
[H,w] = freqz(b,z) => H is frequency response and w is indices
w from [0,pi] becasue real so repetitive from pi to 2pi
Plot w / pi as freq response because easy to visualize
```

```
numerator = [0.1, -0.1176, 0.1];
denominator = [1, -1.7119, 0.81];
[H, w] = freqz(numerator,denominator);

figure(1); clf;
subplot(2,1,1);
plot(w, abs(H));
xlim( [0,pi] );
xlabel("\omega");
title("System Frequency Response H^f(\omega)");

subplot(2,1,2);
plot(w./pi, abs(H),"LineWidth",2);
```

```
xlim( [0,1] );
xlabel("\omega / \pi");
title("Magnitude Frequency Response | H^f(\omega) | (with better axis
 units)");
```

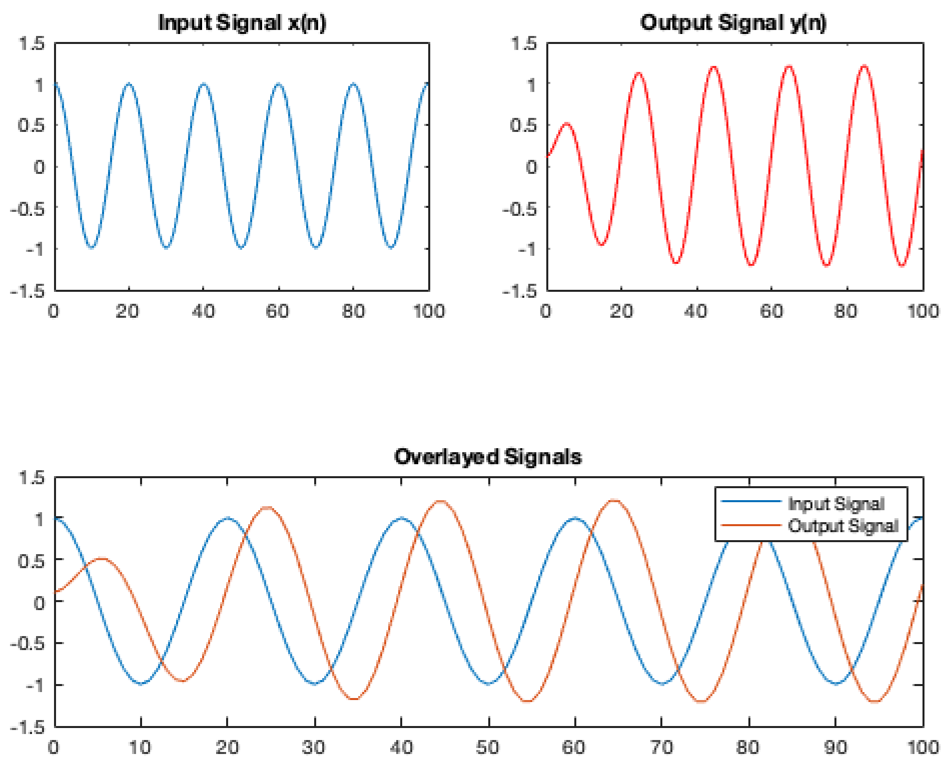


3 Filtered Input to Output

```
Magnitude at 0.1pi is greater than 1 from previous
Results in scaled sinusoid wave (picture) becasue LTI system
Output is same freq with mag and phase shift at that frequency
If at 0.3pi then attenuated so output is 0 line
Slight problem in beginning of input and output sinusoid signal because
 function we have is from 0 to infinity (not start at neg infinity) becaus
 we show transient response of system since it takes time to input full
 signal (steady state response is latter part and transient is beginning
 where scaling is a little off because initially takes time to be
 filtered but once whole signal is inside then its correct). From this,
 we can see how the output signal y(n) will look like (predicted)
Same analysis but we are taking IIR so dies out at higher values (almost
 0) but not quite. Still behaves like steady state response
```

