MONTGOMERY COLLEGE

Laboratory Assignment #3

Investigation of Speech Signals through Fourier Analysis

Ozino Odharo, Brian Ng, & Abby Estabillo
12/3/2012

Laboratory Assignment 3: Investigation of Speech Signals Through Fourier Analysis

The purpose of Laboratory 3 is for students to identify speech wave signals using MATLAB. Students are able to record sound and to save the sound unto MATLAB from the skills they gained from their previous lab which was *Signal Sampling, Manipulation and Playback Abstract*. After Laboratory Assignment 3, students should be able to identify voiced, unvoiced, and silenced portions in a speech wave graph. Second, students should be familiar with implementing mfiles into their programs. Lastly, students should be able to manipulate pre-made mfiles to match their speech signal wave by adjusting the threshold.

<u>Body</u>

In Laboratory 3, we will discuss and analyze details that are depicted from the speech graph. We will also discuss certain behaviors that occurred on the speech graphs based on the manipulation performed in the .m files.

Record and save yourself saying a short phrase in about eight seconds. Determine the appropriate threshold and automatically label voiced, unvoiced, and silenced portions.

Abby Estabillo, a group member, recorded her voice saying "She sells seashells by the seashore" for about eight seconds unto MATLAB using a *.wav* file. Abby left very short pauses between the words so that we could easily identify voiced, unvoiced, and noise in the speech graph. The gaps between the words could help us get an idea of the difference between voiced, unvoiced, and noise by looking at its amplitude.

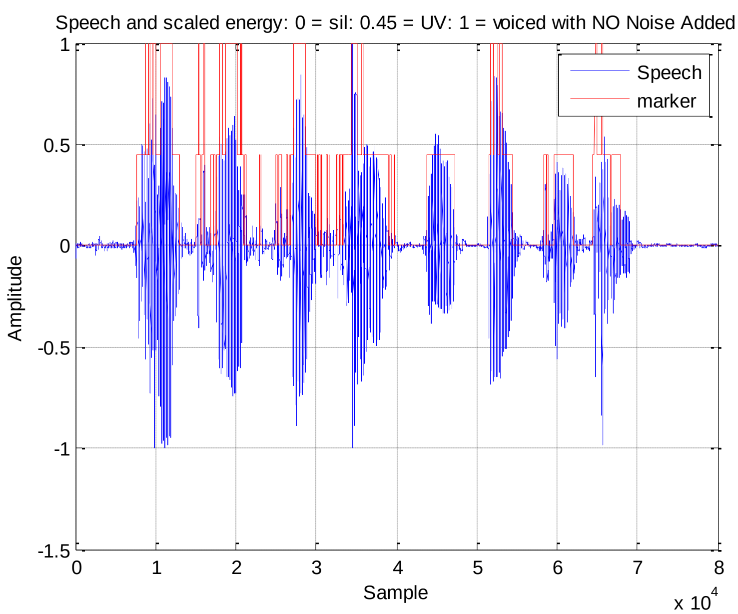

Figure 1: This is an illustration of clean speech signal of Abby's phrase obtained from MATLAB

The speech signal is behaving as we expected. For example, certain words in the recording would be defined from the amplitude as speech or un-speech. The illustration above is a clean speech of "She sells seashells by the Seashore" phrase. Our group could easily depict voiced, unvoiced, and noise by looking at the graph. Threshold is defined as the amplitude of the speech graph. To best identify voiced, unvoiced and noise from our graph, we examined graphs and group each part of the graph depending on their height of amplitude. We changed the threshold of the graph by using the *V_UV_detect.m* file provided from the course website. We changed the sil_threshold and uv_treshold number to .0045 and .045 because it gave us the best result of determining the voiced, unvoiced, and noise on the speech graph.

Add White Noise at 5dB SNR to your speech

We applied two .m files from the course website called *AddNoise.m* and *V_UV_detect*. We used both .m files in MATLAB. We added 5dB SNR to our speech graph and our results are presented below. We noticed that when we added 5dB into the original speech all the noised increased in amplitude. Now some of the unvoiced speech are buried within the noise.

