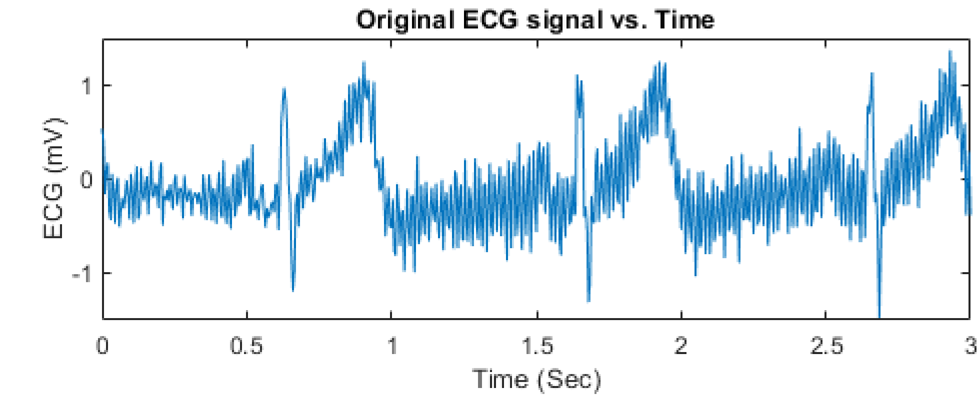
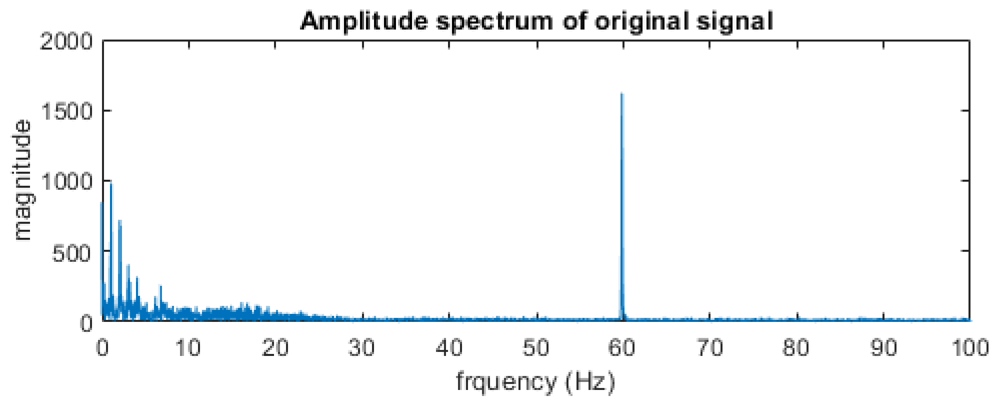
Design a filter using pole-zero placement that will remove the low and high frequency noise. Try to make the order of your filter as low as possible. Show the magnitude response and the output of your filter. What is the order of your filter?

```
clear all; close all; clc;
```

Filter Applications

```
ecg = load('ecg.sig');
Fs = 200; %signal is sampled at 200 Hz
N = length(ecg);
T = 1;
ts = 1/Fs;
t = (0:N-1)*ts;

%Plotting a portion of original ECG Signal vs. Time
subplot(2,1,1);
plot(t, ecg);
title('Original ECG signal vs. Time');
xlabel('Time (Sec)');
ylabel('ECG (mV)');
axis([0 3 -1.5 1.5]);

% fft of the Original signal
Y = fft(ecg);
subplot(2,1,2);
f = linspace(0,Fs,N);
plot(f, abs(Y));
title('Amplitude spectrum of original signal');
xlabel('frquency (Hz)'); ylabel('magnitude');
axis([0 100 0 2000]);
```





Pole-zero Placement Method

It can be seen that there is a noise at 60 Hz. That is where the zero must be placed.

