EECS 3602 Design Project

Table of Contents

Submission Details

Author: Benjamin C

Date: December 1st, 2017

Introduction

This project consisted of three major sections; designing a IFIR filter, designing a FRM filter, and designing a filter for an ECG signal with two noise sources. Also included was refining the designs to meet the specifications given.

Equipment

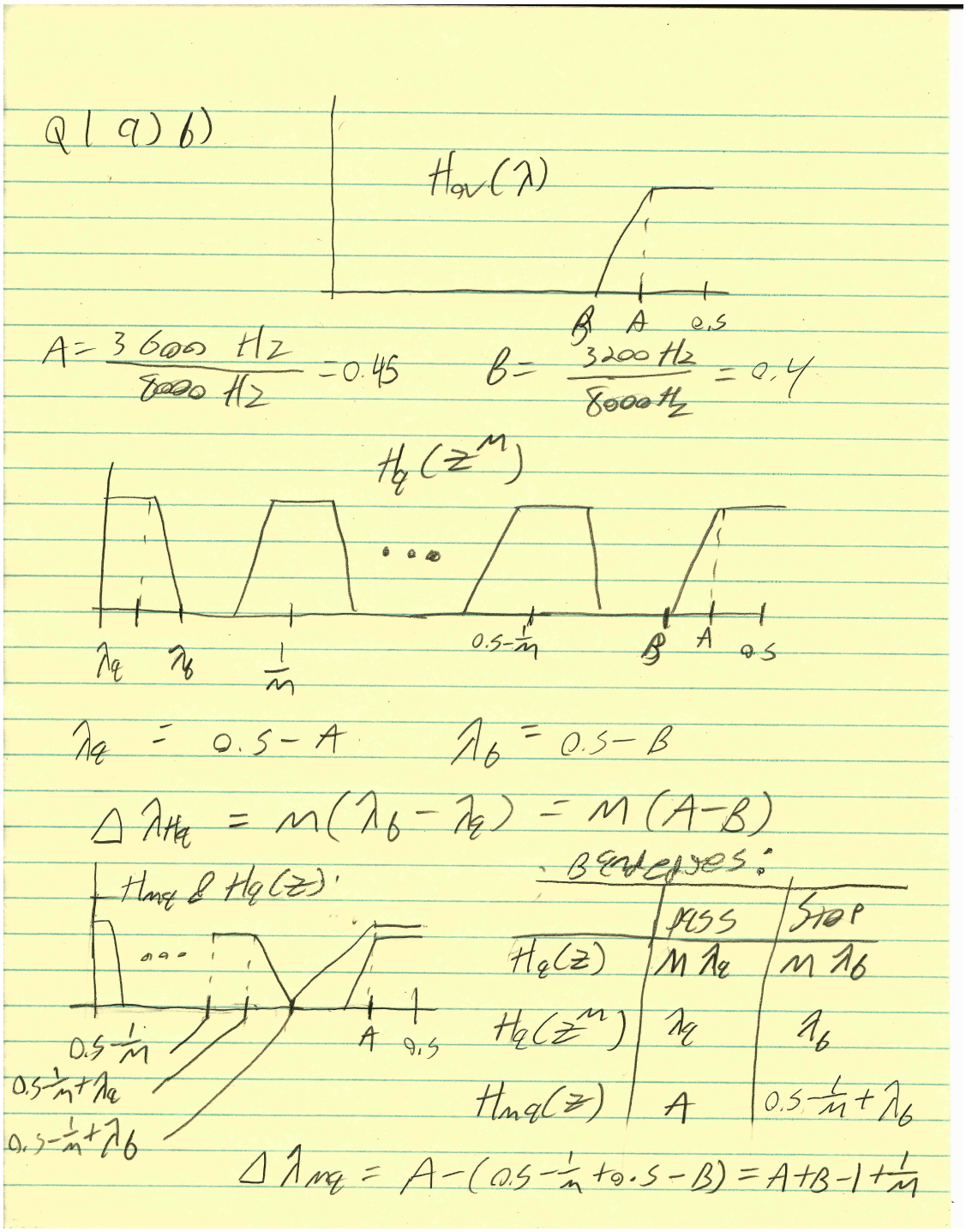
The equipment required for this project included a computer with a copy of MATLAB with its DSP toolbox.

