ECE 311 Lab 4

Jacob Hutter

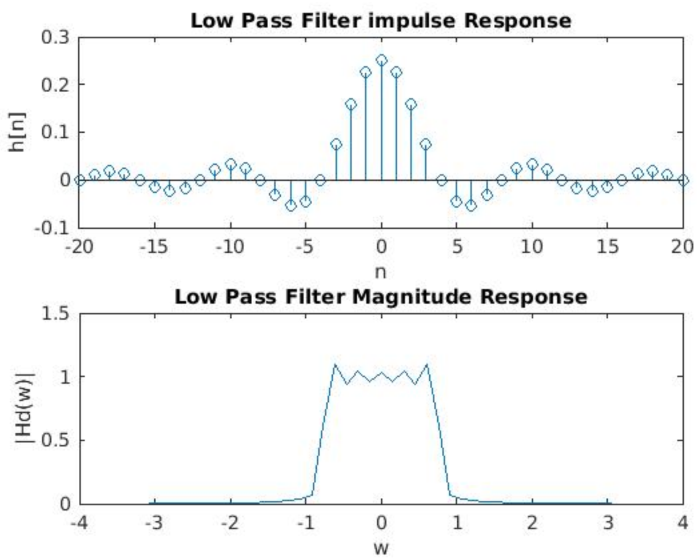
March 27, 2017

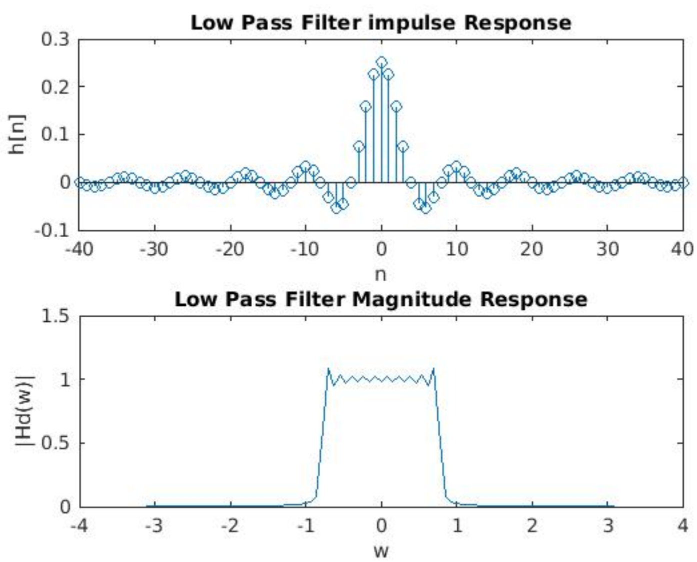
Report Item 1

```
function [  ] = filters(N,wc,w0 )
    d = zeros(1,N*2 + 1);
    d(N+1) = 1; % delta function
    n = N*2+1;
    w = fftshift((0:n-1)/n*2*pi);
    w(1:n/2) = w(1:n/2) - 2*pi;
    N = linspace(-N,N,(N*2)+1); % create -N to N array
    lpi = (wc/pi).*sinc(wc.*N./pi);
    lpm = fftshift(fft(lpi));
    hpi = d-lpi;
    hpm = fftshift(fft(hpi));
    bpi = cos(w0.*N).*lpi;
    bpm = fftshift(fft(bpi));

    figure;
    subplot(211);
    stem(N,lpi);
    title('Low Pass Filter impulse Response');
    ylabel('h[n]');
    xlabel('n');
    subplot(212);
    plot(w,abs(lpm));
    title('Low Pass Filter Magnitude Response');
    ylabel('|Hd(w)|');
    xlabel('w');

    figure;
    subplot(211);
    stem(N,hpi);
    title('High Pass Filter impulse Response');
    ylabel('h[n]');
    xlabel('n');
    subplot(212);
    plot(w,abs(hpm));
    title('High Pass Filter Magnitude Response');
    ylabel('|Hd(w)|');
    xlabel('w');

    figure;
    subplot(211);
    stem(N,bpi);
    title('Band Pass Filter impulse Response');
```