```
n = 0:100;
x = cos( (0.1*pi) * n ) .* step(n,0);
y = filter(numerator,denominator,x);

figure(2); clf;
```

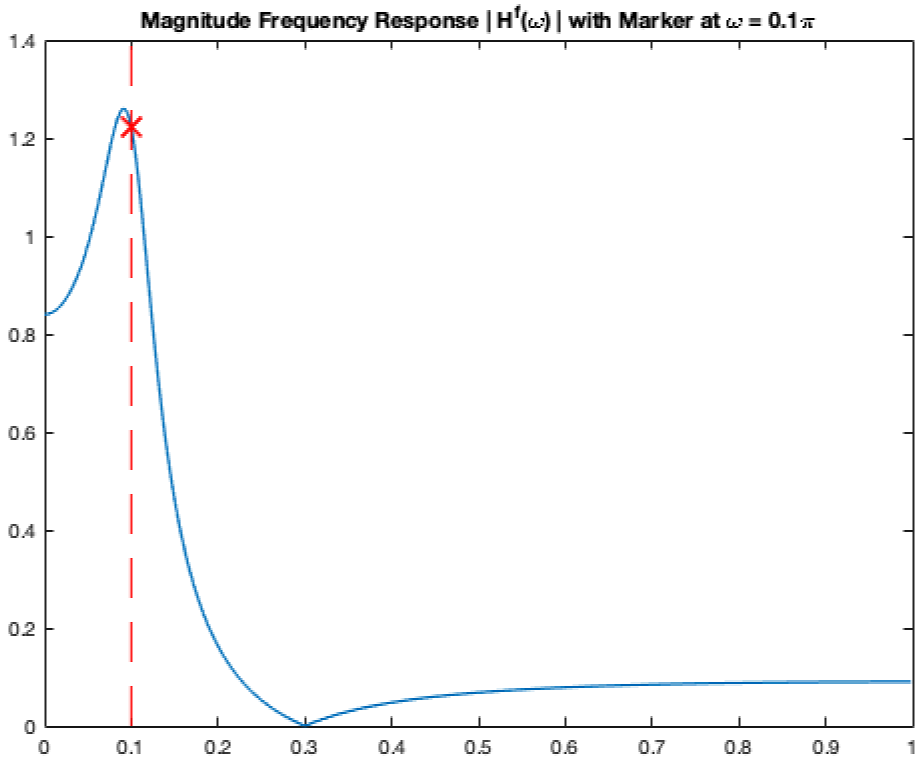
```
subplot(5,2,[1,3]);
plot(0:length(x)-1,x);
ylim( [-1.5,1.5] );
title("Input Signal x(n)");

subplot(5,2,[2,4]);
plot(0:length(y)-1,y,'r');
ylim( [-1.5,1.5] );
title("Output Signal y(n)");

subplot(5,2,[7,8,9,10]);
plot(0:length(x)-1,x);
hold on;
plot(0:length(y)-1,y);
hold off;
ylim( [-1.5,1.5] );
legend( {"Input Signal", "Output Signal"} );
title("Overlayed Signals");
```



4 Evaluate at a Given \omega

```
    Evaluate exactly, put mark at 0.1pi point of freq response
    Evaluate H(z) polynomial exactly at given frequency using polyval

expt = exp(j*0.1*pi);
Hpoint = polyval(numerator,expt) / polyval(denominator,expt);
```

