Digital Signal Processing HW5 MATLAB Part

Name: Jingyang Zhang
Net ID: jz2807

1) Baseline drift correction using dc notch filter: Download the ECG data file 'ecg_lfn.dat' (on NYU Classes, from the folder “Sample Data”.). The sampling frequency is 1000 samples/second. Plot the ECG signal and observe the baseline drift. Design an order-1 recursive filter with a frequency response null at omega = 0, and a pole on the positive real axis. This system is a dc-notch filter. Apply your dc-notch filter to the ECG data to remove the baseline drift. Evaluate the effectiveness of the filter by overlaying the filtered signal with the original ECG signal in the same plot. Comment on the effect of the pole position. Hint: your filter transfer function will have the form of $H(z) = \frac{z-q}{z-p}$. The zero should be $q = 1$. The pole should have a general form of $p = r$ with $0 < r < 1$. You are asked to evaluate the effect of varying r. For each value of r, you should show the frequency response of your filter, and the overlay of the original signal with the filtered signal.

Solution:

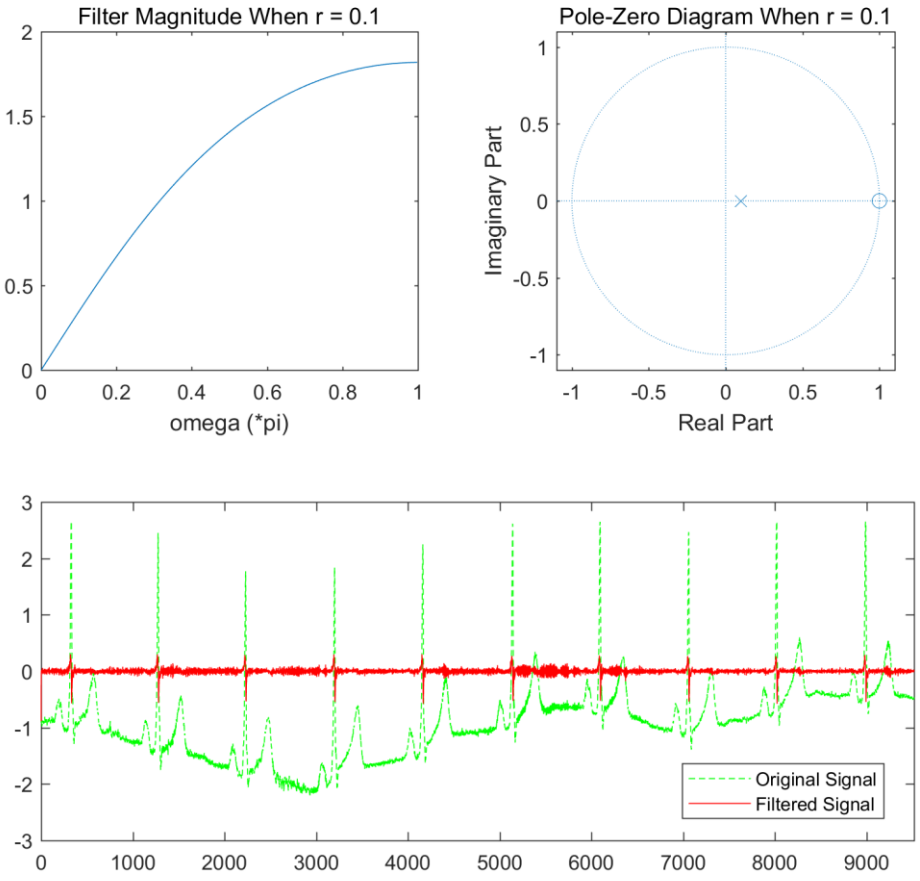
.m file(s): jyz_HW6_1.m

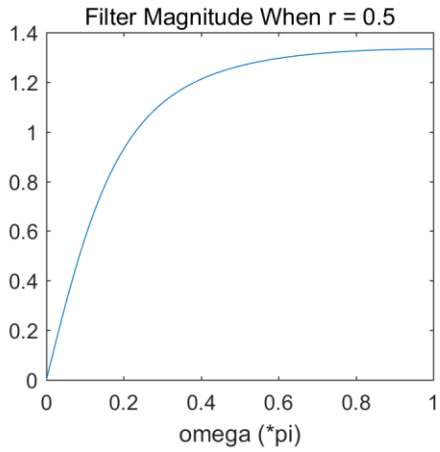
Code:

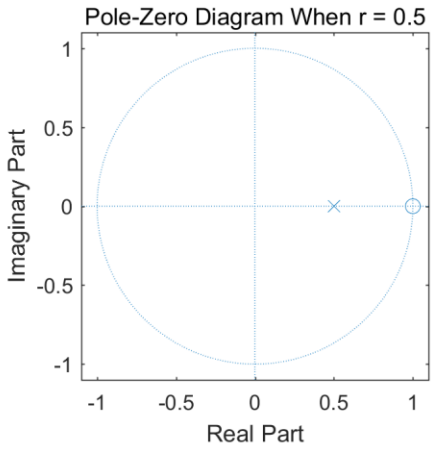
```
clear
close all

load ecg_lfn.dat
x = ecg_lfn;
fs = 1000;
b = [1 -1];
r1 = -0.1; r2 = -0.5; r3 = -0.9;
a1 = [1 r1]; a2 = [1 r2]; a3 = [1 r3];
[H1,w1] = freqz(b,a1);[H2,w2] = freqz(b,a2);[H3,w3] = freqz(b,a3);
y1 = filter(b,a1,x);y2 = filter(b,a2,x);y3 = filter(b,a3,x);

figure(1)
subplot(2,2,1)
plot(w1/pi,abs(H1))
xlabel('omega (*pi)')
title('Filter Magnitude When r = 0.1')
subplot(2,2,2)
zplane(b,a1,'k')
xlim([-1.1,1.1]);ylim([-1.1,1.1]);
title('Pole-Zero Diagram When r = 0.1')
subplot(2,2,[3,4])
plot((0:length(x)-1),x,'g--',(0:length(y1)-1),y1,'r')
xlim([0,length(x)-1])
legend('Original Signal', 'Filtered Signal')

figure(2)
```

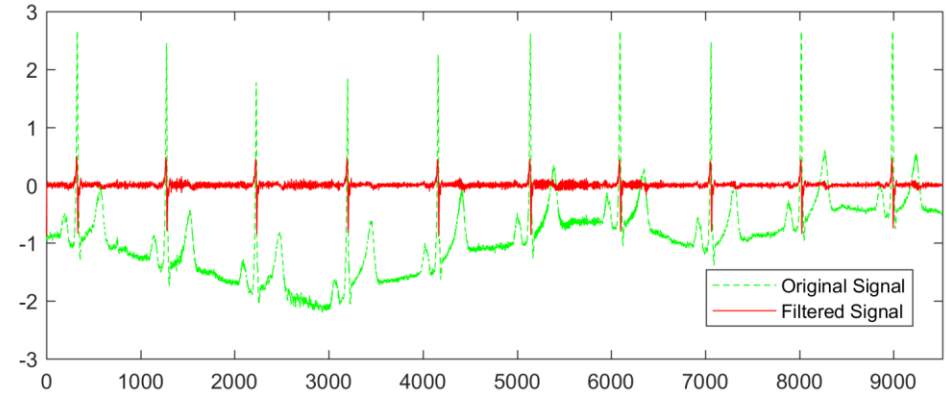
```
subplot(2,2,1)
plot(w2/pi,abs(H2))
xlabel('omega (*pi)')
title('Filter Magnitude When r = 0.5')
subplot(2,2,2)
zplane(b,a2,'k')
xlim([-1.1,1.1]);ylim([-1.1,1.1]);
title('Pole-Zero Diagram When r = 0.5')
subplot(2,2,[3,4])
plot((0:length(x)-1),x,'g--',(0:length(y2)-1),y2,'r')
xlim([0,length(x)-1])
legend('Original Signal', 'Filtered Signal')

figure(3)
subplot(2,2,1)
plot(w3/pi,abs(H3))
xlabel('omega (*pi)')
title('Filter Magnitude When r = 0.9')
subplot(2,2,2)
zplane(b,a3,'k')
xlim([-1.1,1.1]);ylim([-1.1,1.1]);
title('Pole-Zero Diagram When r = 0.9')
subplot(2,2,[3,4])
plot((0:length(x)-1),x,'g--',(0:length(y3)-1),y3,'r')
xlim([0,length(x)-1])
legend('Original Signal', 'Filtered Signal')
```