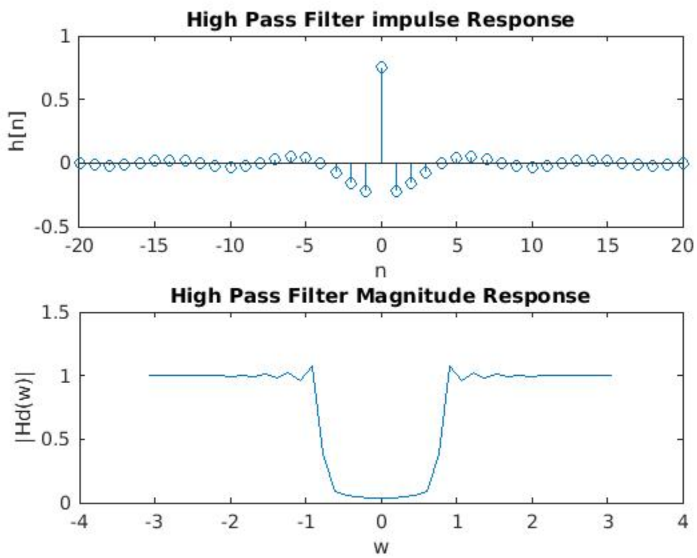
```
    ylabel('h[n]');
    xlabel('n');
    subplot(212);
    plot(w,abs(bpm));
    title('Band Pass Filter Magnitude Response');
    ylabel('|Hd(w)|');
    xlabel('w');
end
```