Results And Discussion

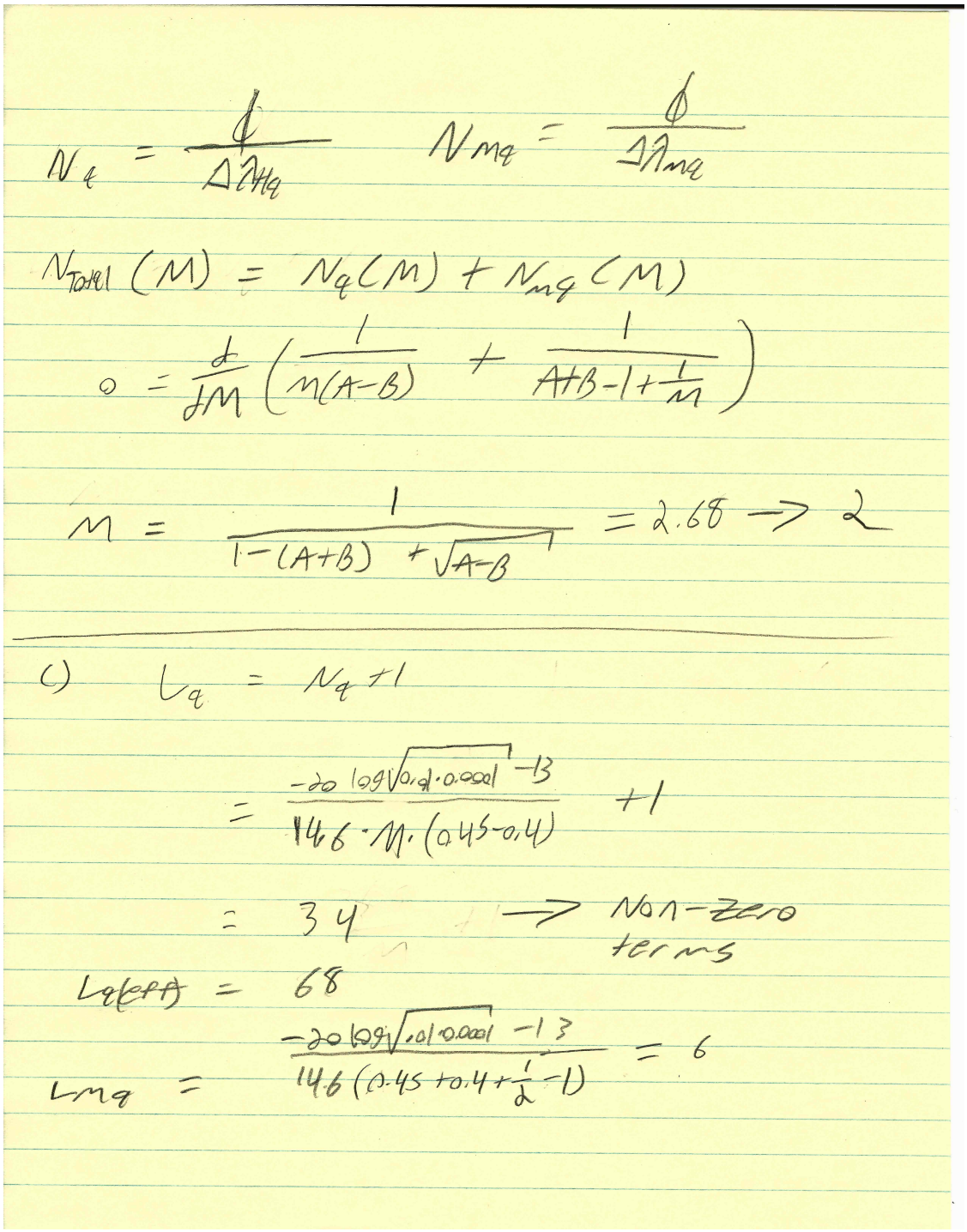
The results and discussion portion of this lab is included in the section specific to the question.

Q1 a, b, & c

The two given bands tell us that this will be a high pass filter, we can continue to calculate the bandedges and the optimal M.



On this page, the bandages are calculated and placed in a table (bottom right)



$$N_q = \frac{\phi}{\Delta\lambda_{Hq}} \qquad N_{mq} = \frac{\phi}{\Delta\lambda_{mq}}$$

$$N_{Total}(M) = N_q(M) + N_{mq}(M)$$

$$0 = \frac{d}{dM}\left(\frac{1}{M(A-B)} + \frac{1}{A+B-1+\frac{1}{M}}\right)$$

$$M = \frac{1}{1-(A+B)+\sqrt{A-B}} = 2.68 \rightarrow 2$$

c) $L_q = N_q + 1$

$$= \frac{-20\log\sqrt{0.1\cdot 0.0001} - 13}{14.6 \cdot M \cdot (0.45-0.4)} + 1$$

$$= 34 \rightarrow \text{Non-zero terms}$$

$L_{q(left)} = 68$

$$L_{mq} = \frac{-20\log\sqrt{.1\cdot.0001} - 13}{14.6(0.45+0.4+\frac{1}{2}-1)} = 6$$

On this page, the optimal interpolation factor is calculated, and the lengths for the two sub filters are calculated

Q1 d

Using the ifir method availible in MATLAB, the two subfilters can be designed. As it turns out, the lengths calculated by hand are slightly off from what the method returns.

```
fp = 3600;
fs = 3200;
fsample = 8000;
pbr = 0.01;
sbr = 0.0001;

% We calculate the needed parameters
lb1 = fp/fsample;% lb -> lambda
lb2 = fs/fsample;
beta = lb1 - lb2;

% Then estimate the interp. length
% a, b, & c, we calculate optimal M
Mopt = 1/(1-(lb1+lb2)+sqrt(beta));
M = floor(Mopt);

M

[h,g]= ifir(2,'high',2.*[lb2 lb1], [sbr pbr]);


M =

     2
```

Q1 e

Using the data cursor tool and the zooming feature on the figure, it can be determined that the overall response has the following ripples:

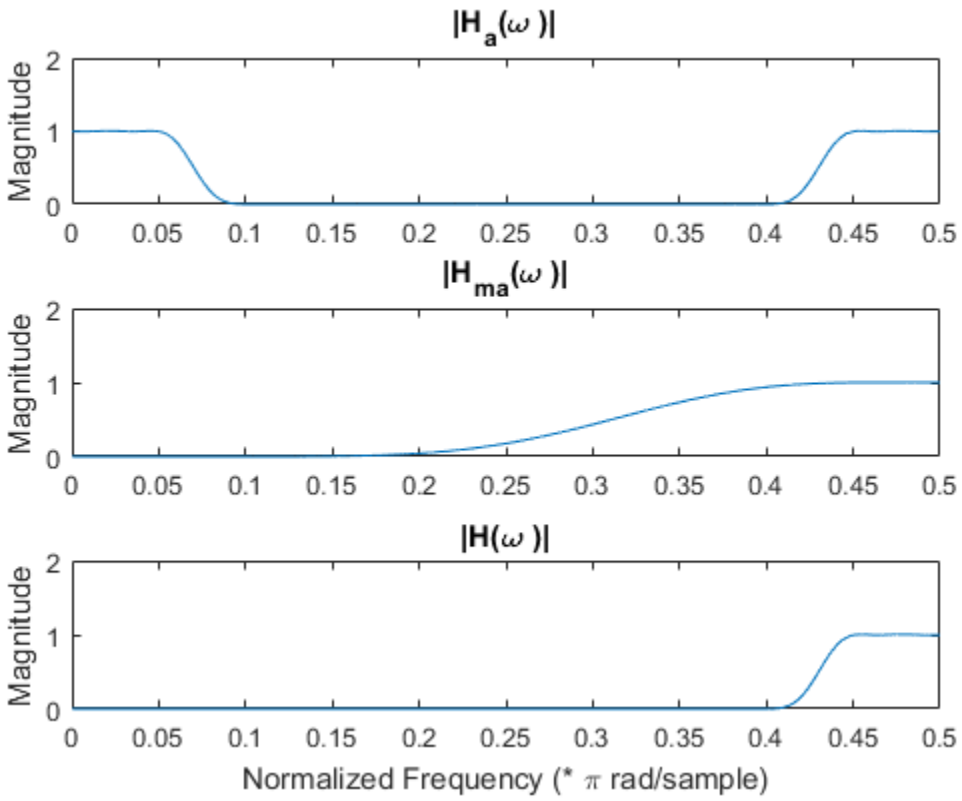
$$Passband \leq 1.004$$

$$Stopband \leq 0.00007237$$

These values lie within the acceptable range specified

```
len = 10000;
Ha = fft(h,len);
Hma = fft(g,len);

H = Ha.*Hma;

w = (1:len/2)./len;

subplot(3,1,1);
plot(w,abs(Ha(1:len/2)));
title('|H_{a}(\omega )|');
ylabel('Magnitude');

subplot(3,1,2);
plot(w,abs(Hma(1:len/2)));
title('|H_{ma}(\omega )|');
```

```
ylabel('Magnitude');

subplot(3,1,3);
plot(w,abs(H(1:len/2)));
title('|H(\omega )|');
ylabel('Magnitude');
xlabel('Normalized Frequency (* \pi rad/sample)');
```



Q2

From the specification given, the filter that needs to be designed is a low pass filter. From the lecture notes, the optimal factor is given as:

$$M_{opt} \approx \frac{1}{2\sqrt{\beta}}$$

Q2 a

Calculating the optimal interpolation factor.

```
clear
close all

pbr = 0.01;
sbr = 0.0001;
```

```
lp = 0.194;
ls = 0.2;
beta = (ls - lp);
Mopt = floor(1/(2*sqrt(beta)));

fprintf('The optimal interpolation factior is: %i', Mopt )
```

```
The optimal interpolation factior is: 6
```