In radian form, $\omega_c = 2\pi \dfrac{60}{200} = 0.60\pi$

Zero

$1 \angle \pm 0.60\pi \rightarrow -0.309 \pm 0.951i;$

Poles

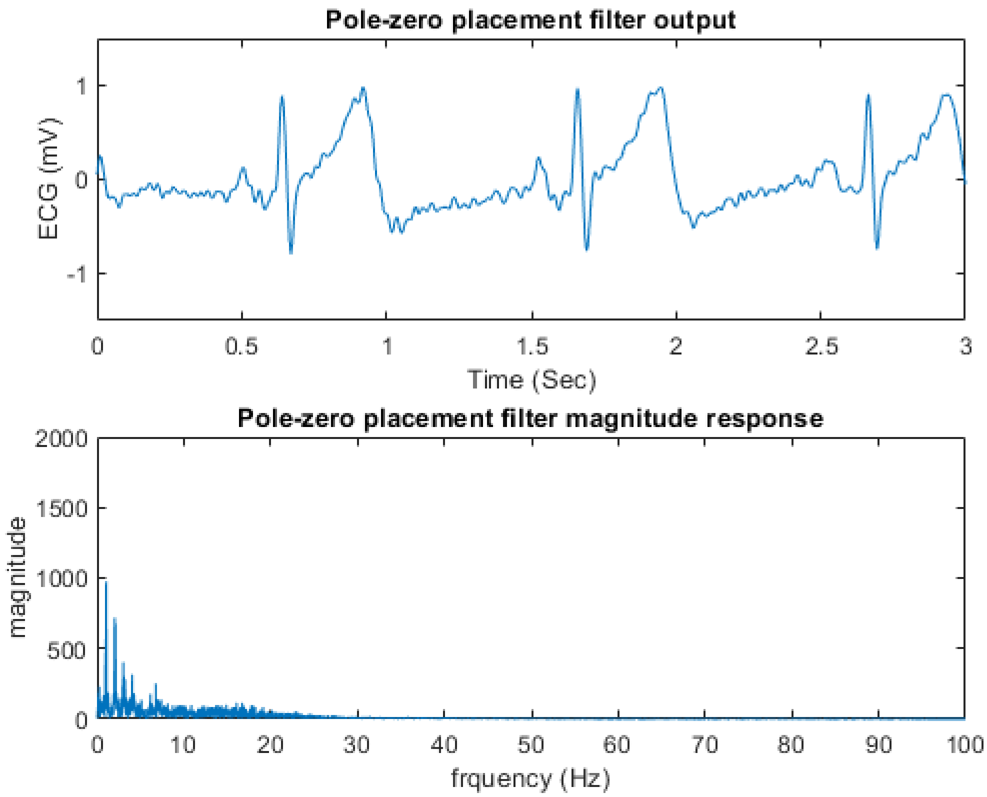
$0.9 \angle \pm 0.50\pi \rightarrow 0.0 \pm 0.90i;$
$0.5 \angle \pm 1.50\pi \rightarrow 0.0 \pm 0.50i$

```
%Notch filter creates a small region of frequencies to be removed
wc = 0.60*pi; % cut0ff frequency

z = [cos(wc)+1i*sin(wc) cos(-wc)+1i*sin(-wc)];
p = [];
g = 0.3820; % gain
NFn = poly(z)*g;
NFd = poly(p);
```

```
%lowpass
p = [];
g = 0.25;
LFn = [1 2 1]*g;
LFd = poly(p);

%highpass
HFn = [1 -1];
HFd = [1 -0.995];

%Cascaded
Cz = conv(NFn, conv(HFn, LFn));
Cp = conv(NFd, conv(HFd, LFd))
```

```
Cp = 1×2
    1.0000   -0.9950
```

```
pzf = filter(Cz, Cp, ecg);

%Plotting a portion of filtered ECG Signal vs. Time
figure();
subplot(2,1,1);
plot(t, pzf);
title('Pole-zero placement filter output');
xlabel('Time (Sec)'); ylabel('ECG (mV)');
axis([0 3 -1.5 1.5]);

% fft of the filtered signal
Y = fft(pzf);
subplot(2,1,2);
f = linspace(0,Fs,N);
plot(f, abs(Y));
title('Pole-zero placement filter magnitude response');
xlabel('frquency (Hz)');
ylabel('magnitude');
axis([0 100 0 2000]);
```



