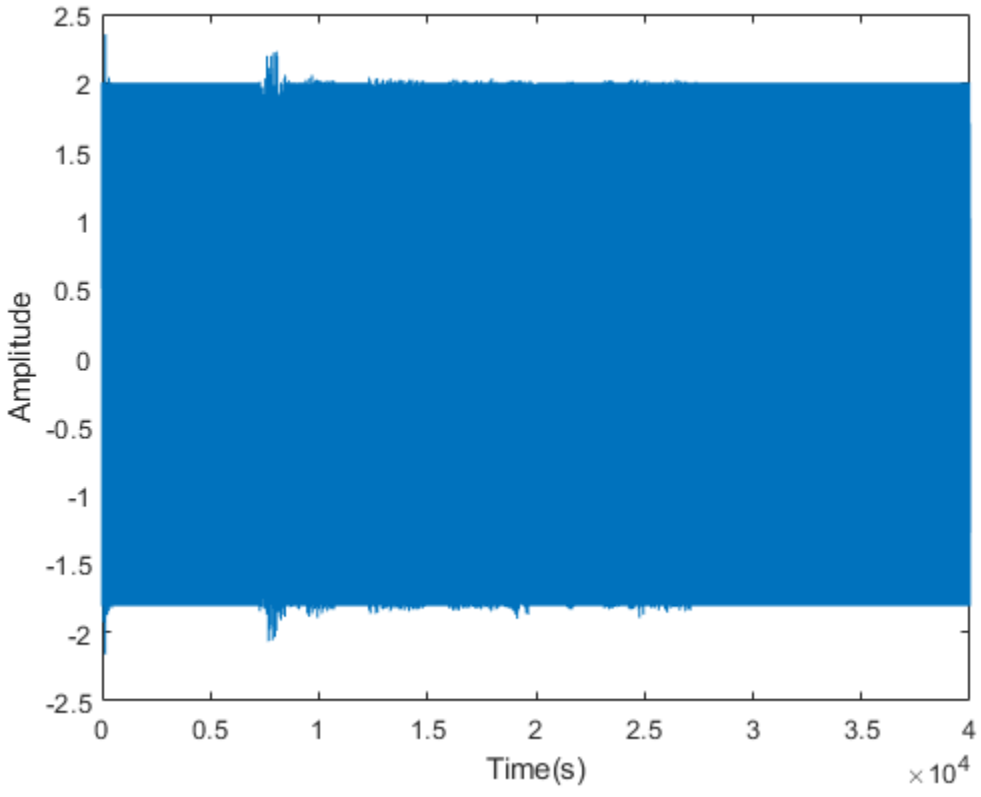
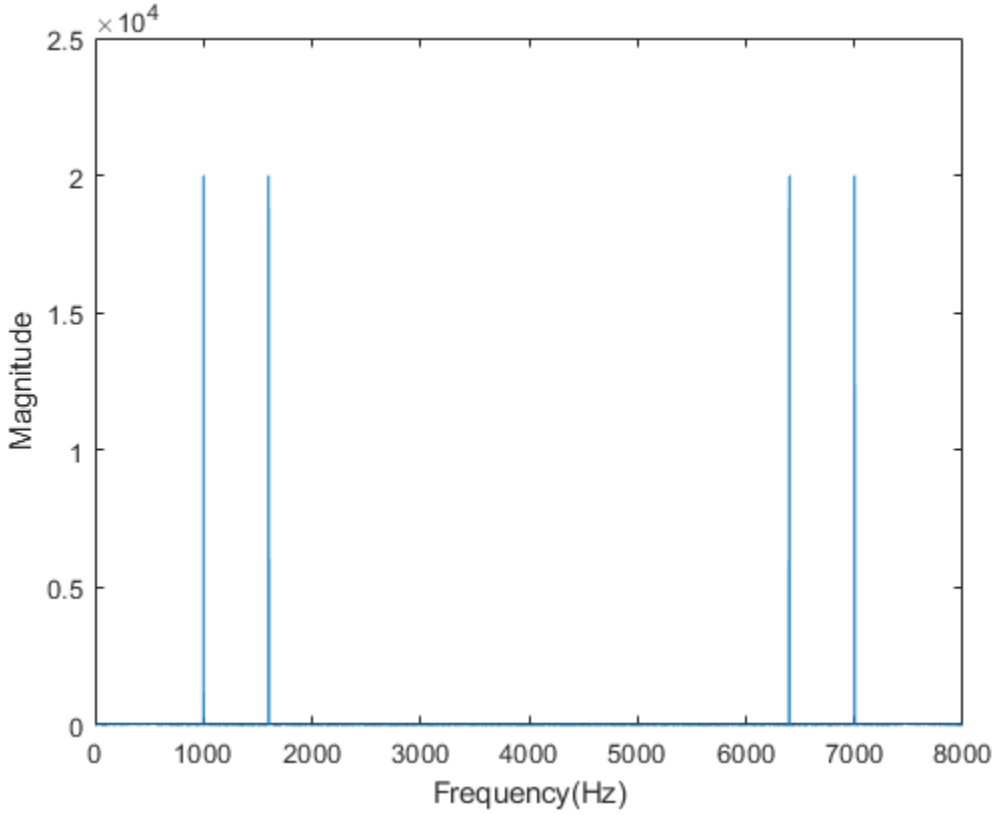
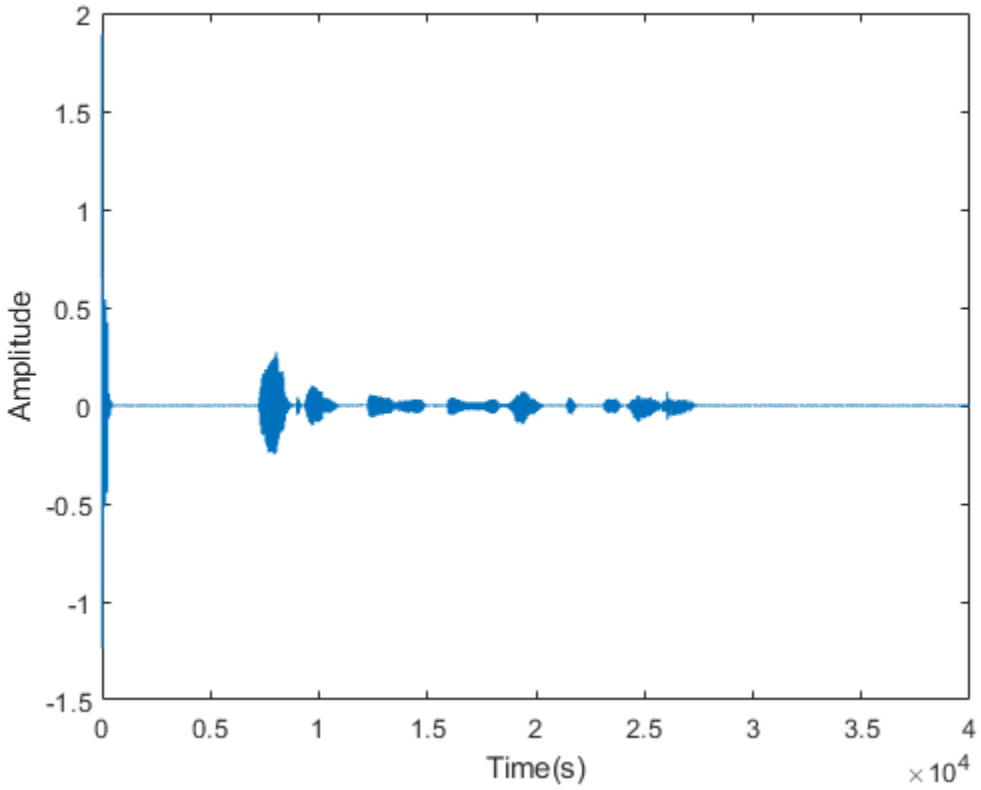
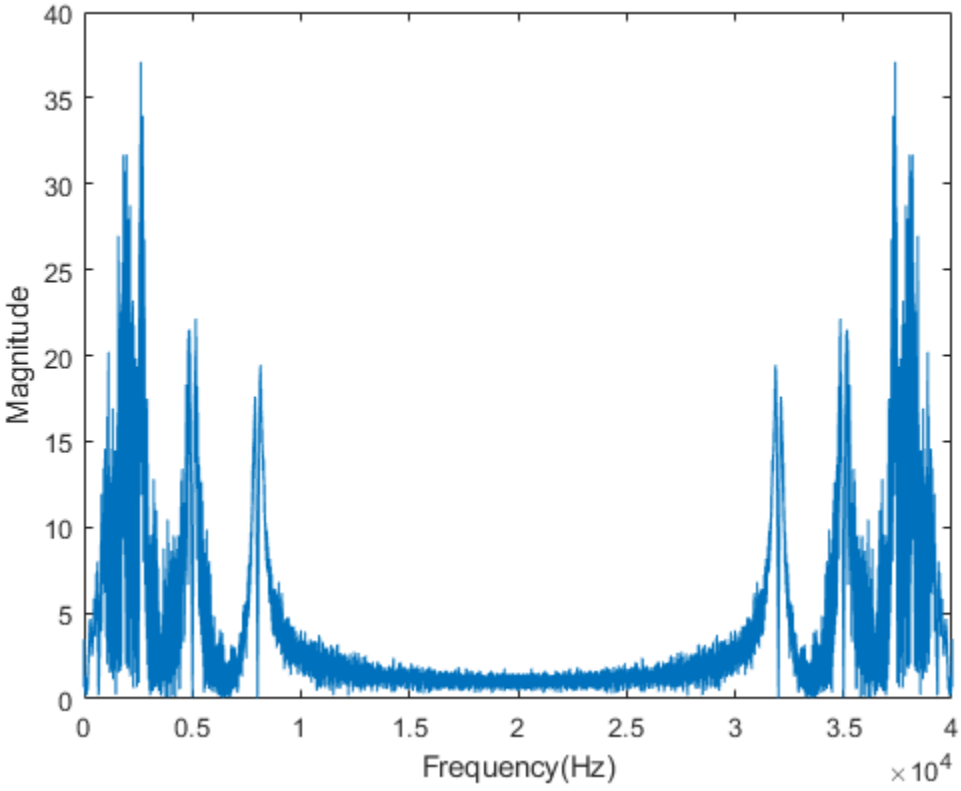
```
f_s = 8000; %8kHz sampling frequency
figure;
plot(corrupted_speech);
xlabel('Time(s)');
ylabel('Amplitude');
%Here I am plotting the amplitude of the corrupted speech signal to find
%any discrepancies in the sound file, and I see right off the bat that
%there is a lot of noise.

%sound(corrupted_speech, f_s); %Recording of the corrupted speech file
%which proves that there is a lot of noise

N = length(corrupted_speech);
speech_fft = fft(corrupted_speech);
f = (0:N-1)*(f_s/N);
figure;
plot(f, abs(speech_fft));
xlabel('Frequency(Hz)');
ylabel('Magnitude');
%At this stage, I take the Fourier Transform of this signal using the fft
%command in order to isolate the frequencies that are causing noise. In
%this case, it looks like the frequencies at which noise is occurring is at
%1kHz and 1.6kHz.