```
figure(3); clf;
plot(w./pi, abs(H), "Linewidth", 1.25);
hold on;
plot(0.1, abs(Hpoint), 'rx', "MarkerSize", 10);
line([0.1 0.1], [0 1.4], "Color", "red", "LineStyle", "--");
hold off;
title( "Magnitude Frequency Response | H^f(\omega) | with Marker at
 \omega = 0.1\pi" );
```

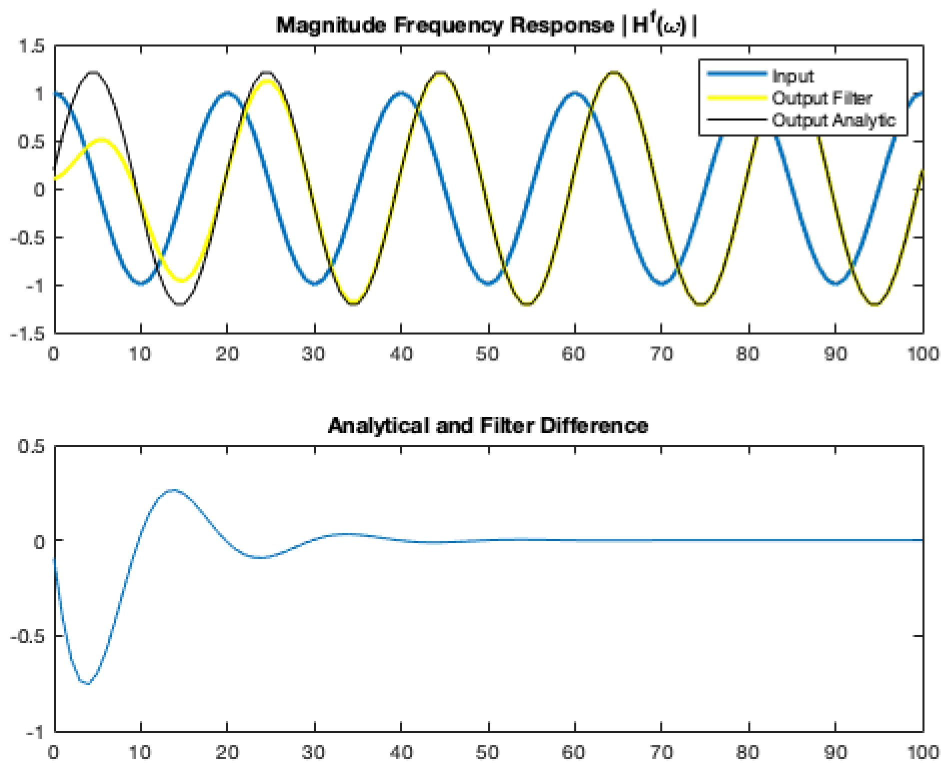


5 Analytic vs. Filter

```
Same as 3 but pure sinusoid, no step function => mag and phase shift
Used filter function implementing difference equation and generate another
 analytical output to compare (resulting in same graph)
After some time transient, filter and analytical are identical
Analytical has no transient because continuous sinusoid
Filter has transient (above Q3) so plot difference in subplot
Observed that s(n) and y(n) are the same after transient dies out
Assume step function implicitley with filter because initial condition
```

```
n = 0:100;
x = cos(0.1*pi*n);
y = filter(numerator,denominator,x);
y2 = abs(Hpoint) .* cos( 0.1*pi*n + angle(Hpoint) );
```

```
figure(4); clf;
subplot(2,1,1);
plot(0:length(x)-1,x,"LineWidth",2);
hold on;
plot(0:length(y)-1,y,'y',"LineWidth",2);
plot(0:length(y)-1,y2,'k');
hold off;
ylim( [-1.5, 1.5] );
legend( {"Input", "Output Filter", "Output Analytic"} );
title("Magnitude Frequency Response | H^f(\omega) |");

subplot(2,1,2);
plot(0:length(y)-1,y-y2);
ylim( [-1, 0.5] );
title("Analytical and Filter Difference");
```

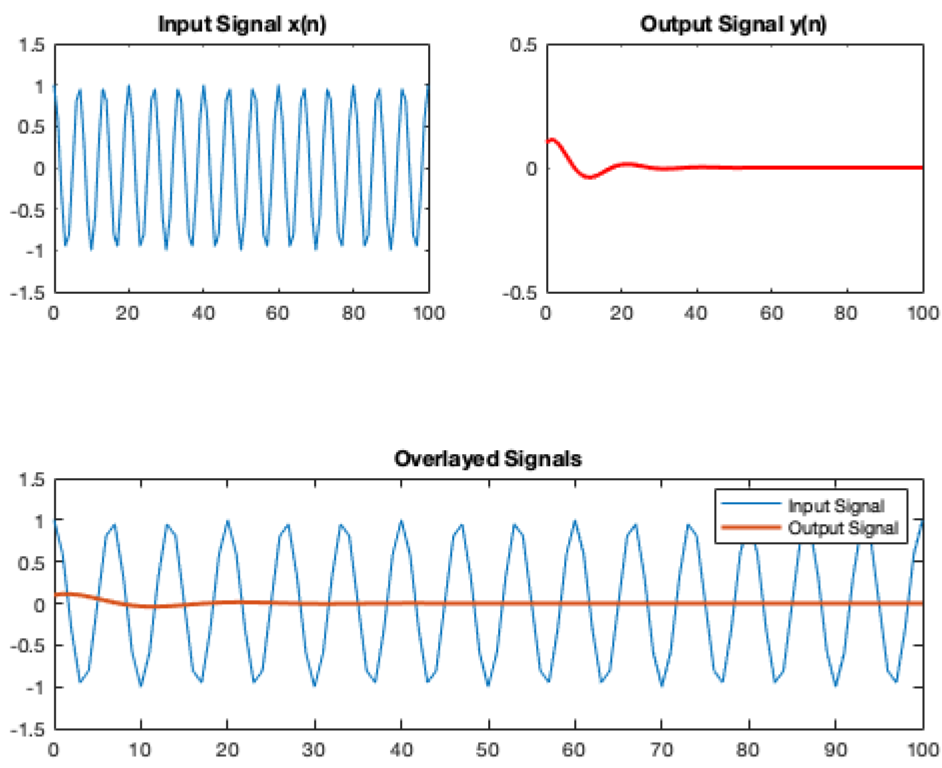


6 Sinusoid Filter