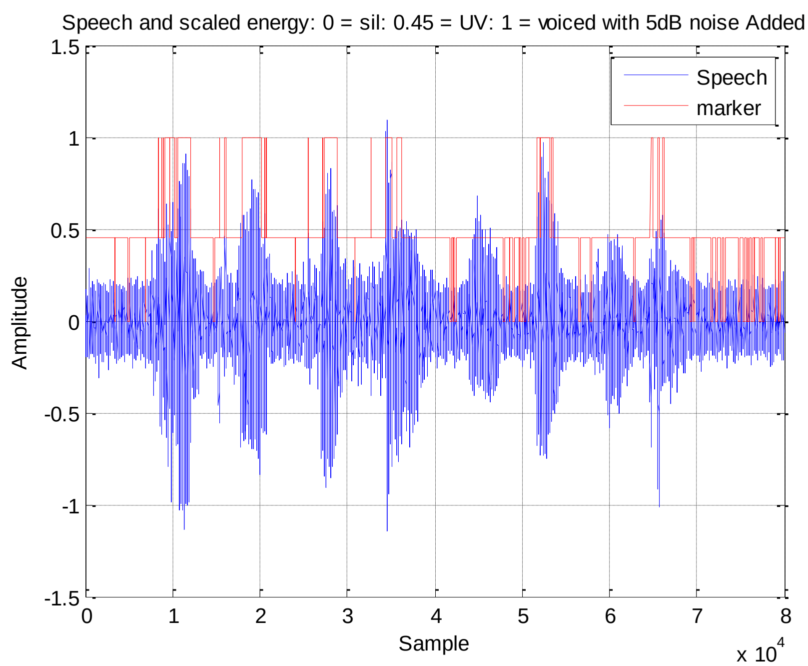

Figure 2: This is the graph of the original speech graph with 5dB SNR added.

When we added the 5dB, the SNR (signal to noise ratio), it became harder for us to detect Abby's voice. We learned that the lower the signal to noise ratio, the harder for our group to identify unvoiced speech. The speech signal graph on the left illustrates the effects adding 5 dB to the original graph. We saved the noisy audio as *noisy_speech.wav*.

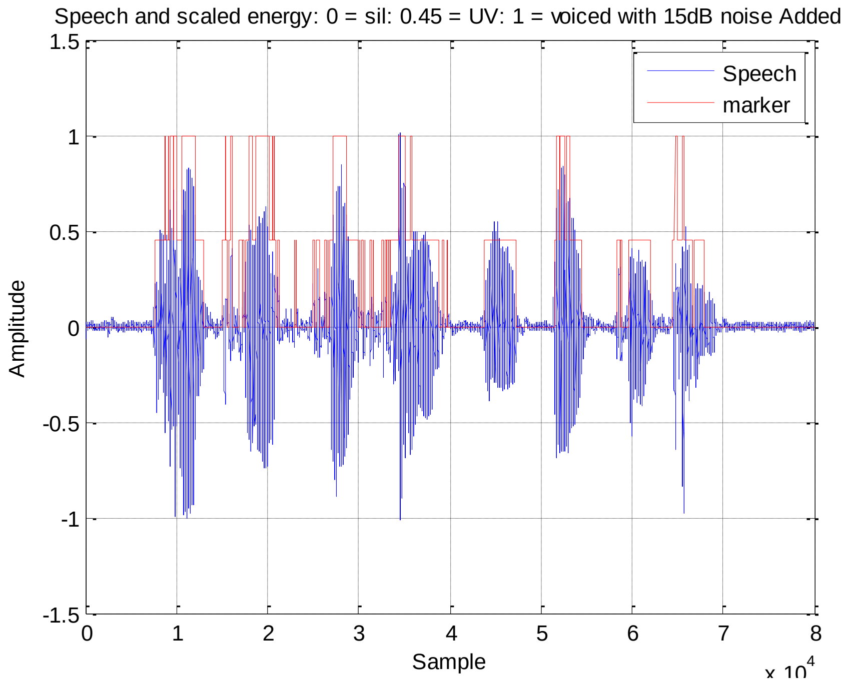

Figure 3: This is the graph of the original speech graph with 15 dB SNR added.

However, the higher the signal to noise ratio added to the original graph, the cleaner the speech will be which will make it for our group to detect unvoiced. When our group listens to the audio with noise added from MATLAB, we could easily hear the added white noise and we could also hear some parts of the speech that were considered voiced on the speech graph.

Use the .m files provided to obtain the spectrum of your speech in short segments.

Our group distinguished the spectrum of our speech by applying the *example_speechFFT2.m* file obtained from the course website to the noisy speech. This .m file is programmed to look at the Fast Fourier Transform (FFT) of voiced and unvoiced portions of noisy_speech.wav. In figure 4 we have the original eight seconds of speech recorded. Then the speech graph below is what happens after noise is added by the noisy speech file. By examining the two graphs we can see that the noise graph has increased.