Q2 b

```
function result = filter_length(rippleA, rippleB, beta)
    result = (-20.*log10(sqrt(rippleA.*rippleB)) - 13)./(14.6.*beta) +
 1;
end


% Generate values for M around the optimal
range = 2; % radius around Mopt
M = Mopt + ((-range):1:range);

% Calculate all the bandedges for a case A filter
% Case A
A_m = floor( lp.*M );
A_ltheta = lp.*M - A_m;
A_lphi = ls.*M - A_m;

A_passHa = (A_m + A_ltheta)./M;
A_stopHa = (A_m + A_lphi)./M;

A_passHma = [lp].*ones(1,range*2+1);
A_stopHma = ((A_m + 1) - A_lphi)./M;

A_passHmc = (A_m - A_ltheta)./M;
A_stopHmc = [ls].*ones(1,range*2+1);

% Determine for which M values a case A filter is possible.
CASEA = (A_ltheta > 0) & (A_ltheta < 0.5) & (A_lphi > 0) & (A_lphi <
 0.5);
CASEA = CASEA & (A_passHa > 0) & (A_stopHa < 0.5);
CASEA = CASEA & (A_passHma > 0) & (A_stopHma < 0.5);
CASEA = CASEA & (A_passHmc > 0) & (A_stopHmc < 0.5);

% Calculate all the bandedges for a case B filter
% Case B
B_m = ceil(ls*M);
B_ltheta = B_m - ls*M;
B_lphi = B_m - lp*M;

B_passHa = (B_m - B_lphi)./M;
B_stopHa = (B_m - B_ltheta)./M;
```

```
B_passHma = (B_m - 1 + B_lphi) ./ M;
B_stopHma = ls.*ones(1,range*2+1);

B_passHmc = lp.*ones(1,range*2+1);
B_stopHmc = (B_m + B_ltheta) ./ M;

% Determine for which M values a case B filter is possible.
CASEB = (B_ltheta > 0) & (B_ltheta < 0.5) & (B_lphi > 0) & (B_lphi <
 0.5);
CASEB = CASEB & (B_passHa > 0) & (B_stopHa < 0.5);
CASEB = CASEB & (B_passHma > 0) & (B_stopHma < 0.5);
CASEB = CASEB & (B_passHmc > 0) & (B_stopHmc < 0.5);

% Merge the two possibility lists
CASE = CASEA.*1 + CASEB.*2;

% Prepare the entries for the table
La = zeros(1,range);
Lma = zeros(1,range);
Lmc = zeros(1,range);
Ltotal = zeros(1,range);
passHa = zeros(1,range);
passHma = zeros(1,range);
passHmc = zeros(1,range);
stopHa = zeros(1,range);
stopHma = zeros(1,range);
stopHmc = zeros(1,range);

lowest = 1; %Assume the first entry is the lowest
for i = 1:5
    if(CASE(i) == 1)%Case A
        passHa(i) = A_ltheta(i);
        passHma(i) = A_passHma(i);
        passHmc(i) = A_passHmc(i);

        stopHa(i) = A_lphi(i);
        stopHma(i) = A_stopHma(i);
        stopHmc(i) = A_stopHmc(i);

        La(i) = ceil(filter_length(pbr,sbr, A_lphi(i) - A_ltheta(i)));
        Lma(i) = ceil(filter_length(pbr,sbr, (A_stopHma(i) -
 A_passHma(i))));
        Lmc(i) = ceil(filter_length(pbr,sbr, (A_stopHmc(i) -
 A_passHmc(i))));
        Ltotal(i) = La(i)+Lma(i)+Lmc(i);
    elseif(CASE(i) == 2)%Case B
        passHa(i) = B_ltheta(i);
        passHma(i) = B_passHma(i);
        passHmc(i) = B_passHmc(i);

        stopHa(i) = B_lphi(i);
        stopHma(i) = B_stopHma(i);
        stopHmc(i) = B_stopHmc(i);
```

```
        La(i) = ceil(filter_length(pbr, sbr,B_lphi(i) - B_ltheta(i)));
        Lma(i) = ceil(filter_length(pbr,sbr, (B_stopHma(i) -
 B_passHma(i))));
        Lmc(i) = ceil(filter_length(pbr,sbr, (B_stopHmc(i) -
 B_passHmc(i))));
        Ltotal(i) = La(i)+Lma(i)+Lmc(i);
    else
        passHa(i) = NaN;
        passHma(i) = NaN;
        passHmc(i) = NaN;

        stopHa(i) = NaN;
        stopHma(i) = NaN;
        stopHmc(i) = NaN;

        La(i) = NaN;
        Lma(i) = NaN;
        Lmc(i) = NaN;
        Ltotal(i) = NaN;
    end
    if(isnan(Ltotal(lowest)) || Ltotal(i) < Ltotal(lowest))
        lowest = i;
    end
end
```

Q2 c

From the data collected in the previous part, a table can be made and displayed to the screen. Here is the result:

M	CASE	passHa	stopHa	passHma	stopHma	passHmc	stopHmc	La	Lma	Lmc	Ltotal
4	2	0.2	0.224	0.056	0.2	0.194	0.3	136	24	32	192
5	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	1	0.164	0.2	0.194	0.3	0.13933	0.2	91	32	55	178
7	1	0.358	0.4	0.194	0.22857	0.091714	0.2	78	95	31	204
8	2	0.4	0.448	0.181	0.2	0.194	0.3	69	171	32	272

```
% Arrange the data into columns
M = M';
CASE = CASE';
passHa = passHa';
stopHa = stopHa';
passHma = passHma';
```

```
stopHma = stopHma';
passHmc = passHmc';
stopHmc = stopHmc';
La = La';
Lma = Lma';
Lmc = Lmc';
Ltotal = Ltotal';


T = table(M,CASE, passHa, stopHa, passHma, stopHma, passHmc, stopHmc,
 La, Lma, Lmc, Ltotal);
fprintf('The optimal interpolation factor is: %i\n', M(lowest));
```

The optimal interpolation factor is: 6

Q2 d

By using the matlab functions firpmord and firpm, with the bandedges the result in the lowest lengths, the desired filter can be created and tweaked by changing the order of the filters with **tweak_Nx**.

```
F=[passHa(lowest), stopHa(lowest)];
A=[1,0];
Dev=0.85.*[pbr,sbr];
fs = 1;

% These values are for part g
tweak_Na = 6;
tweak_Nma = 0;
tweak_Nmc = 2;

%Get the even order to force the length to be odd.
Na = ceil(La(lowest)/2)*2 + tweak_Na;
Nma = ceil(Lma(lowest)/2)*2 + tweak_Na;
Nmc = ceil(Lmc(lowest)/2)*2 + tweak_Na;


[N,Fi,Ai,W]=firpmord(F,A,Dev,fs);
ha = firpm(Na,Fi,Ai,W);

hc = [zeros(1, (length(ha)-1)/2),1,zeros(1, (length(ha)-1)/2)] - ha;

F=[passHma(lowest), stopHma(lowest)];
[N,Fi,Ai,W]=firpmord(F,A,Dev,fs);
hma = firpm(Nma,Fi,Ai,W);

F=[passHmc(lowest), stopHmc(lowest)];
[N,Fi,Ai,W]=firpmord(F,A,Dev,fs);
hmc = firpm(Nmc,Fi,Ai,W);

ha = upsample(ha,M(lowest));
ha = ha(1:(length(ha) - M(lowest) + 1)); % remove trailing zeros as
 they ruin symmetry
hc = upsample(hc,M(lowest));
hc = hc(1:(length(hc) - M(lowest) + 1));
```

Q2 e

This function returns the frequency response of h at all the frequencies (in radians) specified by w. h must be a symmetrical impulse.