```
Magnitude is 0 at point 0.3pi as seen from the zero on the unit circle
But theres step function so not exact zero because transient response
After dies out, steady state is 0 (predicted value of y(n))
See that the transient response for the filter initially behaves like a
 cos wave and dies out eventually as you filter more values of the cos
 wave which averages the integration to 0 => steady state = 0
```

```
n = 0:100;
```

```
x = cos( (0.3*pi) * n ) .* step(n,0);
y = filter(numerator,denominator,x);

figure(5); clf;
subplot(5,2,[1,3]);
plot(0:length(x)-1,x);
ylim( [-1.5,1.5] );
title("Input Signal x(n)");

subplot(5,2,[2,4]);
plot(0:length(y)-1,y,'r',"LineWidth",2);
ylim( [-0.5,0.5] );
title("Output Signal y(n)");

subplot(5,2,[7,8,9,10]);
plot(0:length(x)-1,x);
hold on;
plot(0:length(y)-1,y,"LineWidth",2);
hold off;
ylim( [-1.5,1.5] );
legend( {"Input Signal", "Output Signal"} );
title("Overlayed Signals");
```

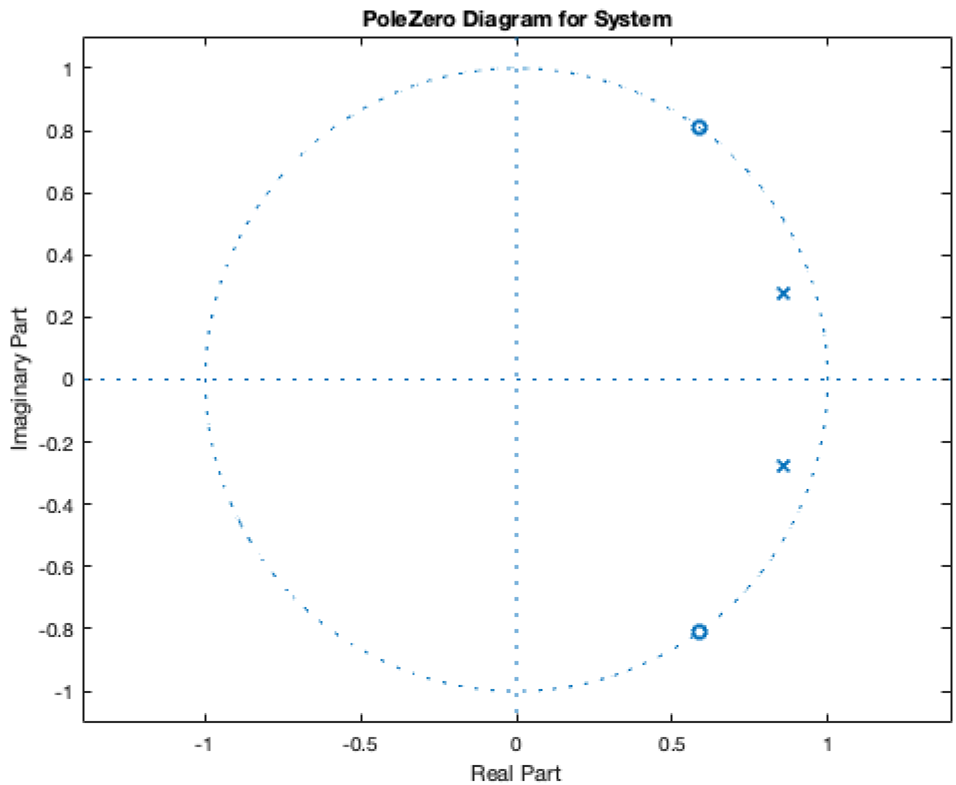


7 Pole-Zero Diagram