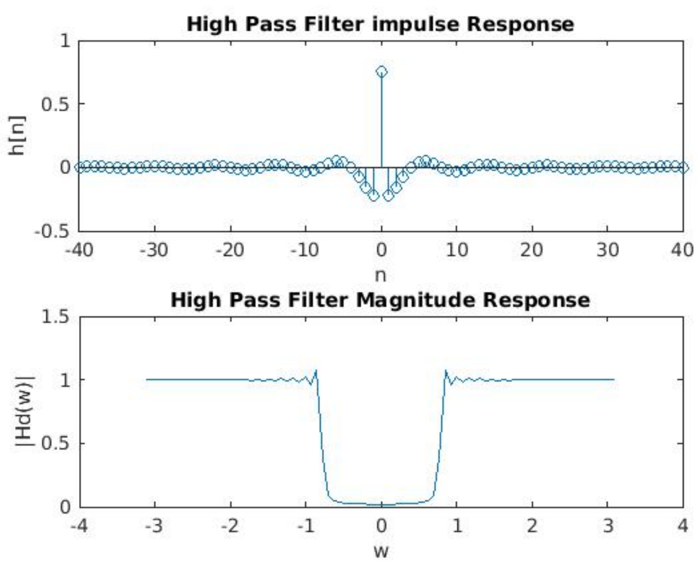
filters.m

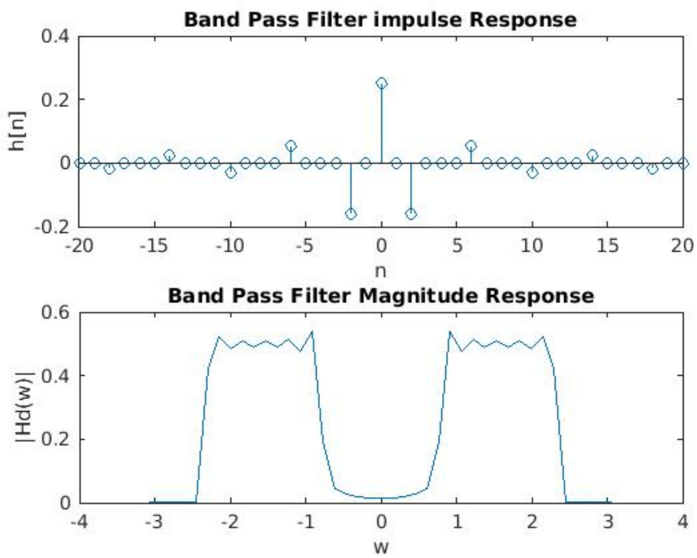


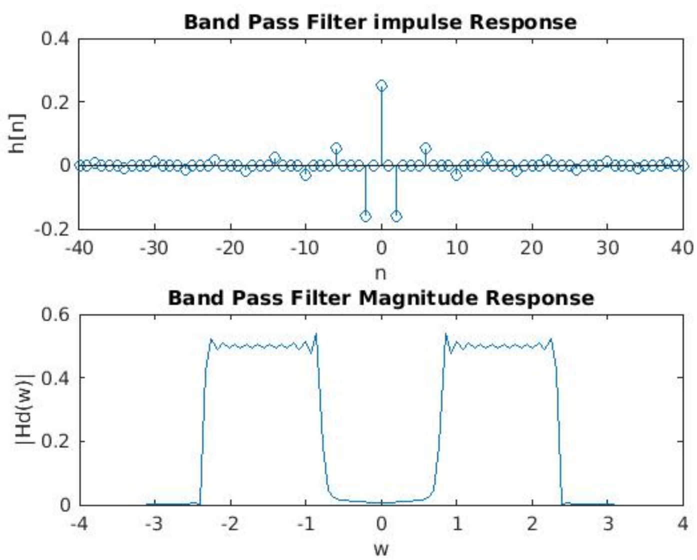


Notice that the magnitude response shows some attenuation and non-vertical edges that would not be present in an ideal model of a filter. This is not possible in a setting with a discrete amount of samples because the jump from low to high or vice versa happens instantaneously.


High Pass Filter impulse Response
h[n]
n
High Pass Filter Magnitude Response
|Hd(w)|
w


High Pass Filter impulse Response
h[n]
n
High Pass Filter Magnitude Response
|Hd(w)|
w

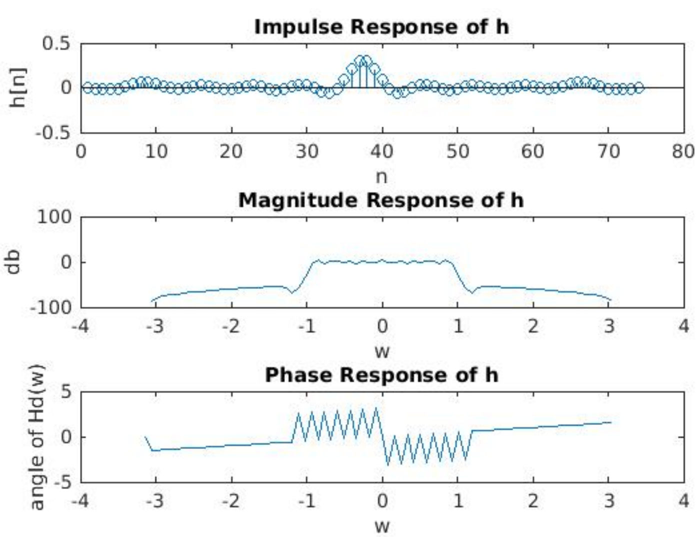

Band Pass Filter impulse Response
h[n]
n
Band Pass Filter Magnitude Response
|Hd(w)|
w


Band Pass Filter impulse Response
h[n]
n
Band Pass Filter Magnitude Response
|Hd(w)|
w