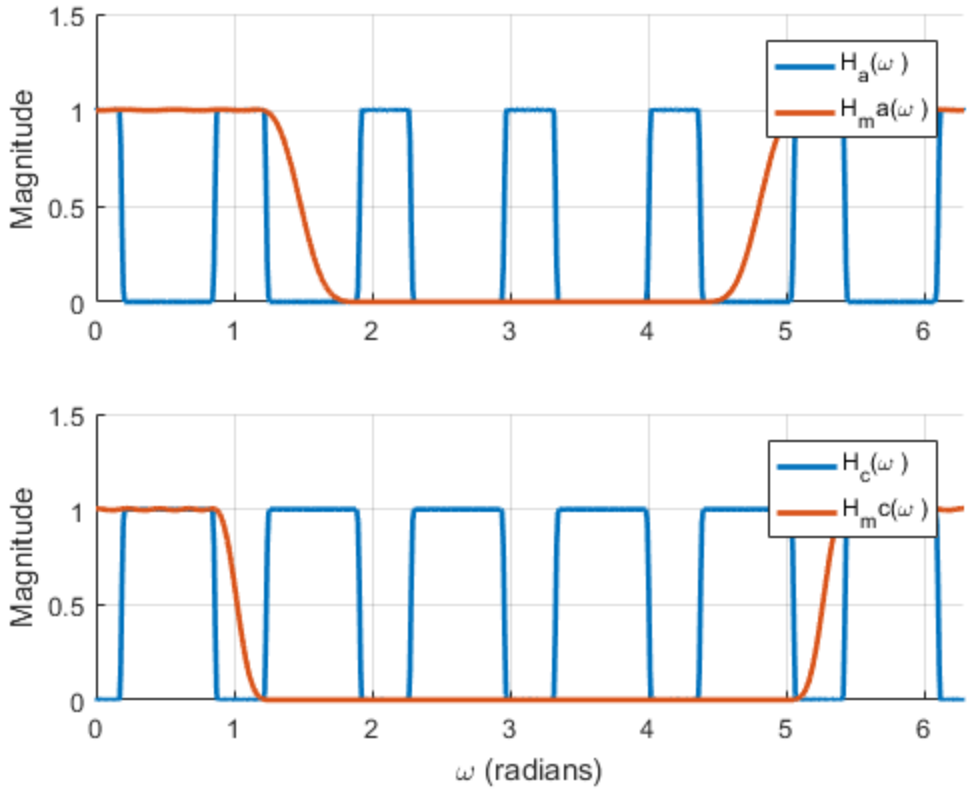
```
function H = t12_freq(h, w)
    N = length(h);

    if(bitand(N,1))
        a = zeros(1,(N - 1)/2 + 1);
        a(1) = h((N-1)/2+1);

        for i = 1:((N-1)/2)
            a(i + 1) = 2*h((N-1)/2 - i + 1);
        end

        n = 0:((N-1)/2);
        cosW = cos((n')*(w));

        H = exp(-1j.*(w)*(N-1)/2).*(a*cosW);
    else
        b = zeros(1,N/2);
        for i = 1:(N/2)
            b(i) = 2*h(N/2 - i + 1);
        end

        n = 1:(N/2);
        cosW = cos((n' - 0.5)*(w));

        H = exp(-1j.*(w)*(N-1)/2).*(b*cosW);
    end
end
```