```
Zeros and poles from transfer function using partial fractions
```

Close to zero = dip; Close to pole = spike
^^^ Close means w is close in terms of w*pi radians
If poles at 0, dont affect magnitude when w moves around unit circle because distance always 1 (this only affects phase)
We can predict the frequency response from the polezero diagram by looking at the placements of the poles and zeros. As mentioned before, poles = spike and zeros = dips. If the zero is on the unit circle, then there is a dip to 0 at that angle.

```
figure(6); clf;
zplane(numerator,denominator);
title("PoleZero Diagram for System");
```



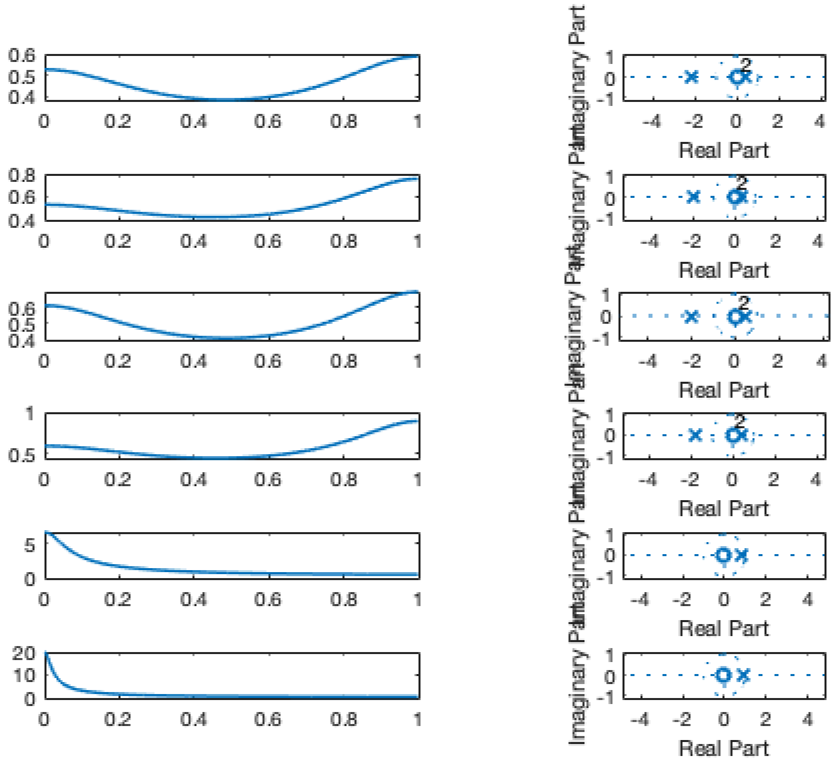
8 Sample Systems

Plot z-plane for all systems and predict freq response
Last one is close to unit circle so large peak
We can predict the transfer functions from the polezero diagram and vice versa even if the graphs are mixed up (without calculations):

```
%  All of them have zeros at the origin which doesnt change anything since
%   nomatter the position on the unit circle changing with w, the distance
%   from the origin is the same
%  The first two are very similar except the poles are slightly closer
```

```
%   (respective to the unit circle or the value of w on the unit
 circle)
%   for the second one resulting in a sharper pole spike (point at w =
 pi is
%   closer to the poles increasing their effect)
%  The next two use the same effect so the second one has a larger
 spike due
%   to closer poles to the w = pi value
%  The last two clearly have poles at w = 0pi = 0 but the second one
 is
%   closer to the unit circle making its peak larger