Report Item 2

```matlab
load impulseresponse.mat
% variable name is h
figure;
subplot
subplot(311);
stem(h);
n = 74;
w = fftshift((0:n-1)/n*2*pi);
w(1:n/2) = w(1:n/2) - 2*pi;
title('Impulse Response of h');
xlabel('n');
ylabel('h[n]');
subplot(312);
h_m = abs(fftshift(fft(h)));
h_m = mag2db(h_m);
plot(w,h_m);
title('Magnitude Response of h');
xlabel('w');
ylabel('db');
subplot(313);
h_p = angle(fftshift(fft(h)));
plot(w,h_p);
title('Phase Response of h');
xlabel('w');
ylabel('angle of Hd(w)');

%find pass band ripple
top = max(h_m);
bottom_range = h_m(28:48);
bottom = min(bottom_range);
passband_ripple = top - bottom;
% result is 8.0126
%passband edge is approximately .75 rad to 1.25 rad so .5 rad
```

impresp.m

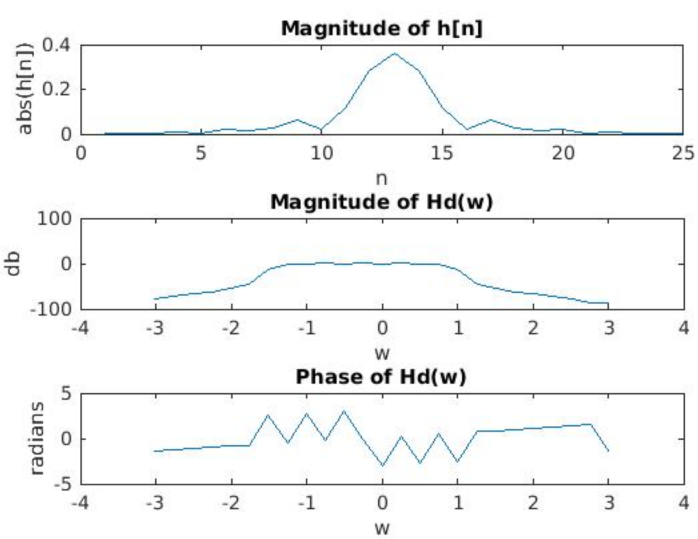


The transition bandwidth is around .5 radians wide. I calculated the maximum passband ripple(max - min) to be around 8db. The stopband ripple(max - min) seems to be around 20db.

Report Item 3

```
N = 25;
M = (N-1)/2;
w = fftshift((0:N-1)/N*2*pi); % 1. define omega as you would for
    FFT
w(1:N/2) = w(1:N/2) - 2*pi;   %
i=sqrt(-1);
for j=1:N
    if(abs(w(j)) < pi/3), % 2.
        g_w(j) = 1 * exp(-i*M*w(j));
    else
        g_w(j) = 0;
    end
end

g_n = ifft(fftshift(g_w)); % 3. find g[n], should be shifted
w_n = hamming(N)'; % window (transposed)
h_n = g_n .* w_n;% h_n is impulse response
figure;
subplot(311);
plot(abs(h_n));
title('Magnitude of h[n]');
xlabel('n');
ylabel('abs(h[n])');
subplot(312);
plot(w,mag2db(abs(fftshift(fft(h_n)))));
title('Magnitude of Hd(w)');
xlabel('w');
ylabel('db');
subplot(313);
plot(w,angle(fftshift(fft(h_n))));
title('Phase of Hd(w)');
xlabel('w');
ylabel('radians');
```