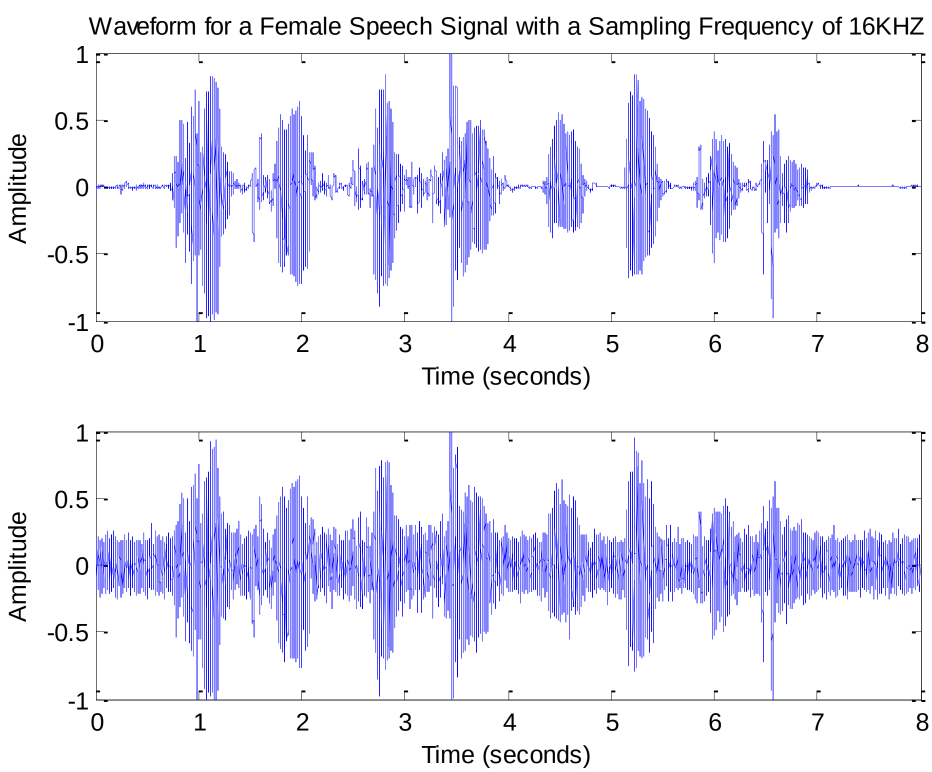

Figure 4 : Top graph: original graph with no noise added. Bottomgraph: original graph with noise added.

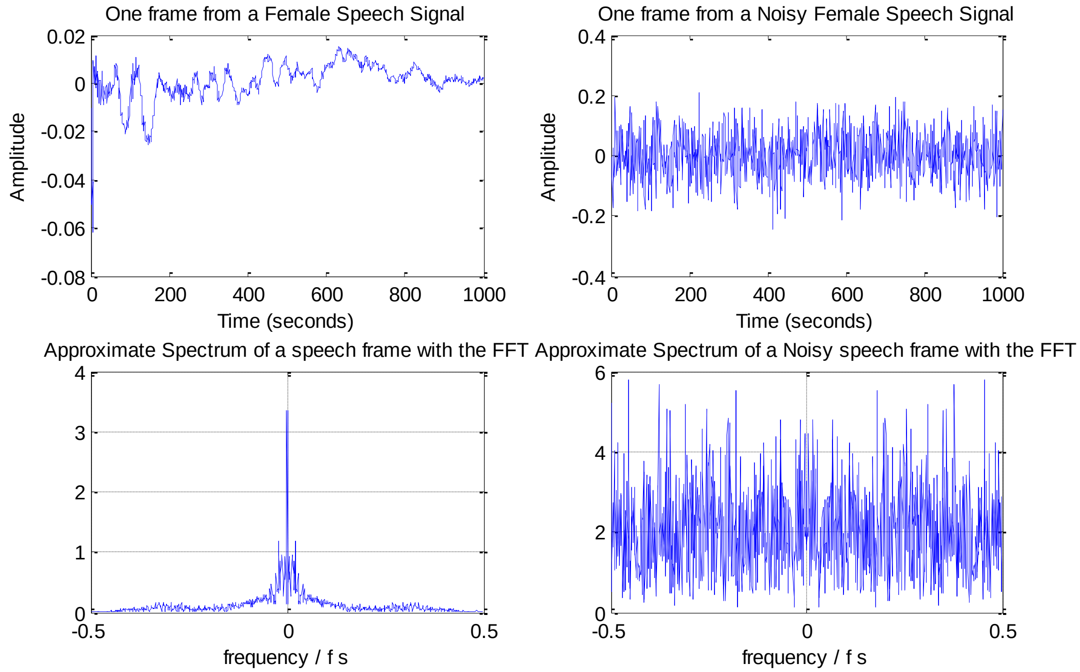

Figure 5: This illustration is broken into four graphs that consists both clean and noisy speech files

In Figure 5, we broke the graphs into smaller segments of the noisy graph, so we are able to see in detail what is changing as time goes by and see when it is voiced or unvoiced. We noticed that the graph is all noise and so we needed to keep moving to next segments to identify if the graph consists voiced and/or unvoiced.

Manually locate at least 5 unvoiced segments that were lost in the noisy signal.

We used the *example_speechFFT2.m* file to locate the 5 unvoiced segments in our *ssh2.wav* audio file. We were able to determine the voice and unvoiced in certain segments by setting conditions so that we could compare if a certain portion is voiced or not. Our conditions were that in order for a segment to be considered as voiced if the amplitude is higher than 25 dB or there is a high peak. If it is not voiced, then the other the choices could be noise or unvoiced. A segment could be considered as unvoiced because it is similar to spike as voiced but the amplitude will be below 25 dB. Noise is easily depicted by looking if the frequency's amplitude is less than or equal to 3.5. Below are five figures where unvoiced speech existed in our *ssh2.wav* file.