[H1, w1] = freqz(1,[1 1.8 -0.9]);
[H2, w2] = freqz(1,[1 1.6 -0.72]);
[H3, w3] = freqz(1,[1 1.53 -0.9]);
[H4, w4] = freqz(1,[1 1.4 -0.72]);
[H5, w5] = freqz(1,[1 -0.85]);
[H6, w6] = freqz(1,[1 -0.95]);

figure(7); clf;
subplot(6,2,1);
plot(w1./pi, abs(H1), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,2);
zplane(1,[1 1.8 -0.9]);

subplot(6,2,3);
plot(w2./pi, abs(H2), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,4);
zplane(1,[1 1.6 -0.72]);

subplot(6,2,5);
plot(w3./pi, abs(H3), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,6);
zplane(1,[1 1.53 -0.9]);

subplot(6,2,7);
plot(w4./pi, abs(H4), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,8);
zplane(1,[1 1.4 -0.72]);

subplot(6,2,9);
plot(w5./pi, abs(H5), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,10);
zplane(1,[1 -0.85]);

subplot(6,2,11);
plot(w6./pi, abs(H6), "LineWidth", 1.5);
%xlim( [0,1] );
subplot(6,2,12);
```

```
zplane(1,[1 -0.95]);
```

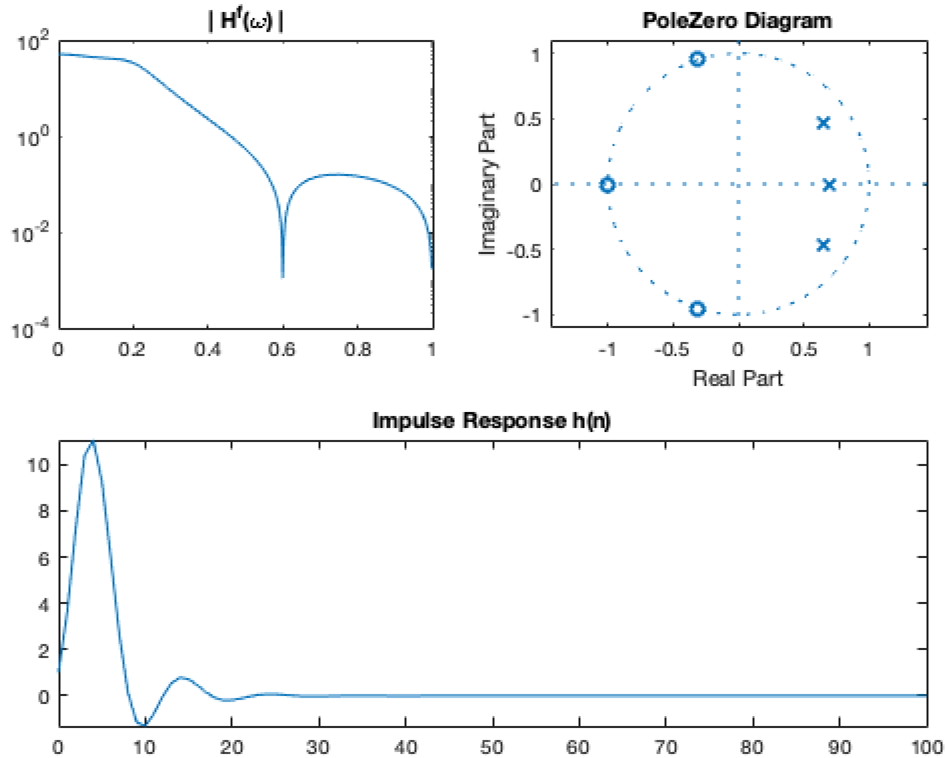


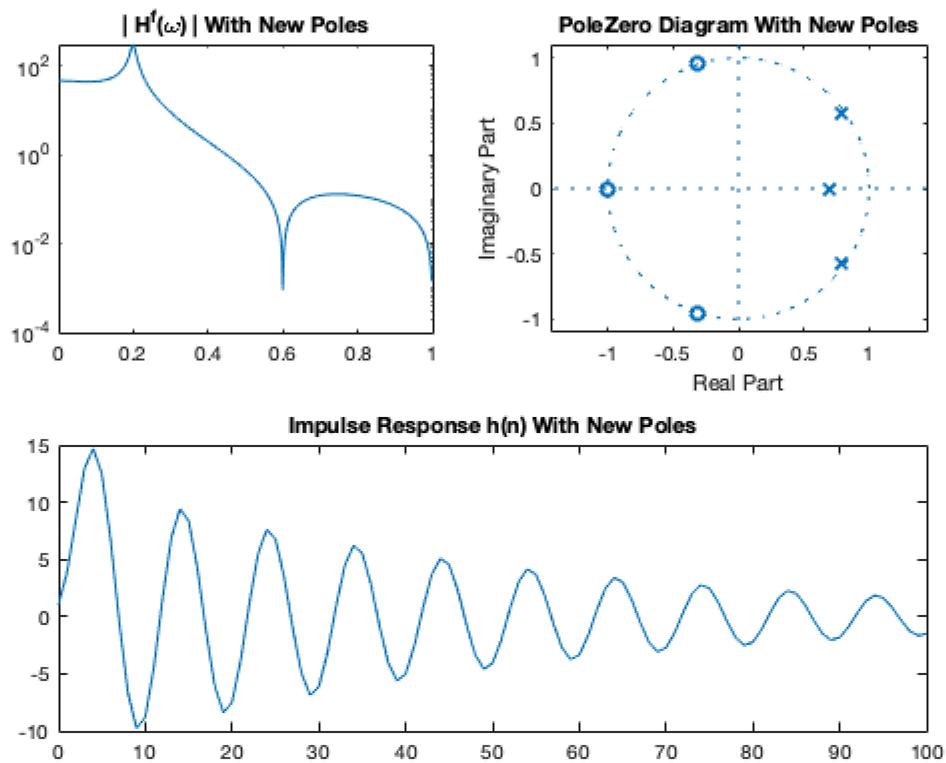
9 Systems with Given Zeros and Poles