Q2 f

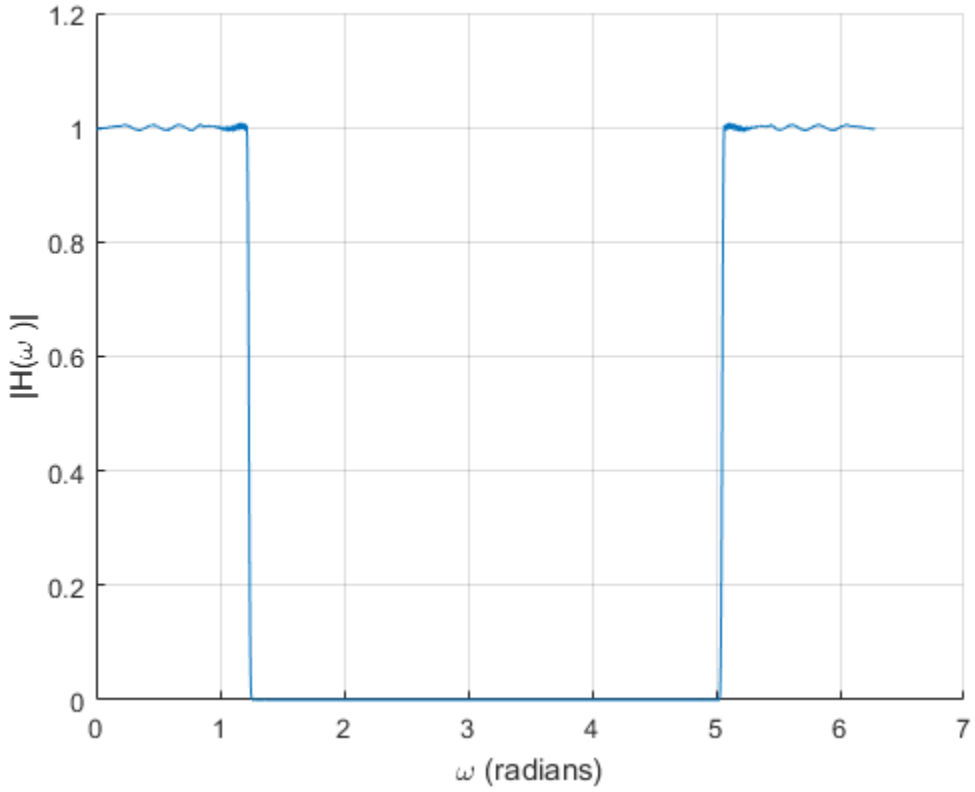
Using the function made in the previous part, the frequency response of each filter can be plotted by giving a range of omegas.

```
w = 0:1/10000:2*pi;

diff = length(hmc) - length(hma); %the number of zeros to pad

% have two branches for the filter
ubranch = conv(ha, [zeros(1,diff/2),hma, zeros(1,diff/2)]);
dbranch = conv(hc, hmc);

h = dbranch + ubranch;

%Use the function to generate the frequency response
Ha = t12_freq(ha, w);
```

```
Hc = t12_freq(hc, w);
Hma = t12_freq(hma, w);
Hmc = t12_freq(hmc, w);
H = t12_freq(h, w);

figure
subplot(2,1,1);
hold on;
grid on
plot(w,abs(Ha), 'LineWidth', 2);
plot(w,abs(Hma), 'LineWidth', 2);
xlim([0, 2*pi]);
legend('H_a(\omega )','H_ma(\omega )');
ylabel('Magnitude');


subplot(2,1,2);
hold on;
grid on
plot(w,abs(Hc), 'LineWidth', 2);
plot(w,abs(Hmc), 'LineWidth', 2);
xlim([0, 2*pi]);
legend('H_c(\omega )','H_mc(\omega )');
xlabel('\omega (radians)');
ylabel('Magnitude');


figure
hold on
grid on
plot(w,abs(H));
xlabel('\omega (radians)');
ylabel('|H(\omega )|');
```


Magnitude
1.5
1
0.5
0
0
1
2
3
4
5
6
$H_a(\omega)$
$H_ma(\omega)$

Magnitude
1.5
1
0.5
0
0
1
2
3
4
5
6
$H_c(\omega)$
$H_mc(\omega)$
ω (radians)


$|H(\omega)|$
1.2
1
0.8
0.6
0.4
0.2
0
0
1
2
3
4
5
6
7
ω (radians)

Q2 g

Displaying the table, the following image is generated

M	CASE	passHa	stopHa	passHma	stopHma	passHmc	stopHmc	La	Lma	Lmc	Ltotal
4	2	0.2	0.224	0.056	0.2	0.194	0.3	136	24	32	192
5	0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	1	0.164	0.2	0.194	0.3	0.13933	0.2	91	32	55	178
7	1	0.358	0.4	0.194	0.22857	0.091714	0.2	78	95	31	204
8	2	0.4	0.448	0.181	0.2	0.194	0.3	69	171	32	272

```
%get the complement filter to get the stopband ripple
h2 = [zeros(1, (length(h)-1)/2),1,zeros(1, (length(h)-1)/2)] - h;
H2 = t12_freq(h2, w);

fprintf('Passband ripple: %f\n', max(abs(H)) - 1);
fprintf('Stopband ripple: %f\n', max(abs(H2)) - 1);
```

```
Passband ripple: 0.008357
Stopband ripple: 0.000104
```