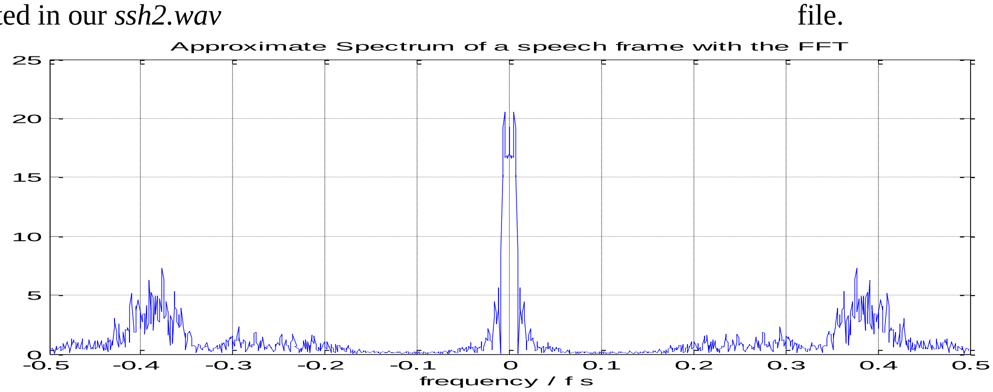

Figure 6: The illustration above is the graph of an approximate spectrum of a speech.

In Figure 6, the spectrum graph shows an unvoiced segment. Our group applied conditions we created based on the amplitudes we collected from the speech graph. Based on the spectrum frame, the highest amplitude is above 5 dB. Since the amplitude of Figure 6 is less than 25 dB as we indicated in our condition, this speech graph includes an unvoiced segment. The unvoiced segment occurs from approximately 0.35 fs to 0.4 fs. The channel noise can be easily depicted because it is an arbitrary signal which can be ignored in this graph.

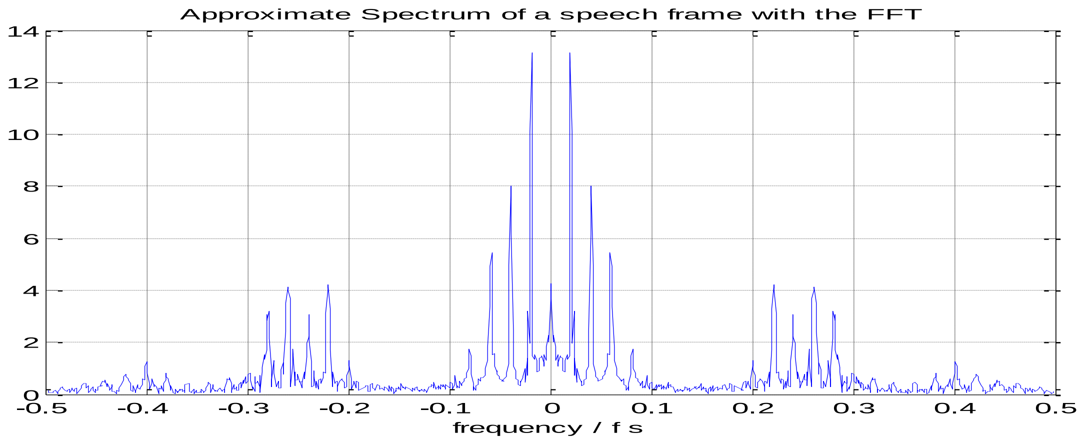

Figure 7: The speech graph above shows unvoiced segments that occurred during our speech.

Figure 7 is another example where an unvoiced segment occurred in our speech. The unvoiced segment occurred from approximately 0.2 fs to 0.3 fs. The amplitude of the signal from 0.2 fs to 0.3 fs is 4 dB. Therefore, our group concluded that the figure consists of unvoiced segment because the amplitude is less than 25 dB. We also ignored the highest peak in this graph because we knew that it was channel noise which we could ignore.

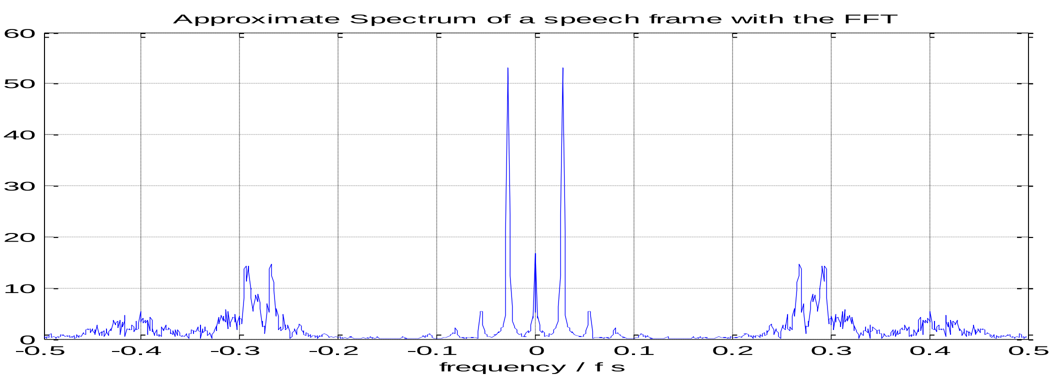

Figure 8: The highest amplitude of the signal (ignoring the channel noise) is approximately 15 dB. Therefore, the signal is an unvoiced segment.

