**Part 3 (a)**



**The input to this system is the audio file provided in the project for our group (2). In order to plot the magnitude spectrum of the output, a Fast Fourier Transform needed to be performed on the output. The function fftshift was used to match the frequencies in w. Finally, the magnitude was plotted on both a regular and dB scale. The following page contains the graphs of these plots, with the noise, speech, and tone areas labelled; specifically, the noise in the graphs are from approximately -1.0 to -0.5 and 0.5 to 1.0. The tone can be seen as spikes at around 0.2 and -0.2, with another tone being seen around 0 on the dB graph. Finally, the speech signal is between the two noise areas.**

**Matlab Script:**

%Part A

clear

load group2.mat

Fs = 10000; % Sampling Frequency

N = 2^nextpow2(length(y)); % Length

k=0:N-1;

w=2\*pi\*k./N; % Frequency samples

w=w-pi;

Y = fft(y,N); % Fast Fourier Transform

Y=fftshift(Y); % Shift FFT

Magnitude=abs(Y); % Magnitude

Ydb = mag2db(abs(Y)); % Convert to DB

figure(1);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(a): Magnitude Response of Y(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

text(-0.9,1000,'Noise')

text(-0.42,1000,'Tone')

text(0.7,1000,'Noise')

text(0.3,1000,'Tone')

text(-0.1,1000,'Speech')

subplot(2,1,2)

plot(w./pi,Ydb);

title('Part 3(a): Magnitude Response (dB) of Y(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude(dB)')

text(-0.9,-200,'Noise')

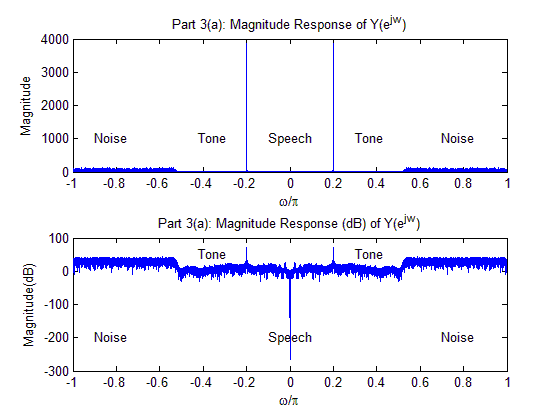
text(-0.42,50,'Tone')

text(0.7,-200,'Noise')

text(0.3,50,'Tone')

text(-0.1,-200,'Speech')