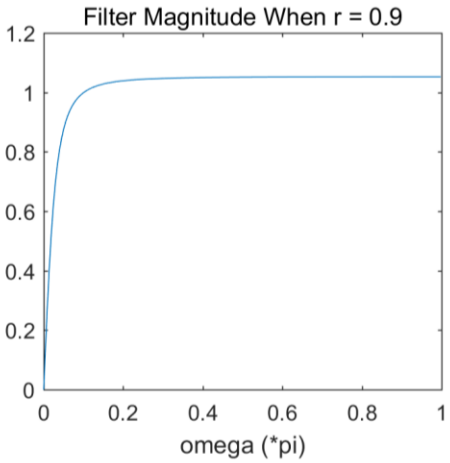
Result(plots):

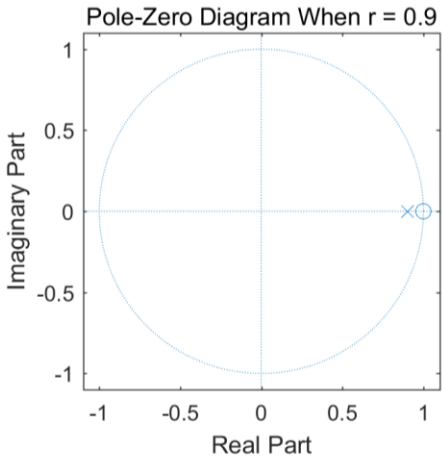


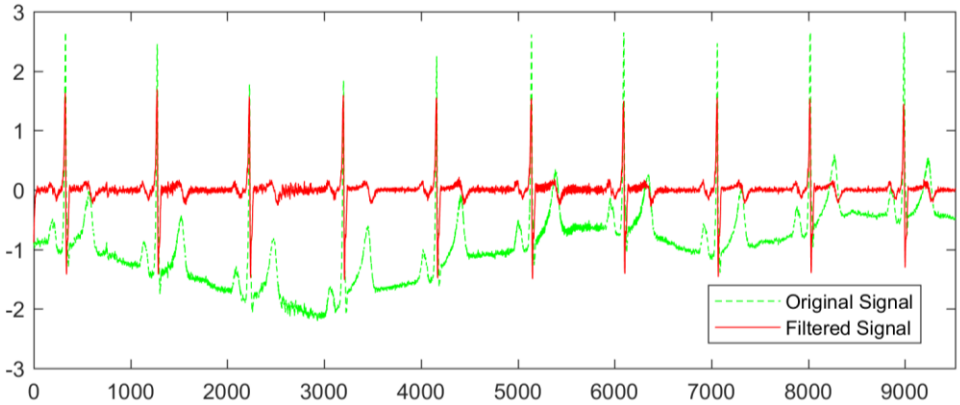












Comments: When r is close to 0, the filter not only eliminate the signal at $\omega = 0$, but also suppress the signal near it (in a considerably large range), so the filtered signal is reduced in scale, the baseline drift is annihilated very well. As r gets close to 1, the suppressed range of ω becomes narrow, the filtered signal scale is not suppressed acutely any more, however, there is some noise left in the signal, which could be considered as an expense of good effect of the narrow dc-notch filter.

2) In the Notch Filter Demo that we went through in the class (uploaded under Lecture 6), we designed a second order IIR filter to remove a toner noise. You may have noticed that the toner noise was not completely removed as you can still hear it. In fact this is because the toner noise is not exactly at a single frequency f0=500Hz. It actually occupies a narrow band from f1=500-500/297 Hz to f2=500+500/297 Hz. One way to further suppress the toner frequency is by using zeros to knock out frequencies at f1 and f2. Unfortunately, if you only use zeros (FIR filter), it will significantly attenuate low frequency below f1. To compensate for this, you could use a IIR filter with polls at these frequencies as well. Please repeat the experiment similar to the Notch filter demo, comparing a 4^{th} order FIR filter (with zeros at +/- f1 and +/- f2) with a 4^{th} order IIR filter (with zeros at +/- f1 and +/- f2 and poles at +/- f1 and +/- f2, but the poles should be inside the unit circle). Compare the frequency responses, zero-pole patterns, the difference equations of the filters and the impulse response of the filters. Please also compare the quality of the filtering through both playing the sound file and displaying the spectrum. Furthermore, compare these results with those using a second order filter (FIR and IIR) to only knock out the center frequency of 500Hz (You can use the demo program for the 2^{nd} order filters). You can download the speech files (clean.wav, noisy.wav) from NYUClasses Sample Data folder. You should submit your derivation of the appropriate FIR and IIR filter coefficients, in addition to the MATLAB codes and plots and your observations.