Figure 8 displays an unvoiced segment based on our speech. The unvoiced segment occurred from approximately 0.25 fs to 0.33 fs. We determined that the unvoiced segment occurred in this region because during this segment, a high amplitude of about 12 dB occurred whereas the other frequencies are very low. Therefore this figure is another example of speech segment that includes an unvoiced segment.

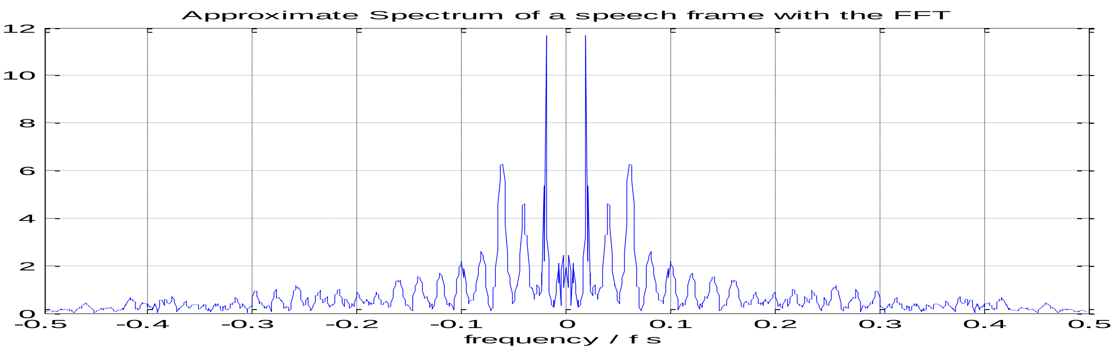

Figure 9: We could easily depict that this graph consists of unvoiced segment because of its highest amplitude (ignoring the channel noise) which is 6 dB.

In Figure 9, an unvoiced segment appears in the speech graph. At first, it was difficult for our group to conclude that this speech graph includes an unvoiced segment because we thought the signals were just random amplitudes therefore we thought that this graph only had noise. We observed the graph for a second time and we noticed that the amplitudes of the frequencies were relatively low. But the highest peak in this graph is 6 dB. Since 6dB is less than 25 dB but greater than 3.5 dB, thus we conclude that an unvoiced segment exist in this speech graph between 0.4 fs to 0.1 fs.

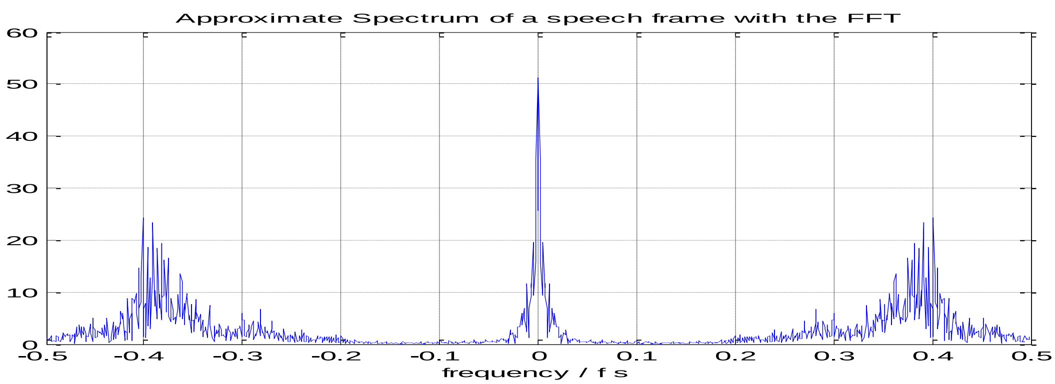

Figure 10: This is an example of a cleaner speech graph that includes an unvoiced signal.

In Figure 10, the unvoiced signal occurred from approximately 0.3 fs to 0.4 fs. The highest amplitude is 25 dB. Since 25 dB is greater than 3.5 dB, this graph is not a noise signal. We consider this graph to have an unvoiced signal because there is a large range of frequencies with high amplitudes whereas the other signals have lower amplitudes.

Write a MATLAB script that will automatically label voiced segments on the spectrum.