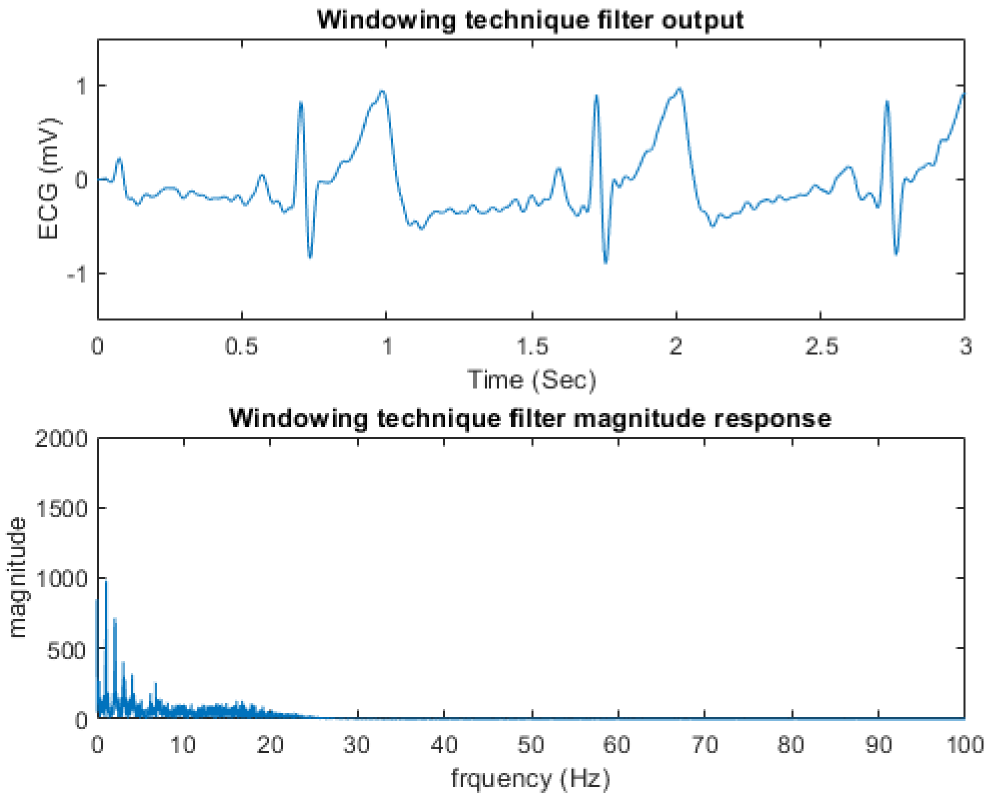

```
%The order of the filter is 2
```

Windowing Tehcnique

$\mathrm{Wc} = 0.24\pi$

```
%Hamming window is used
hHam = compHd(0.24*pi, 31).*transpose(hamming(31));
[magHam31, wHam31] = freqzM(hHam, [1]);
y = filter(hHam, [1], ecg);

%Plotting a portion of filtered ECG Signal vs. Time
figure();
subplot(2,1,1);
plot(t, y);
title('Windowing technique filter output');
xlabel('Time (Sec)');
ylabel('ECG (mV)');
axis([0 3 -1.5 1.5]);

Y = fft(y);
subplot(2,1,2);
f = linspace(0,Fs,N);
```

```
plot(f, abs(Y));
title('Windowing technique filter magnitude response');
xlabel('frquency (Hz)');
ylabel('magnitude');
axis([0 100 0 2000]);
```




```
filtord(hHam, [1])
```