Solution:

.m file(s): jyz_HW6_2.m

Code:

```
clear
close all

del = @(n) (n == 0); % Function
nn = 0:20;
delta = del(nn);
[c, fsc] = audioread('clean.wav');
[n, fsn] = audioread('noisy.wav');
t = 0:1/fsn:(length(n)-1) / fsn;
f1a = 500-500/297; f1d = 2*pi*f1a/fsn;
f2a = 500+500/297; f2d = 2*pi*f2a/fsn;
fa = 500; fd = 2*pi*fa/fsn;
r = 0.95;
cf = fft(c(12954:13977));
nf = fft(n(12954:13977));
b = [1 -2*cos(fd) 1];
b = b/polyval(b, -1); % make peak gain = 1
a = [1 -2*r*cos(fd) r^2];
a = a/polyval(a, -1); % make peak gain = 1
[H, om] = freqz(b,a);
h = filter(b,a,delta);
y = filter(b, a, n);
yf = fft(y(12954:13977));
bFIR = conv([1 -2*cos(f1d) 1], [1 -2*cos(f2d) 1]);
bFIR = bFIR/polyval(bFIR, -1); % make peak gain = 1
```

```
aFIR = [1 0 0 0 0];
[HFIR, omFIR] = freqz(bFIR, aFIR);
yFIR = filter(bFIR, aFIR, n);
yFIRf = fft(yFIR(12954:13977));
hFIR = filter(bFIR,aFIR,delta);
bIIR = conv([1 -2*cos(f1d) 1], [1 -2*cos(f2d) 1]);
bIIR = bIIR/polyval(bIIR, -1); % make peak gain = 1
aIIR = conv([1 -2*r*cos(f1d) r^2], [1 -2*r*cos(f2d) r^2]);
aIIR = aIIR/polyval(aIIR, -1); % make peak gain = 1
[HIIR, omIIR] = freqz(bIIR, aIIR);
yIIR = filter(bIIR, aIIR, n);
yIIRf = fft(yIIR(12954:13977));
hIIR = filter(bIIR,aIIR,delta);

figure(1)
plot(om/(2*pi)*fsn, abs(H), omFIR/(2*pi)*fsn, abs(HFIR), omIIR/(2*pi)*fsn, abs(HIIR))
xlabel('Frequency(cycles/second = Hz)')
title('Frequency Response')
legend('2nd Order IIR Filter in Demo','4th Order FIR Filter','4th Order IIR Filter')

figure(2)
subplot(2,3,1)
zplane(b,a)
xlim([-1.1 1.1]);ylim([-1.1 1.1]);
title('2nd Order IIR Filter in Demo')
subplot(2,3,3)
zplane(bFIR,aFIR)
xlim([-1.1 1.1]);ylim([-1.1 1.1]);
title('4th Order FIR Filter')
subplot(2,3,5)
zplane(bIIR,aIIR)
xlim([-1.1 1.1]);ylim([-1.1 1.1]);
title('4th Order IIR Filter')
subplot(2,3,4)
zplane(bIIR,aIIR)
xlim([0.87 0.94]);ylim([0.35 0.39]);
title('4th Order IIR Filter-Zoom in 1')
subplot(2,3,6)
zplane(bIIR,aIIR)
xlim([0.87 0.94]);ylim([-0.39 -0.35]);
title('4th Order IIR Filter-Zoom in 2')

figure(3)
subplot(3,2,1)
plot(0:fsn/2/512:fsn/2-fsn/2/512, abs(cf(1:512)));
title('Spectrum of Clean Signal Segment')
xlabel('Frequency (Hz)')
subplot(3,2,2)
plot(0:fsn/2/512:fsn/2-fsn/2/512, abs(nf(1:512)));
```

```
title('Spectrum of Noisy Signal Segment')
xlabel('Frequency (Hz)')
subplot(3,2,3)
plot(0:fsn/2/512:fsn/2-fsn/2/512, abs(yf(1:512)));
title('Spectrum of Filtered Signal Segment by 2nd Order IIR Filter in Demo')
xlabel('Frequency (Hz)')
subplot(3,2,[5,6])
plot(0:fsn/2/512:fsn/2-fsn/2/512, abs(yFIRf(1:512)));
title('Spectrum of Filtered Signal Segment by 4th Order FIR Filter')
xlabel('Frequency (Hz)')
subplot(3,2,4)
plot(0:fsn/2/512:fsn/2-fsn/2/512, abs(yIIRf(1:512)));
title('Spectrum of Filtered Signal Segment by 4th Order IIR Filter')
xlabel('Frequency (Hz)')

figure(4)
subplot(3,1,1)
stem(0:20,h,'.')
xlim([-1 21])
xlabel('n')
title('Impulse Response of 2nd Order IIR Filter in Demo')
subplot(3,1,2)
stem(0:20,hFIR,'.')
xlim([-1 21])
xlabel('n')
title('Impulse Response of 4th Order FIR Filter')
subplot(3,1,3)
stem(0:20,hIIR,'.')
xlim([-1 21])
xlabel('n')
title('Impulse Response of 4th Order IIR Filter')
sound(y,fsn);pause;
sound(yFIR,fsn);pause;
sound(yIIR,fsn);pause;
```