After viewing the spectrum in problem 6, we determined the voiced segments for each frame by writing a few lines of code that will detect whether the frame we are looking at is voiced, unvoiced, or silenced. We started off by viewing the script file containing the code for the spectrum and identified a vector with 1024 points that holds the data for the frequency of the signal. We then decided to search the data for the maximum amplitude and check to see if what was above a specific value. By doing this we would be able to determine whether or not the signal is voiced, unvoiced, or silenced. In question 6 we had to identify 5 unvoiced segments, during that process we came to determine that if the maximum amplitude in any given frame is below 4 Hz, we considered the whole frame to be silenced and set a marker to zero. If the max amplitude is between 4 and 25 the speech during that frame is unvoiced and set a marker to 0.5. If the amplitude is greater than 25 we view the speech in that frame as voiced and set a marker to 1.

Using these facts, we wrote a bunch of conditional statements that checked the amplitudes and set the markers. At the end of the existing function, we plotted the noisy speech vector as well as the marker on the same graph. When the program is run, the last graph produced is a plot of the speech with noise in addition to a marker. The figure 11, below, shows the results for our noisy speech applied

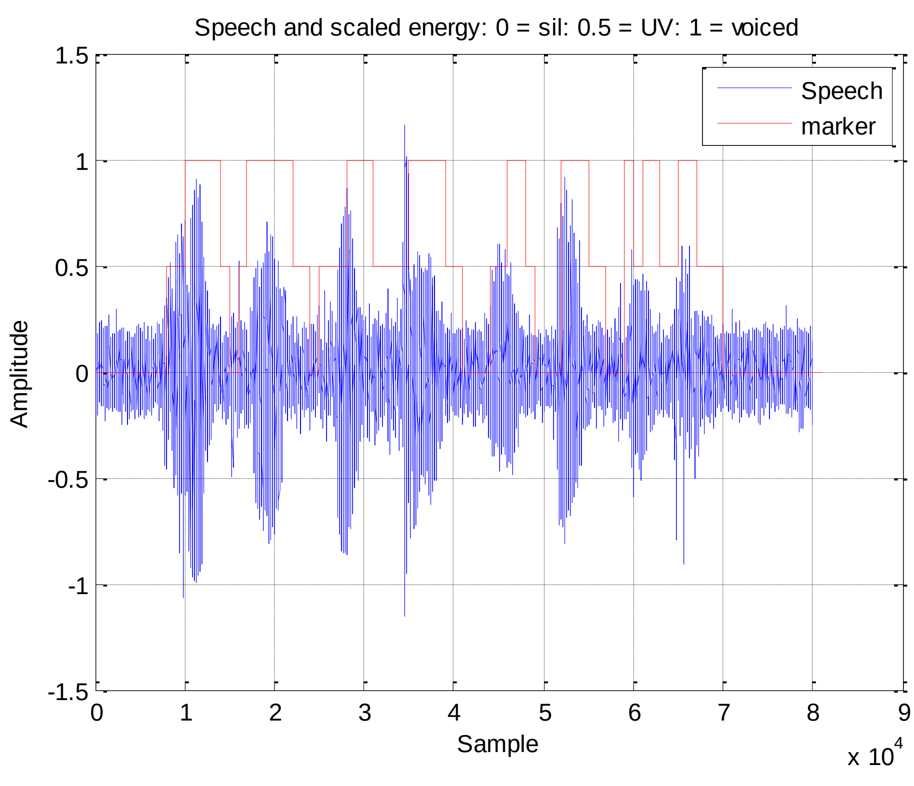

Figure 111: Plot of noisy speech (5dB added) with a marker

Figure 11 is a graph of our noisy speech with marker labeling the voiced, unvoiced, and silenced segments. In the graph, we noticed that the marker does a much better job labeling these different segments when compared to the procedure that uses the energy in the signal. We would have to say that setting the marker based on the amplitude gives a much better estimation of the different types of segments in speech. The reason is that we break the speech into different sounding segments is to look into much more detail at each specific frequency produced and analyze 512 frequencies in one frame. This method gives a much better look into how to label the segments because it looks at the individual frequencies within a frame instead of calculating the energy within a given frame. The energy method will yield more errors because it takes averages of the energies within a single frame.

10-point Moving Average Filter

After doing some extensive research, we found many ways of implementing a 10-pt moving average filter. I chose the simplest method I could find out. This method requires us to read in the noisy speech from a .wav file, create a 1X10 matrix with the number 1/10 repeated ten times, and thirdly store the number 1 into a variable. We then called in the built MATLAB function 'filter'. Passed in all three variables and stored the result in a variable named *y*. The variable y will contain the filtered speech. Plots of the noisy speech as well as the filtered speech are both shown in the figures 12 and 13 below. We found the filter information from the Matlab http://www.mathworks.com/help/matlab/data_analysis/filtering-data.htmlwebsite.