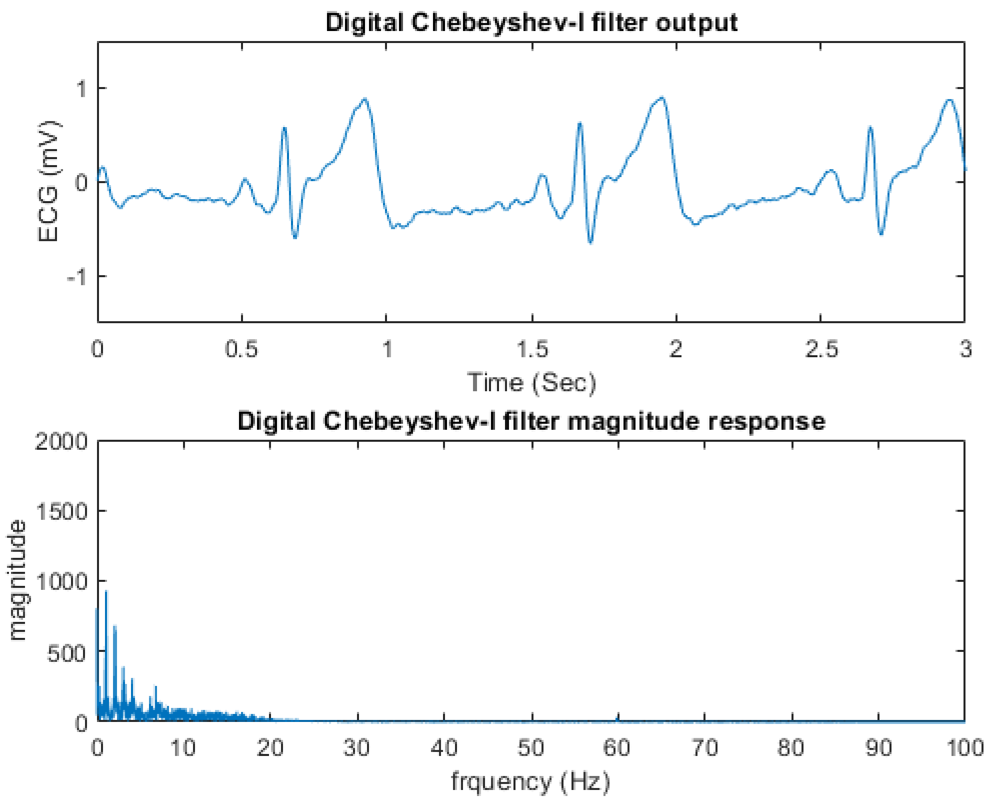
```
ans = 30
```

```
%The order of the filter is 30.
```

Digital Chebeyshev-I

```
[nCheb1, OmegaC] = cheb1ord(20/200, 65/200, 0.5, 10);
[Bz, Az] = cheby1(nCheb1, 0.5 , OmegaC);
[Hz, Wz] = freqzM(Bz, Az);
y = filter(Bz, Az, ecg);

%Plotting a portion of filtered ECG Signal vs. Time
figure();
subplot(2,1,1);
plot(t, y);
title('Digital Chebeyshev-I filter output');
```

```
xlabel('Time (Sec)');
ylabel('ECG (mV)');
axis([0 3 -1.5 1.5]);

Y = fft(y);
subplot(2,1,2);
f = linspace(0,Fs,N);
plot(f, abs(Y));
title('Digital Chebeyshev-I filter magnitude response');
xlabel('frquency (Hz)');
ylabel('magnitude');
axis([0 100 0 2000]);
```




```
nCheb1
```

```
nCheb1 = 2
```

```
%The order of the filter is 2
```

freqzM() function from Ingle and Proakis

```
function [mag w] = freqzM(b,a);
    [H,w] = freqz(b,a,1000,'whole');
    H = (H(1:1:501));
    w = (w(1:1:501));
    mag = abs(H);
```

```
end
```

compHd() function from Ingle and Proakis

```
function hd = compHd(wc,M);
    alpha = (M-1)/2;
    n = [0:1:(M-1)];
    m = n-alpha;
    fc = wc/pi;
    hd = fc*sinc(fc*m);
end
```

Reference: MATLABExperiments/Filtering of the ECG for the Removal of Noise.m at c2864303b3bb5d54e789ce41d3bdd3a58550a59e · DForshner/MATLABExperiments · GitHub