**Part 3 (a) Graphs:**

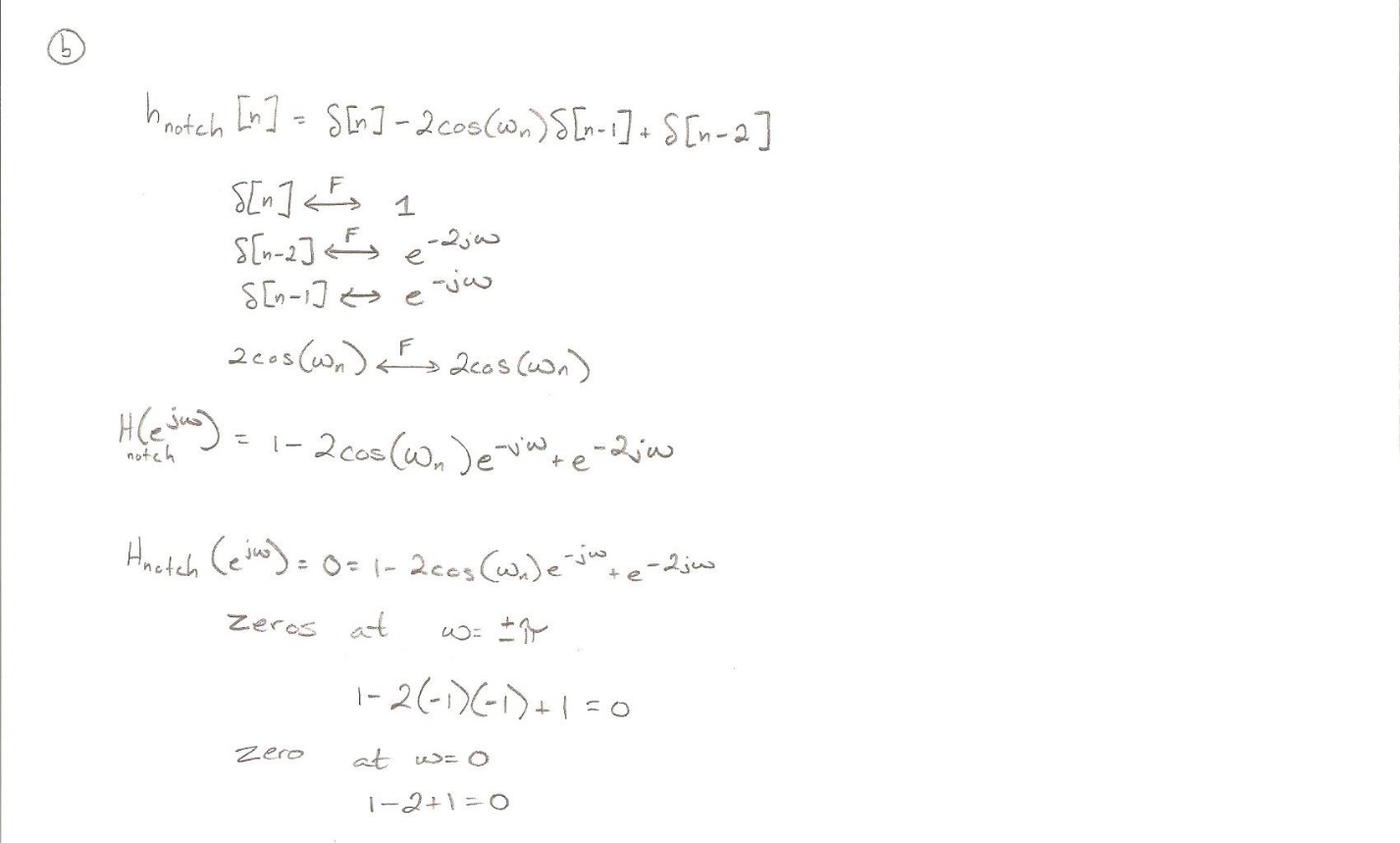
****

**Part 3 (b)**



**The following scan shows Hnotch(ejw) being found analytically. The Fourier Transform of the impulse response was taken and the resulting signal was the frequency response. The zeros were found to occur at +/- π and at 0.**

**Analytical Part 3 (b):**

****

**Part 3 (c)**



**In this part, a vector needed to be created to use the zeros that were previously found analytically. Attempting to use these roots, the tone still persisted. An additional technique was deployed here by finding the poles and zeroes and creating a notch filter from those. This notch filter was then converted into the impulse response. In order to find the notch frequency, the plots from part (a) were examined. In this plot, it is plain to see that around +/- 0.2 is where the tones occur. Converting that into frequency we get 1000Hz (0.2 \* 5000Hz = 1000Hz). The graph of the frequency response was then plotted on the accompanying graphs. This matches our data from the analytic answer by recognizing that 0.2 = π in this case. Examining the graphs we see the stop bands occur at 0.2 or 1000Hz.**

**Matlab Script:**

% Part C

% From this line until noted, partially borrowed from stackoverflow:

% http://dsp.stackexchange.com/questions/1088/filtering-50hz-using-a-notch-filter-in-matlab

% Notch frequency was found independently of this solution

f0 = 1000; % Notch Frequency

fn = Fs/2; % Nyquist frequency

freqRatio = f0/fn; % Notch/Nyquist

notchWidth = 0.1; % Width of Each Notch

% Compute zeros

zeros = [exp( sqrt(-1)\*pi\*freqRatio ), exp( -sqrt(-1)\*pi\*freqRatio )];

poles = (1-notchWidth) \* zeros; % Compute Poles

b = poly( zeros );

a = poly( poles );

% End of code borrowed

hnotch = impz(b,a); % Impulse Response

Hnotch = fft(hnotch,N);

Hnotch=fftshift(Hnotch);

HMagnitude=abs(Hnotch);