whole_spectrum = abs(speech_fft / N);
single_spectrum = whole_spectrum(1:N/2+1); %There are 2 sides of the
spectrum which are given by the FFT plot, but I only need to work with one
side so I isolate that side.
mean_val = mean(single_spectrum);
std_val = std(single_spectrum);
threshold = mean_val + 2 * std_val;
[peaks, locs] = findpeaks(single_spectrum, 'MinPeakHeight', threshold);
tone_freqs = f(locs);
%At this point, I use the threshold, which I set from the mean and standard
%deviation of the signal on one side to find the frequencies at which noise
%is being made and set them as values on the workspace.

f1 = tone_freqs(1); %1kHz
f2 = tone_freqs(2); %1.6kHz
bw = 50; %might have to adjust this value later
wn1 = [(f1-bw/2)/(f_s/2), (f1+bw/2)/(f_s/2)];
wn2 = [(f2-bw/2)/(f_s/2), (f2+bw/2)/(f_s/2)]; %naturalizing tone frequencies
to set them as inputs for for the iir filter
[b1, a1] = butter(2, wn1, 'stop');
[b2, a2] = butter(2, wn2, 'stop'); %using butterworth iir filter. might need
to change later

filtered_speech1 = filter(b1, a1, corrupted_speech);
filtered_speech = filter(b2, a2, filtered_speech1);
figure;
plot(filtered_speech)
xlabel('Time(s)');
```

```
ylabel('Amplitude');
%I have plotted the filtered speech at this point, and it is apparent that
%a lot of the noise has been filtered out. Now it's time to plot the
%frequency response.

N_filt = length(filtered_speech);
filtered_fft = fft(filtered_speech);
figure;
plot(abs(filtered_fft));
xlabel('Frequency(Hz)');
ylabel('Magnitude');
%There isn't any noise in this response.

sound(filtered_speech, f_s); %playing the audio confirms that the noise has
been filtered, and now I'm able to hear the message "Welcome to ECE161A".
```




×10^4
2.5
2
1.5
1
0.5
0
Magnitude
0
1000
2000
3000
4000
5000
6000
7000
8000
Frequency(Hz)


2
1.5
1
0.5
0
-0.5
-1
-1.5
Amplitude
0
0.5
1
1.5
2
2.5
3
3.5
4
Time(s)
×10^4



Published with MATLAB® R2024b