FIR_FILTER.m

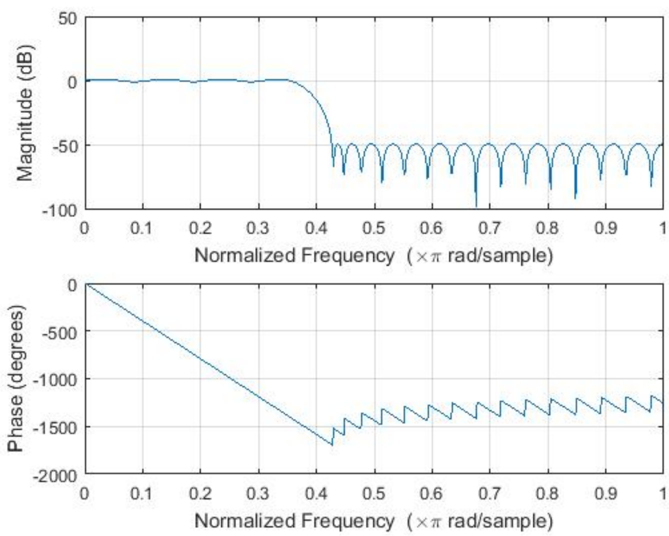


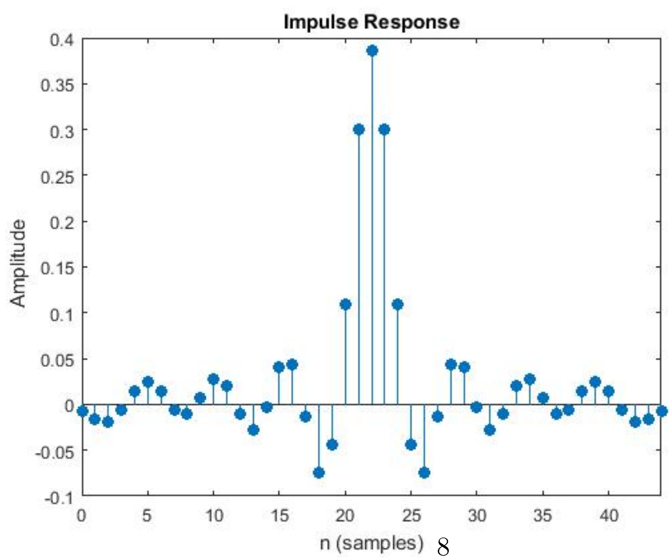
The passband ripple apears to be about 5db, stopband, about 25db. The passband edge frequency is 0.9 radians and the stopband edge frequency is about 1.2 radians.

Report Item 4

```
clc, clear all, close all
f = [54,64];
a = [1,0];
rp = [2];
rs = 50;
fs = 300;
dev = [(10^(rp/20)-1)/(10^(rp/20)+1)  10^(-rs/20)];
[n,fo,mo,w] = firpmord(f, a, dev, fs);
b = firpm(n,fo,mo,w);
freqz(b,1);
figure
impz(b,1);
```