Hnotchdb = mag2db(abs(Hnotch));

figure(2);

subplot(2,1,1)

plot(w./pi,HMagnitude);

title('Part 3(c): Magnitude Response of Hnotch(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

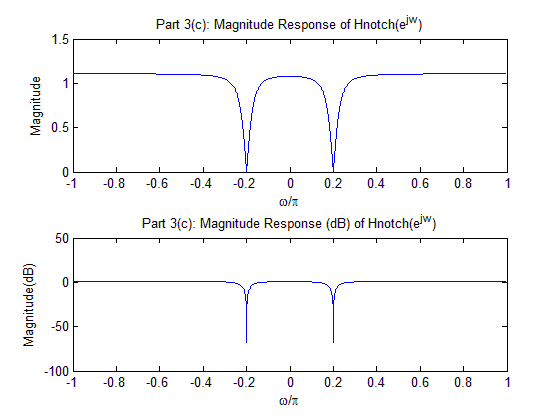
plot(w./pi,Hnotchdb);

title('Part 3(c): Magnitude Response (dB) of Hnotch(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude(dB)')

**Part 3 (c) Graphs:**

****

**Part 3 (d)**



**In this part, a filtered signal *r* was created by filtering the notch filter across the zeros. Again the FFT was taken and shifted to provide the graphs for the magnitude spectrum of R(ejw). Listening to the output signal *r*, the tone is completely removed from the signal. From the graph the tone spike can still be seen, but its magnitude is approximately 10 times lower than in the original signal and inaudible to the human ear. Changing the notch frequency to either 999 or 1001 Hz makes the magnitude larger than 400, therefore it is proven that 1000 Hz is the optimum notch frequency in this case.**

**Matlab Script:**

% Part D

r = filter(b,hnotch,y);

R = fft(r,N);

R=fftshift(R);

RMagnitude=abs(R);

Rnotchdb = mag2db(abs(R));

figure(3);

subplot(2,1,1)

plot(w./pi,RMagnitude);

title('Part 3(d): Magnitude Response of R(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

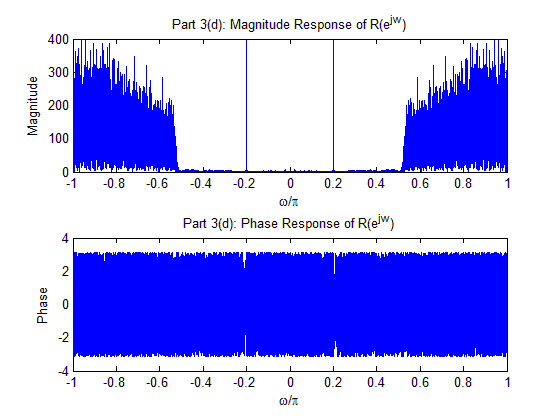
plot(w./pi,angle(R));

title('Part 3(d): Phase Response of R(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

**Part 3 (d) Graphs:**

****

**Part 3 (e)**

****

**The noise corrupting the speech signal is seen in the plot from part (a) as the sections of the plot from roughly 0.5 to 1.0 and -1.0 to -0.5; that is to say, it is the sections of the plot where there is noise appearing in that area of the graph. In order to filter out this noise, a lowpass filter needed to be used. Based on this plot and zooming in on the targeted area, a cutoff frequency of 0.42 was used. This cutoff frequency was then used as input for the fir1() function. The resulting magnitude response spectrum (real and imaginary, as well as phase) of the frequency response was plotted .The plots for these graphs are included on the following page. Examing these graphs, we can see that the passband goes from -0.42 to 0.42, thus eliminating the frequencies before -2100 Hz and after 2100Hz. This filter matches what would be expected to eliminate the noise portion that occurs before those frequencies.**

**Matlab Script:**

% Part E

alpha = 0.42; % Chosen by inspecting graph

hlpf = fir1(100, alpha);

figure(4);

H=fft(hlpf,N);

H=fftshift(H);

subplot(3,1,1)

plot(w./pi,real(H));

title('Part 3(e): Magnitude Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude (real)')

subplot(3,1,2)

plot(w./pi,imag(H));

title('Part 3(e): Magnitude Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude (imag)')

subplot(3,1,3)