Derivation of the coefficients:
For 4^{th} order FIR filter, zeros should be at unit circle and f = +/- f1 and +/- f2. For 4^{th} order IIR filter, it should be with zeros at f = +/- f1 and +/- f2 and poles at f = +/- f1 and +/- f2, but the poles should be inside the unit circle, the modulus of poles are set to 0.95.

In MATLAB, it is easy to find the coefficients with *conv()* function. For 4^{th} order FIR filter, coefficients of frequency response numerator should be conv([1,-2*cos(f1),1], [1,-2*cos(f2),1]), coefficients of denominator should be [1,0,0,0,0] to ensure it is a causal system. For 4^{th} order FIR filter, coefficients of frequency response numerator should be conv([1,-2*cos(f1),1], [1,-2*cos(f2),1]), coefficients of denominator should be conv([1,-2*r*cos(f1),r^2], [1,-2*r*cos(f2),r^2]), where r = 0.95.

Finally, to ensure the peak gain ($H^f(\pi)$ or H(-1)) is equal to 1, it is necessary to normalize the scale of the coefficients. In MATLBA, it can be implemented by the following code:

For 4^{th} order FIR filter:

b = b/polyval(b, -1);

For 4th order FIR filter:

b = b/polyval(b, -1);

a = a/polyval(a, -1);

In this program, the final coefficients of the filters are:

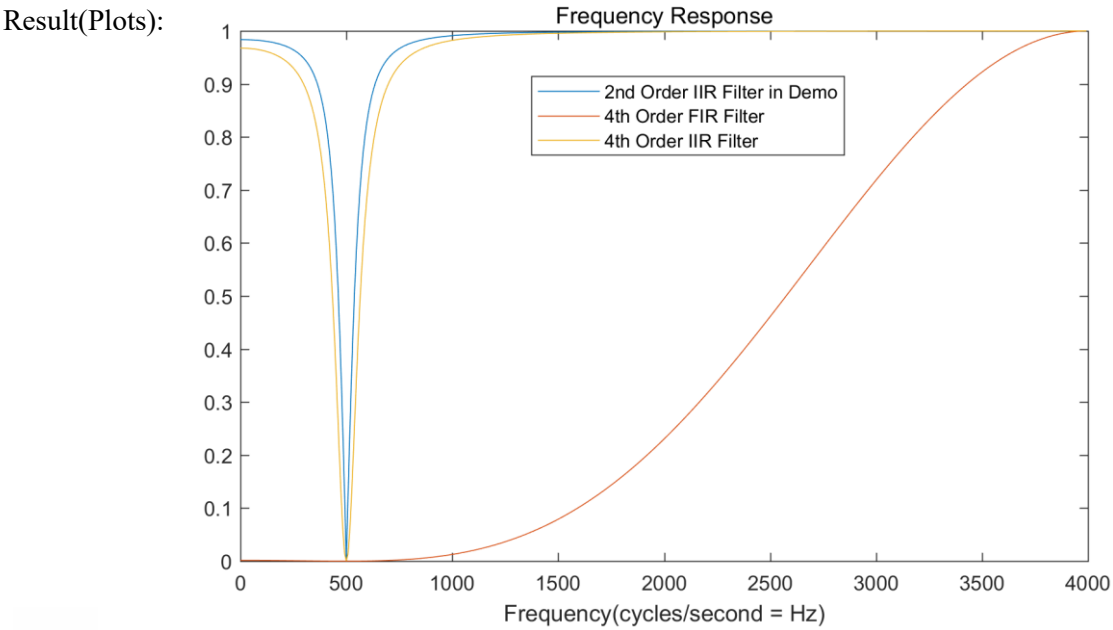
bFIR = [0.0675, -0.2496, 0.3657, -0.2496, 0.0675]