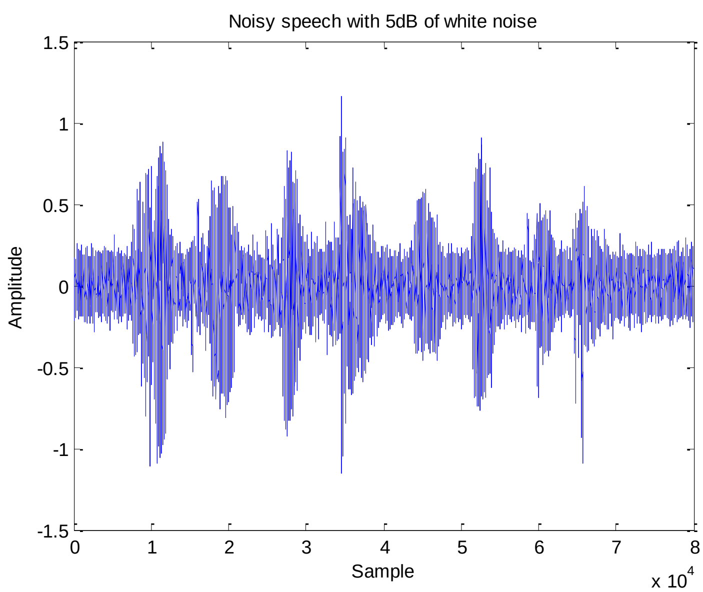

Figure 12: Plot of Noisy speech with 5dB of white noise

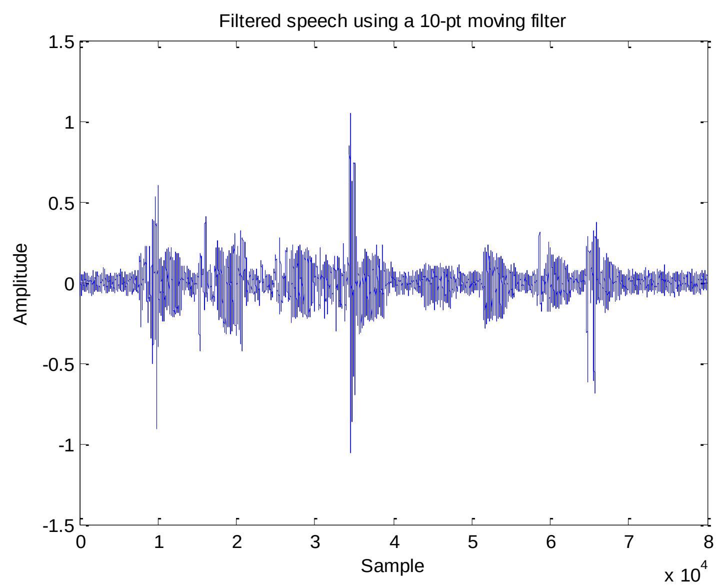

Figure 12: Plot of the filtered speech using a 10-pt moving average

In the figures above, the noisy speech plotted by the user has been filtered using a 10-pt moving average filter. The changes I noticed by observing the graphs are that the noisy speech has been filtered in approximately by a factor of 10. The effect on the audio are that the speech still sounds like it has noise in it, but that noise is at a much further reduced volume. Not only was some of the noise removed, but some of the speech was also removed. The audio effect on the speech is similar to the effect on the noise where that the speech sounds significantly quieter than the noisy speech. The moving average filter has the effect of lowering the total volume of the noisy speech.

Repeat Problem 3 while using Moving Average Filter

After generating the filtered speech, I followed the same procedures in problem three but this time for the filtered speech. I started by separating the speech into frames, initializing them, and then using afor loop to check the threshold and markers to each frame. Once I completed these steps, I graphed the filtered speech with the markers. The figure14, below, shows the results.

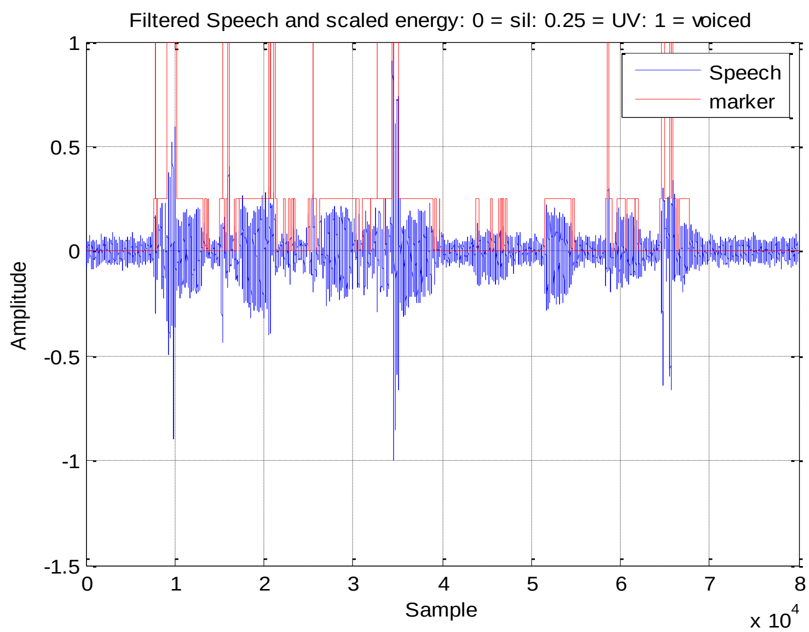

Figure 14: Plot of filtered speech. Markers are set from the range of 0.04 to 0 4

In the figure 14, above, a graph of filtered speech is shown with markers that where set using energy thresholds for each frame. Based on the graph we can tell that by filtering the speech, we lose part of the signal which causes the volume and the frequency to be reduced for the recorded speech. With some of the speech lost, it makes it much harder to determine the unvoiced, voiced, and silenced portions within a single frame. The energy technique does not provide the most accurate method for determining whether the frame is voiced, unvoiced, or silenced.