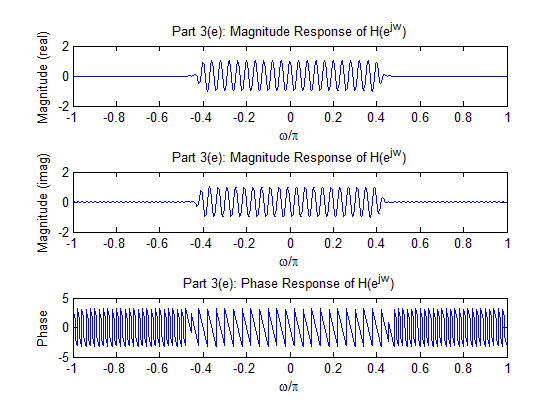
plot(w./pi,angle(H));

title('Part 3(e): Phase Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

**Part 3 (e) Graphs:**



**Part 3 (f)**



**The tone-free signal that was created in part (d) is now filtered with the result of the fir1() function and then played back. The resulting speech signal is now easier to understand. Audibly, it appears that all of the noise has been removed and additionally when the graph is inspected it can be seen that all noise is indeed removed. The resulting magnitude spectrum graphs for S(ejw) is attached on the next page.**

**Matlab Script:**

% Part F

s = filter(hlpf,1,r);

S=fft(s,N);

S=fftshift(S);

Magnitude=abs(S);

figure(5);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(f): Magnitude Response of S(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

Phase=angle(S);

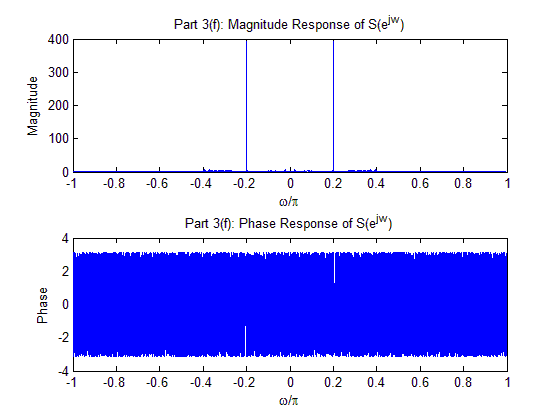
plot(w./pi,Phase)

title('Part 3(f): Phase Response of S(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

**Part 3 (f) Graphs:**



**Part 3 (g)**



**Using both hnotch and hlpf, which were created in the previous sections, a filter hcombo can be created. In order to create a single filter from two separate filters, it is simply a matter of first convolving those filters and then performing a filter on that new combo filter. The output *s2* was created and the sound was played back. Listening to *s2*, all tone and noise is absent from the signal, thus producing the desired result. Visually inspecting the graph for this filter, we can see that the stopbands occur where there was noise present in the original output signal as well as where the tones occur, therefore this filter shows that it will give the desired effect.**

**Matlab Script:**

% Part G

hnotch = impz(b,a);

hlpf = fir1(100, alpha);

hcombo = conv(hlpf,hnotch);

s2 = filter(hcombo,1,y);

soundsc(s2,Fs)

Hc=fft(hcombo,N);

Hc=fftshift(Hc);

Magnitude=abs(Hc);

figure(6);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(g): Magnitude Response of Hcombo(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

Phase=angle(Hc);

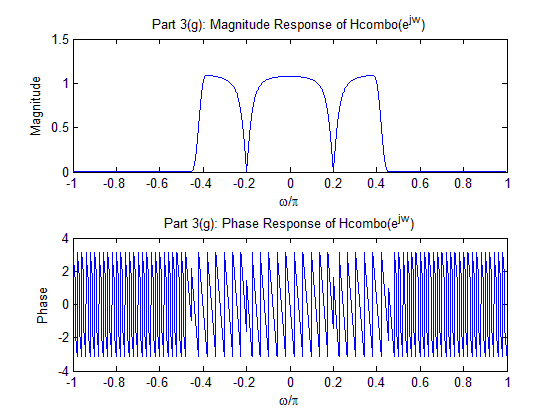
plot(w./pi,Phase)

title('Part 3(g): Phase Response of Hcombo(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