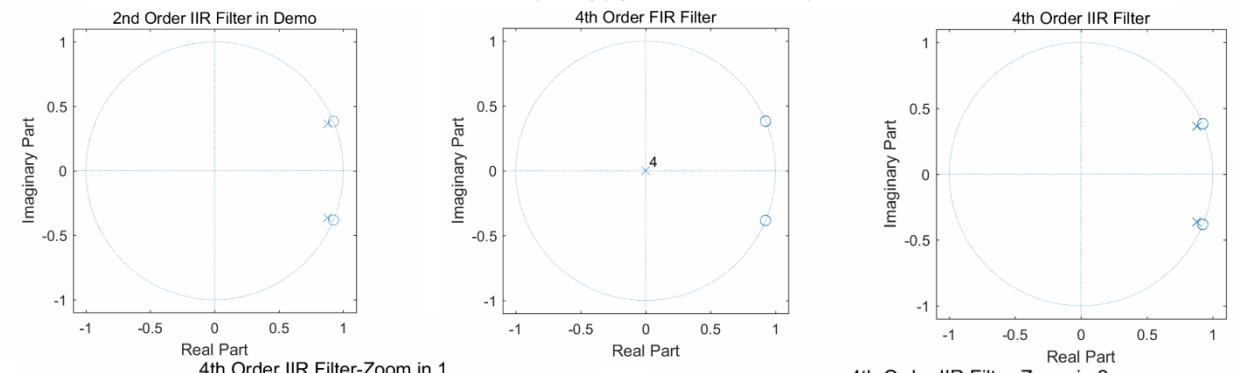
aFIR = [1, 0, 0, 0, 0]

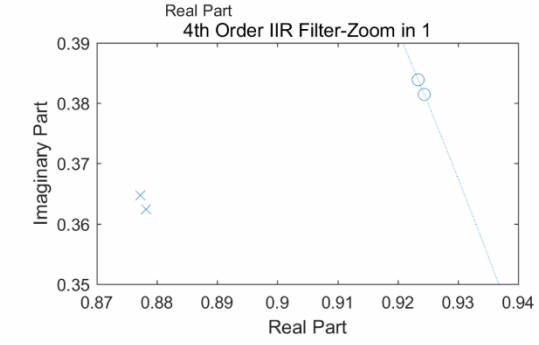
bIIR = [0.0675, -0.2496, 0.3657, -0.2496, 0.0675]

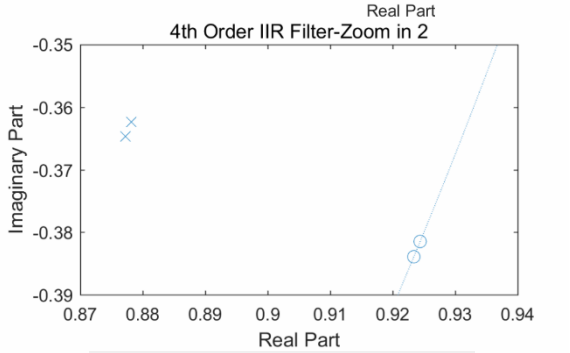
aIIR = [0.0747, -0.2624, 0.3652, -0.2368, 0.0609]

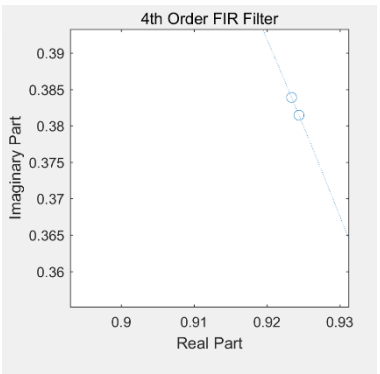
Result(Plots):