Q3

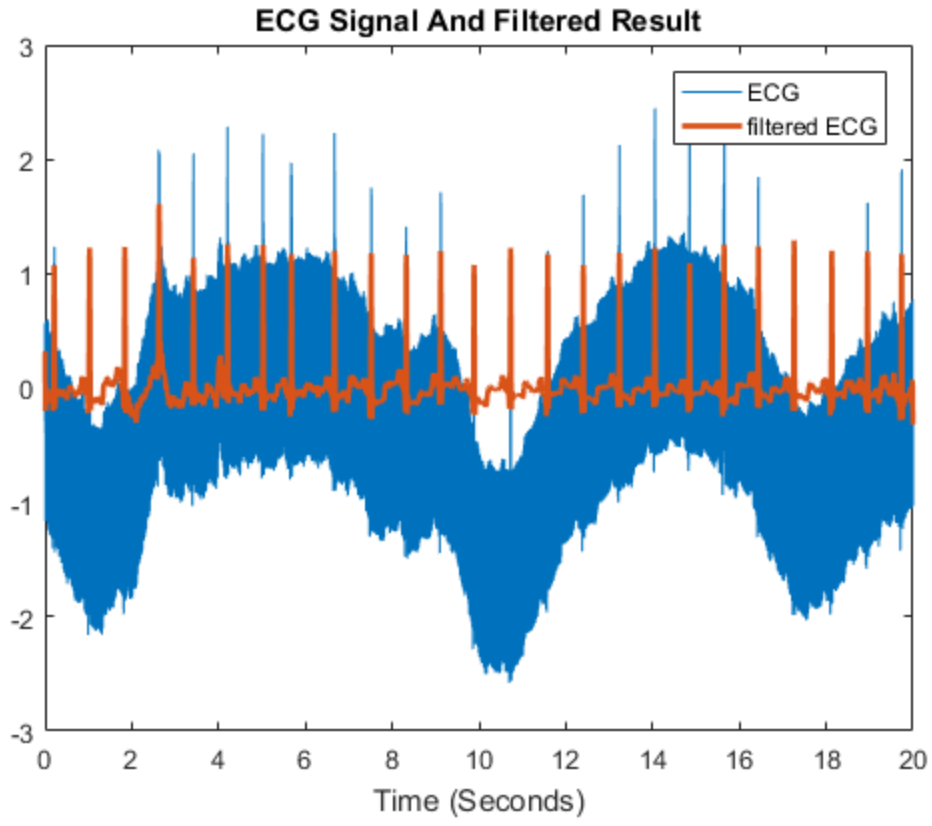
The filter is formed by two sub filters. The first is an interpolated lowpass filter for the 0.17 Hz to 1 Hz range, which is then complemented. This is done becuase an interpolated high pass only works for narrow band. The second filter is an interpolated low pass with no extra steps. The performance of the filter was also determined through the use of the MATLAB function max, and the knowlege of how to find a complementary filter.

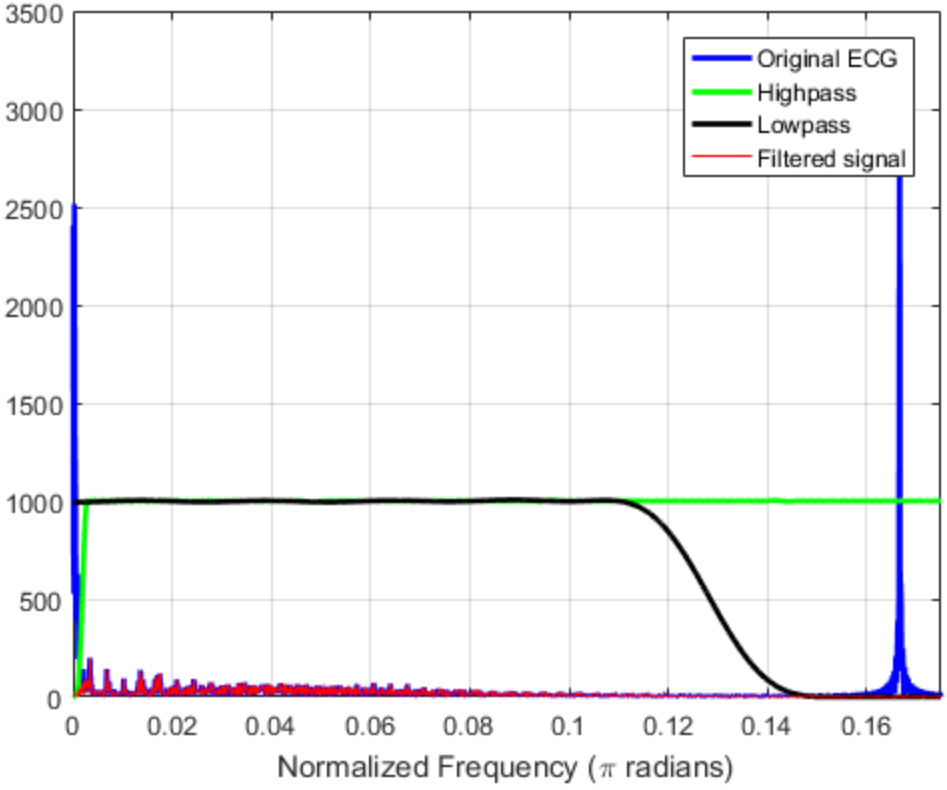
```
clear;
close all

load('ECG_signal.mat');

pbr = 0.01;
sbr = 0.001;

% By using the low pass formula for optimal M, 19.5 is calculated.
 This
% prompts us to use 19,20, and 21.  21 is the better factor.
M = 21;
```