report4.m

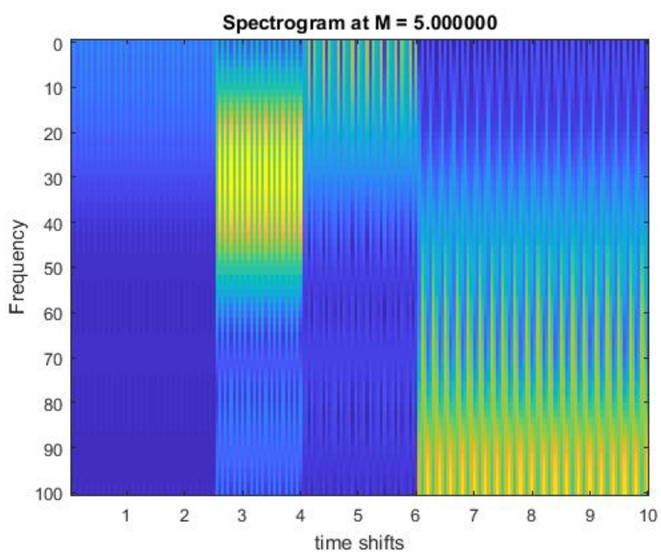


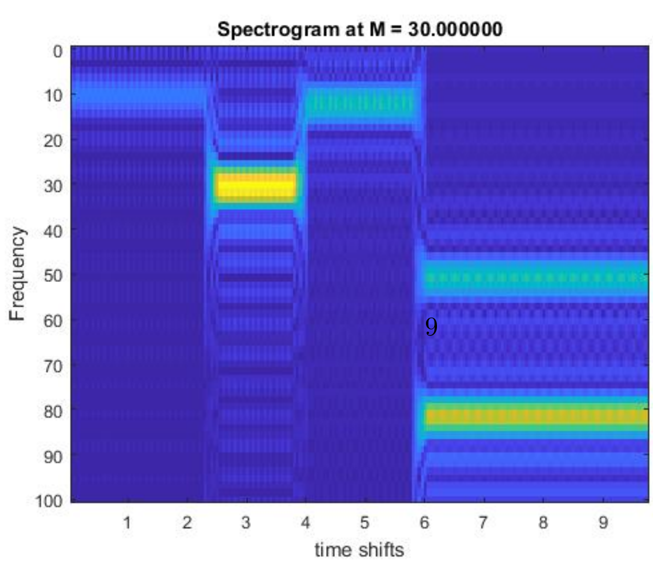


Report Item 5

```
function [a, b, c] = mySTDFT(x, M, D, P, f_s )
a_dim1 = ceil(P/2);
a_dim2 = floor(((length(x) - M)/D) + 1);
dft_mat = zeros(a_dim1,a_dim2);
time_shifts = zeros(1,a_dim2);
for i=1:a_dim2,
    time_shifts(i) = (i*D)/f_s; % vector of shifts
    time_slice = x((1+D*(i-1)):(((i-1)*D)+M)); % get time slice
    time_slice = fft(time_slice,P); % dft zero padded to P
    time_slice = time_slice(1,1:a_dim1); % only take half of P
    dft_mat(:,i) = time_slice; %assign column in dft_mat
end
    a = dft_mat; % output 1
    b = linspace(0,f_s,a_dim1); % output 2
    c = time_shifts; % output 3
    imagesc(c,b,abs(a));
    ylabel('Frequency');
    xlabel('time shifts');
    str = sprintf('Spectrogram at M = %f',M);
    title(str);
end
```