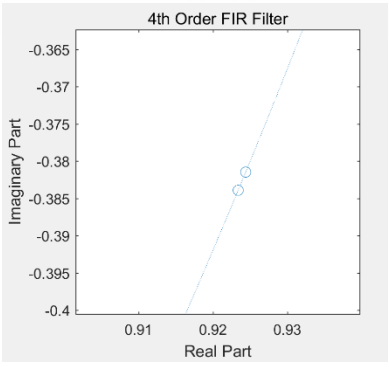


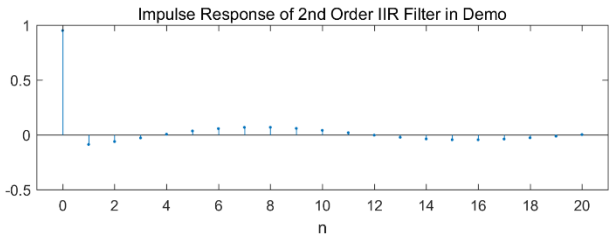


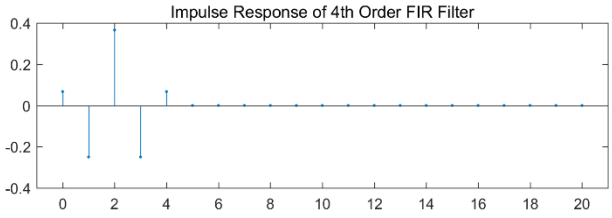


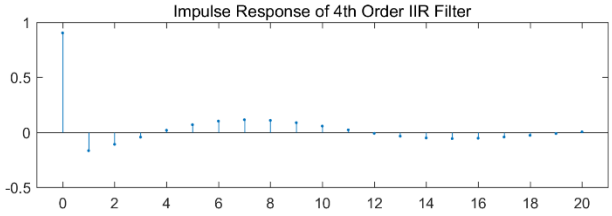


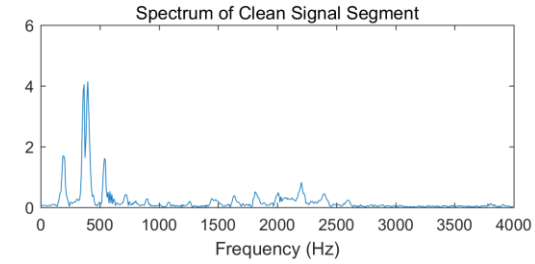


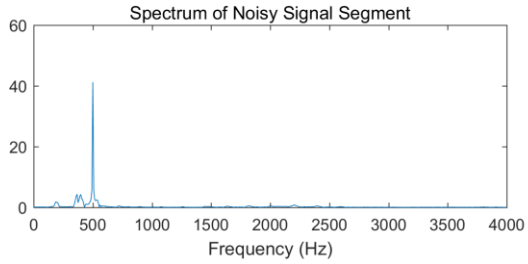


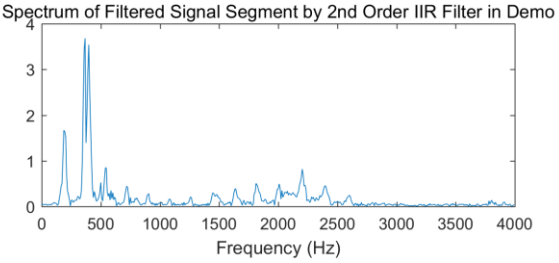


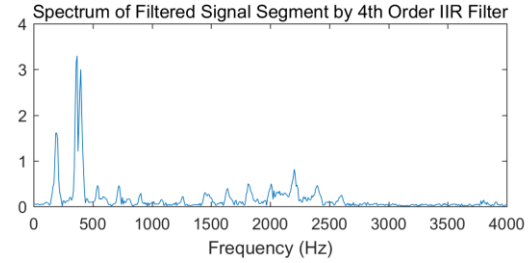


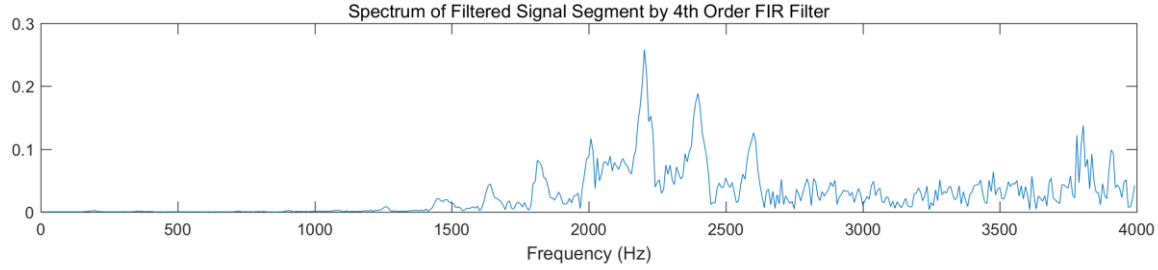












Comment:

1) Using the coefficients of filters, it is easy to find the difference equations of the filters.

a) For 2nd Order IIR Filter in Demo:

b = [0.2599, -0.4802, 0.2599], a = [0.2733, -0.4798, 0.2467]

The difference equations:

$$0.2733y(n) - 0.4798y(n-1) + 0.2467y(n-2) = 0.2599x(n) - 0.4802x(n-1) + 0.2599x(n-2)$$

b) For designed 4th order FIR filter:

bFIR = [0.0675, -0.2496, 0.3657, -0.2496, 0.0675], aFIR = [1, 0, 0, 0, 0]

The difference equations:

$$y(n) = 0.0675x(n) - 0.2496x(n-1) + 0.3657x(n-2) - 0.2496x(n-3) + 0.0675x(n-4)$$

c) For designed 4th order IIR filter:

bIIR = [0.0675, -0.2496, 0.3657, -0.2496, 0.0675], aIIR = [0.0747, -0.2624, 0.3652, -0.2368, 0.0609]

The difference equations:

$$0.0747y(n) - 0.2624y(n-1) + 0.3652y(n-2) - 0.2368y(n-3) + 0.0609y(n-4) =$$
$$0.0675x(n) - 0.2496x(n-1) + 0.3657x(n-2) - 0.2496x(n-3) + 0.0675x(n-4)$$

2) From the plot of different filters' frequency responses, we can find that both IIR filters knock out the signal at f = 500Hz. After zooming in the plot of 4th order IIR filter's frequency response, we can see it is actually a bandpass filter which knock out the signal between (f1, f2). As for the 4th order FIR filter, it significantly suppresses the signal of low frequency below f1.

3) From the zero-pole patterns of filters, we can find that:

a) For 2nd Order IIR Filter in Demo:
It has two zeros at f = 500Hz and two poles at f = 500Hz but with the modulus of r = 0.95.

b) For designed 4th order FIR filter:
It has 4 zeros at f =±f1 and ±f2 on the unit circle, and a 4th order pole at 0.

c) For designed 4th order IIR filter:
It has 4 zeros at f =±f1 and ±f2 on the unit circle, and 4 zeros at f =±f1 and ±f2, but with the modulus of r = 0.95.

4) From the plots of different filters' impulse responses, we can find that the FIR filter on has non-zero values at n = 0, 1, 2, 3, 4, for it is a 4th order FIR filter, so it should have finite impulse response and the maximum of non-zero point should be at n = 4. Both IIR filter have infinite impulse response, as it can be seen on the plot, they share a very similar shape.

5) From the spectrum of the signal of clean sound, noisy sound, output sound filtered by 2nd Order IIR Filter in Demo, output sound filtered by designed 4th order FIR filter, and output sound filtered by designed 4th order IIR filter, we can find that:

a) The noise is not exactly at a single frequency f = 500Hz, it actually occupies a narrow band from f1 to f2.

b) By using the 2nd Order IIR Filter in Demo, the noise was not completely removed as we can still see some noise around 500 Hz on the spectrum.

c) By using the designed 4th order FIR filter, the signal of frequency below f1 is significantly attenuated. The scale of the output signal is very small comparing to the other output signals. But still, it knocks out the noise to some extent.

d) By using the designed 4th order IIR filter, the output signal get rid of the noise eventually, the scale of the signal is also similar to the original clean signal as well, it is safe to say that the designed 4th order IIR filter has the best noise-eliminating effect among all the filters.

6) At last, when we listen to all the output sounds, we can still hear the toner noise in the signal filtered by the second order IIR filter; There is no more noise in the output signal filtered by the designed 4th order FIR filter, but it sounds significantly low in volume, which is the worst quality among three; The output sound signal filtered by the designed 4th order IIR filter sounds almost the same to the original clean one, and it has the reasonable volume, has the best quality among three with no doubt.