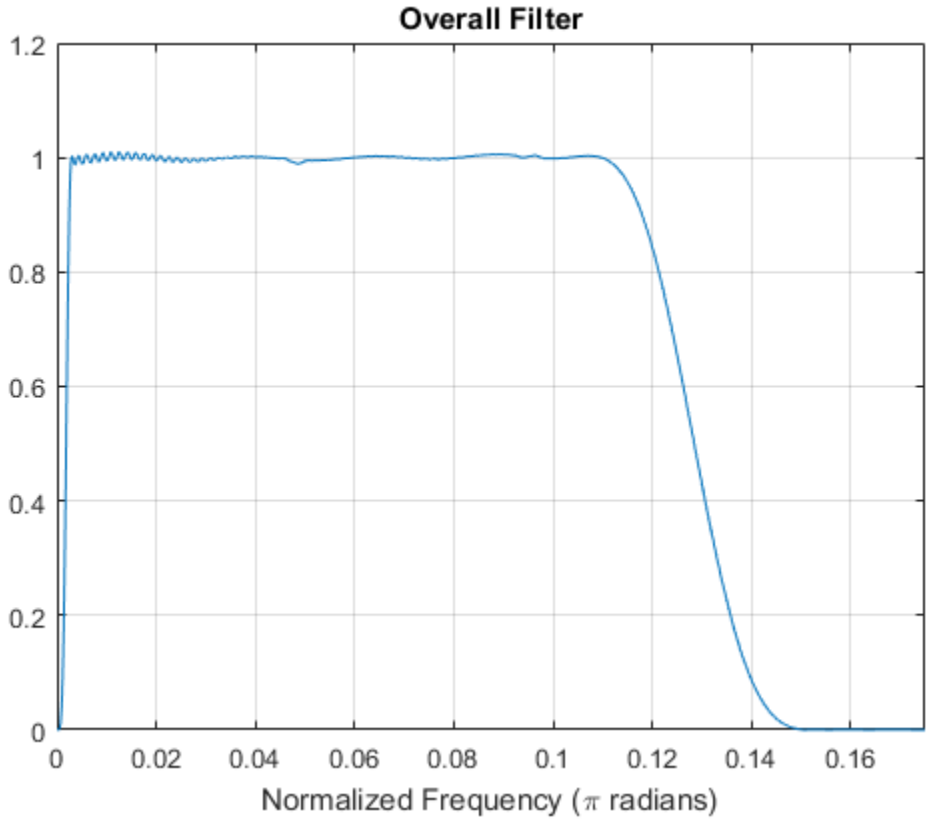
```
F=[0.17, 1];
F2=[40, 54];
A=[1,0];
Dev= 0.85.*[sbr,pbr];
fs = 360;

% Design for low, then take the complement for high
[h, g] = ifir(M, 'low', 2.*F/fs, Dev);
h = conv(h,g);
h = [zeros(1, (length(h)-1)/2),1,zeros(1, (length(h)-1)/2)] - h;

% Designing the 40/54 lowpass
Mlow = 2;
type = 'low';
dev = 0.85.*[pbr sbr];
[h2, g2] = ifir(Mlow, type, 2.*F2/fs, dev);
h2 = conv(h2,g2);

%convert all the filters and the signal to the frequency domain, apply
%convolution propery, and then convert back to time domain with the
 same
%length of the original ECG signal to see the improvement on the same
 graph
LEN = 10000;% Length to use
ECG = fft(ecg, LEN)';
H = fft(h, LEN);
H2 = fft(h2, LEN);

%Calculate the final result and remove the delay with built in matlab
%function, grpdelay.
delay = round(mean(grpdelay(conv(h,h2))));
y = conv(conv(ecg, h), h2);
y(1:delay) = [];
y((end - delay + 1):end) = [];
Y = fft(y , LEN);

%Plot everything
plot(t,ecg, 'LineWidth', 0.5);
hold on
plot(t,y, 'LineWidth', 2);
title('ECG Signal And Filtered Result')
xlabel('Time (Seconds)')
legend('ECG','filtered ECG')
figure


w = (0:(1/LEN):(1-1/LEN)); %in pi radians

plot(w,abs(ECG), 'b', 'LineWidth', 2)
hold on
plot(w,1000.*abs(H), 'g', 'LineWidth', 2)
hold on
plot(w,1000.*abs(H2), 'k', 'LineWidth', 2)
hold on
```

```
plot(w,abs(Y), 'r')
grid on
xlim([0, 0.175])
xlabel('Normalized Frequency (\pi radians)')
legend('Original ECG','Highpass','Lowpass','Filtered signal');

figure
plot(w,abs(fft(conv(h,h2),LEN)))
grid on
xlim([0, 0.175])
title('Overall Filter')
xlabel('Normalized Frequency (\pi radians)')

% Dsiplay the overall filter ripple
h_ov = conv(h,h2);
%take the complement to find the stopband ripple
h_ov_c = [zeros(1, (length(h_ov)-1)/2),1,zeros(1, (length(h_ov)-1)/2)]
 - h_ov;
H_ov = fft(h_ov, LEN);
H_ov_c = fft(h_ov_c, LEN);

fprintf('Passband ripple: %f\n', max(abs(H_ov)) - 1);
fprintf('Stopband ripple: %f\n', max(abs(H_ov_c)) - 1);
```

```
Passband ripple: 0.009352
Stopband ripple: 0.000615
```




Original ECG
Highpass
Lowpass
Filtered signal
3500
3000
2500
2000
1500
1000
500
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
Normalized Frequency (π radians)


Overall Filter
1.2
1
0.8
0.6
0.4
0.2
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
Normalized Frequency (π radians)

Conclusion

This project was fully successful. All the filters were created with a reasonable cost. During this lab, there were muliple comparisons made between FIR, IFIR, and FRM, which showed when each could be used, and the benefits achieved by using one over another. For example, the ECG filter initially used a combination of FIR and IFIR, which when converted to two IFIR filters, cut the number of multiplications in half.

Published with MATLAB® R2016b