mySTDFT.m

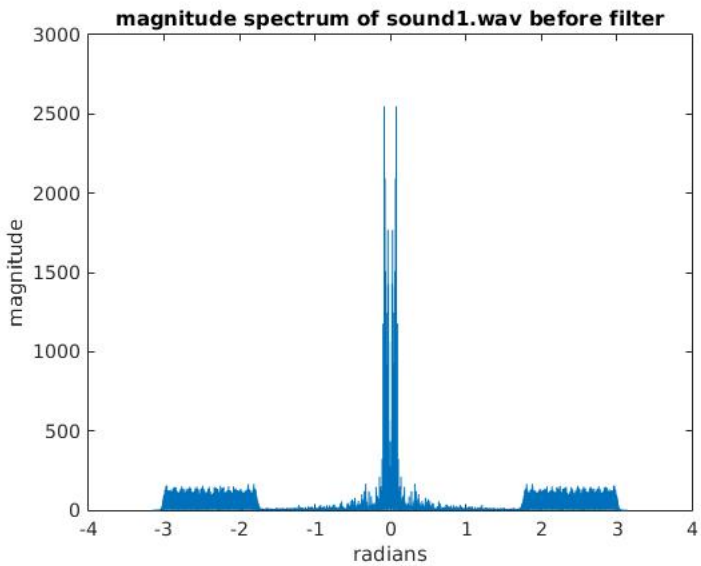


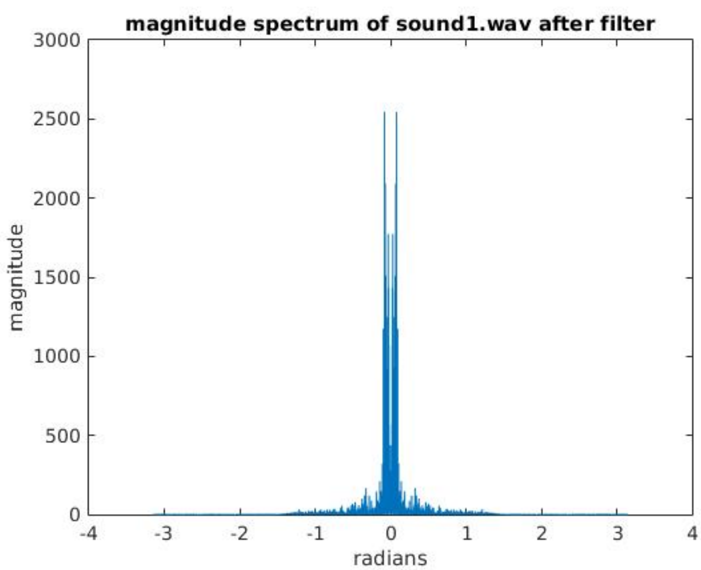


By the spectrogram we can see that(M = 5) there are frequencies ranging from 10 to 50 hz at 2.5 to 4 seconds. Then there are also 80 to 100 hz at 6 to 10 seconds. With M =30, We see frequencies 10hz at 1:2 seconds, 20hz at 2.5:4 seconds, 10hz at 4:6 seconds, and 50 and 85 hz at 6:10 seconds. When we increase M to 30 our minimum frequency to be distinguished goes down and therefore the frequency bands are more distinguishable. Note that N$\Delta t = 1/fmin, increasing N will decrease fmin.$

```

[y fs] = audioread('sound1.wav');

N = length(y);
w = fftshift((0:N-1)/N*2*pi); % define omega as you would for FFT
w(1:N/2) = w(1:N/2) - 2*pi;   %
y_w = fftshift(fft(y));
figure;
plot(w,abs(y_w));
ylabel('magnitude');
xlabel('radians');
title('magnitude spectrum of sound1.wav before filter');


f= [0 .4 .5 1];
a = [1 1 0 0];
b = firpm(50,f,a);
b_w = fftshift(fft(b,length(y)))'; % get filter

y_w = b_w .* y_w; % apply filter
figure;
plot(w,abs(y_w));
ylabel('magnitude');
xlabel('radians');
title('magnitude spectrum of sound1.wav after filter');

y = ifft(ifftshift(y_w));
soundsc(y);
filename = 'filtered1.wav';
audiowrite(filename,y,fs);
```