```
Semilog y for logscale in y axis, use plot if don't want log scale
Smaller second peak between 0.6 and 1 because farther from all
 poles at 0.8pi versus at 0 (also this means 0.8pi or 0pi at UNIT CIRCLE)
First peak is larger because at 0pi = w, close to all 3 poles
```

```
poles = [ 0.8*exp(j*pi*0.2), 0.8*exp(-1j*pi*0.2), 0.7 ];
zeros = [ -1, exp(j*pi*0.6), exp(-j*pi*0.6) ];
num = poly( zeros );
denom = poly( poles );

[H9, w9] = freqz(num,denom);

n = 0:100;
x = delta(n,0);
h = filter(num,denom,x);

figure(8); clf;
subplot(2,2,1);
semilogy(w9./pi, abs(H9));
% plot(w9./pi, abs(H9));
title("| H^f(\omega) |");
```

```
subplot(2,2,2);
zplane(num,denom);
title("PoleZero Diagram");

subplot(2,2,[3 4]);
plot(0:length(x)-1,h);
title("Impulse Response h(n)");

% ------------------------- %
% New Poles
% ------------------------- %
poles = [ 0.98*exp(j*pi*0.2), 0.98*exp(-1j*pi*0.2), 0.7 ];
denom = poly( poles );

[H9, w9] = freqz(num,denom);

h = filter(num,denom,x);

figure(9); clf;
subplot(2,2,1);
semilogy(w9./pi, abs(H9));
% plot(w9./pi, abs(H9));
title("| H^f(\omega) | With New Poles");

subplot(2,2,2);
zplane(num,denom);
title("PoleZero Diagram With New Poles");

subplot(2,2,[3 4]);
plot(0:length(x)-1,h);
title("Impulse Response h(n) With New Poles");
```


| $H^f(\omega)$ |
PoleZero Diagram
Imaginary Part
Real Part
Impulse Response h(n)


| $H^f(\omega)$ | With New Poles
PoleZero Diagram With New Poles
Imaginary Part
Real Part
Impulse Response h(n) With New Poles

Published with MATLAB® R2018a