**Part 3 (g) Graphs:**



**FINAL CODE FOR PART 3:**

%Part A

clear

load group2.mat

Fs = 10000; % Sampling Frequency

N = 2^nextpow2(length(y)); % Length

k=0:N-1;

w=2\*pi\*k./N; % Frequency samples

w=w-pi;

Y = fft(y,N); % Fast Fourier Transform

Y=fftshift(Y); % Shift FFT

Magnitude=abs(Y); % Magnitude

Ydb = mag2db(abs(Y)); % Convert to DB

figure(1);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(a): Magnitude Response of Y(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

text(-0.9,1000,'Noise')

text(-0.42,1000,'Tone')

text(0.7,1000,'Noise')

text(0.3,1000,'Tone')

text(-0.1,1000,'Speech')

subplot(2,1,2)

plot(w./pi,Ydb);

title('Part 3(a): Magnitude Response (dB) of Y(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude(dB)')

text(-0.9,-200,'Noise')

text(-0.42,50,'Tone')

text(0.7,-200,'Noise')

text(0.3,50,'Tone')

text(-0.1,-200,'Speech')

% Part B is analytical

% Part C

% From this line until noted, partially borrowed from stackoverflow:

% http://dsp.stackexchange.com/questions/1088/filtering-50hz-using-a-notch-filter-in-matlab

% Notch frequency was found independently of this solution

f0 = 1000; % Notch Frequency

fn = Fs/2; % Nyquist frequency

freqRatio = f0/fn; % Notch/Nyquist

notchWidth = 0.1; % Width of Each Notch

% Compute zeros

zeros = [exp( sqrt(-1)\*pi\*freqRatio ), exp( -sqrt(-1)\*pi\*freqRatio )];

poles = (1-notchWidth) \* zeros; % Compute Poles

b = poly( zeros );

a = poly( poles );

% End of code borrowed

hnotch = impz(b,a); % Impulse Response

Hnotch = fft(hnotch,N);

Hnotch=fftshift(Hnotch);

HMagnitude=abs(Hnotch);

Hnotchdb = mag2db(abs(Hnotch));

figure(2);

subplot(2,1,1)

plot(w./pi,HMagnitude);

title('Part 3(c): Magnitude Response of Hnotch(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

plot(w./pi,Hnotchdb);

% Part D

r = filter(b,hnotch,y);

R = fft(r,N);

R=fftshift(R);

RMagnitude=abs(R);

Rnotchdb = mag2db(abs(R));

figure(3);

subplot(2,1,1)

plot(w./pi,RMagnitude);

title('Part 3(d): Magnitude Response of R(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

plot(w./pi,angle(R));

title('Part 3(d): Phase Response of R(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

% Part E

alpha = 0.42; % Chosen by inspecting graph

hlpf = fir1(100, alpha);

figure(4);

H=fft(hlpf,N);

H=fftshift(H);

subplot(3,1,1)

plot(w./pi,real(H));

title('Part 3(e): Magnitude Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude (real)')

subplot(3,1,2)

plot(w./pi,imag(H));

title('Part 3(e): Magnitude Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude (imag)')

subplot(3,1,3)

plot(w./pi,angle(H));

title('Part 3(e): Phase Response of H(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

% Part F

s = filter(hlpf,1,r);

S=fft(s,N);

S=fftshift(S);

Magnitude=abs(S);

figure(5);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(f): Magnitude Response of S(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

Phase=angle(S);

plot(w./pi,Phase)

title('Part 3(f): Phase Response of S(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')

% Part G

hnotch = impz(b,a);

hlpf = fir1(100, alpha);

hcombo = conv(hlpf,hnotch);

s2 = filter(hcombo,1,y);

soundsc(s2,Fs)

Hc=fft(hcombo,N);

Hc=fftshift(Hc);

Magnitude=abs(Hc);

figure(6);

subplot(2,1,1)

plot(w./pi,Magnitude);

title('Part 3(g): Magnitude Response of Hcombo(e^{jw})')

xlabel('\omega/\pi')

ylabel('Magnitude')

subplot(2,1,2)

Phase=angle(Hc);

plot(w./pi,Phase)

title('Part 3(g): Phase Response of Hcombo(e^{jw})')

xlabel('\omega/\pi')

ylabel('Phase')