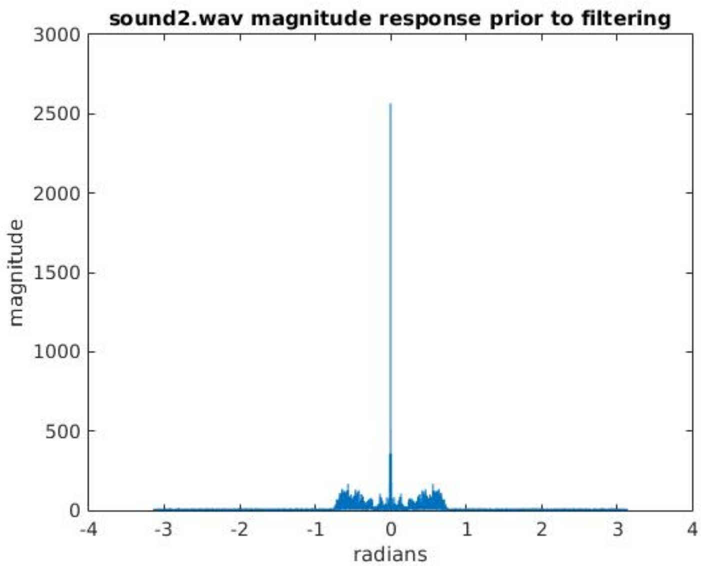
report6.m

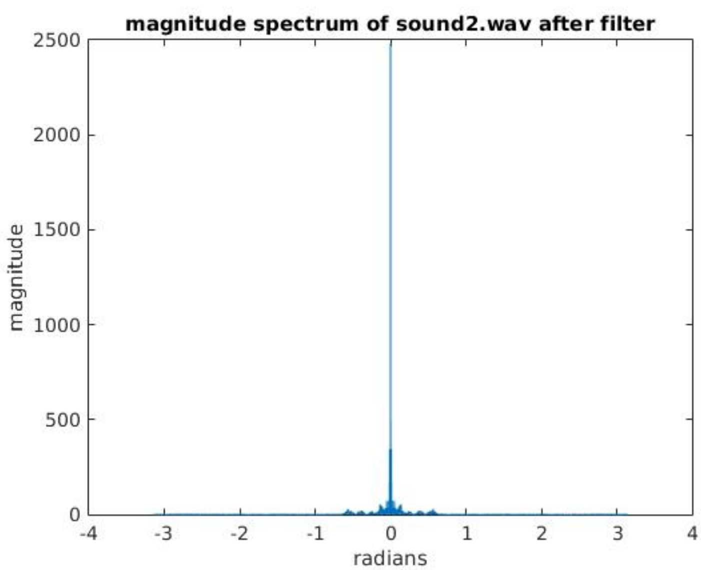

magnitude spectrum of sound1.wav before filter
magnitude
radians
3000
2500
2000
1500
1000
500
0
-4
-3
-2
-1
0
1
2
3
4


magnitude spectrum of sound1.wav after filter
magnitude
radians
3000
2500
2000
1500
1000
500
0
-4
-3
-2
-1
0
1
2
3
4

```
[y fs] = audioread('sound2.wav');
%soundsc(y);
N = length(y);
w = fftshift((0:N-1)/N*2*pi); % define omega as you would for FFT
w(1:N/2) = w(1:N/2) - 2*pi;   %

yw = fftshift(fft(y));
ys = spectrogram(y);
figure;
plot(w,abs(yw));
title('sound2.wav magnitude response prior to filtering');
xlabel('radians');
ylabel('magnitude');
figure;
imagesc(abs(ys));

f= [0 .1 .2 1];
a = [1 0 0 0];
b = firpm(50,f,a);
b_w = fftshift(fft(b,length(y)))'; % get filter

yw = b_w .* yw; % apply filter
figure;
plot(w,abs(yw));
ylabel('magnitude');
xlabel('radians');
title('magnitude spectrum of sound2.wav after filter');


y = ifft(ifftshift(yw));
ys = spectrogram(y);
figure;
imagesc(abs(ys));
soundsc(y);
filename = 'filtered2.wav';
audiowrite(filename,y,fs);
```

report7.m


sound2.wav magnitude response prior to filtering
magnitude
radians
3000
2500
2000
1500
1000
500
0
-4
-3
-2
-1
0
1
2
3
4


magnitude spectrum of sound2.wav after filter
magnitude
radians
2500
2000
1500
1000
500
0
-4
-3
-2
-1
0
1
2
3
4

```
y = load('speechsig.mat');
x = y.x;
xnoise = y.xnoise;

N = length(xnoise);
w = fftshift((0:N-1)/N*2*pi); % define omega as you would for FFT
w(1:N/2) = w(1:N/2) - 2*pi;   %

xnoisew = fftshift(fft(xnoise));

h = hamming(N);
xnoisewh = xnoisew .* h;

xnoise = ifft(ifftshift(xnoisewh),N);
%soundsc(real(xnoise));

r = zeros(N,1);
r(2000:4999,1) = rectwin(3000);

xnoisewr = xnoisew .* r;
plot(abs(xnoisew));
xnoise = ifft(ifftshift(xnoisewr));


%soundsc(real(xnoise));
```

report8.m