Conclusion

In conclusion, we would have to say that the third Assignment was very successful as we were able to solve all the problems given as well as learn new capabilities of MATLAB. All the results are acceptable because they were consistent throughout the lab and met our expectations. Most of the graphs were very accurate in the information they revealed about the speech. However, the graphs for the energy and amplitudes were not as accurate. There are times were we expected voiced speech but the marker was set at unvoiced. The problems included in the assignment sufficiently achieved its goal of exposing the student to MATLAB's ability to analyze and record speech. Overall, the assignment was a complete success.

Discussion

During the course of the lab, we did not have any trouble but stumbled on questions 6 and 7. Problem 6 posed some challenges because we did not know how to analyze the spectrum produced by the Fourier transform. After some careful observations and interpretation of the spectrum, we came to find that our speech had some possible channel noise added to the recording. This additional channel noise is a possible for the errors seen in the speech with markers. The channel noise made it harder to determine whether a particular portion of the speech was unvoiced, voiced, or silence. We quickly noticed that the channel noise added additional peaks in the spectrum close to the midpoint of the data in that region. We decided to ignore any signals that were located around halfway in the input signal of each frame. Once we eliminated the channel noise, we were better able to guess whether the speech in that particular frame was voiced, unvoiced, or silenced.

In problem 7, as we attempted to write some MATLAB script to determine whether speech is voiced, unvoiced, or silence we struggled to figure out what data to analyze. After figuring out that we need to observe the amplitudes of the frequencies, we came up with a range that would determine whether the speech in a specific frame is voiced, unvoiced, and speech. However, we remembered the corrections we made in problem 6 and applied those same corrections to problem 7 in order to keep the consistency throughout the lab. At this point, we were able to come up with a graph that gave a better approximation of which segments in the speech were voiced, unvoiced, or silence when compared to the graph given to us earlier using the energy in the signal.

While doing the lab, we learned a couple of new commands as well as capabilities of MATLAB that we did not know before. We learned how to perform a Fourier Transform on speech. This feature would be very effective when processing and analyzing speech. We also learned that MATLAB could be used to record and playback speech as well as plot the amplitude of the speech signal.

We did not notice many inconsistencies in this assignment except for one in which MATLAB gave a warning that some of the data was cutoff when creating the .wav file for the noisy speech and filtered speech. Although, MATLAB warned that some data was lost during the writing process when played back there was not a noticeable loss in data. The difficulty of the assignment is ok and it exposes an individual to different abilities of the MATLAB software.

APPENDIX

1. To record and save speech file to MATLAB

```
fs = 10000;
speech = wavrecord(80000,fs);
wavwrite(speech, fs, ‘ssh2.wav’);
```

2. To display clean and added noise speech graph. To use the Add Noise program we applied the program provided on the course website.

```
%TITLE: V_UV_detect.m
%this script detects voice speech in an utterance
%need a wave file called shout in current dir.
clear all; close all; clc;
[sp, fs] = wavread('ssh2'); %reqad the speech file
sp = sp(1:80000);
sp = sp/max(sp); % normalizing the speech the speech signal
sp = AddNoise(sp, 15);

%Segment speech into frames
fr_sz = floor(10/1000*fs); %frame size of 10ms
len = length(sp); %length of the speech
n_fr  = floor(len/fr_sz); %number of frames

beg = 1; enn = fr_sz; %initializing frames

% sil_thresh = .0005;
% uv_thresh = .005;
sil_thresh = .0045;
uv_thresh = .045;

for i = 1:n_fr,
spf = sp(beg:enn);
en = sum(spf.^2)/fr_sz;
 if en <= sil_thresh,
mrkr(beg:enn) = 0;%marker for silence
elseif en <=uv_thresh,
        mrkr(beg:enn) = 0.45; %marker for unvoiced
 else
        mrkr(beg:enn) = 1; %marker for voiced
end

beg = enn + 1;
enn = enn + fr_sz;
```

```
end

figure;
 plot(sp); hold on; plot(mrkr, 'r');
title('Speech and scaled energy: 0 = sil: 0.45 = UV: 1 = voiced with NO Noise Added');
xlabel('Sample'); ylabel('Amplitude');
legend('Speech', 'marker');
grid;

sil_ind = find(mrkr == 0);
uv_ind = find(mrkr == 0.45);
v_ind =find(mrkr == 1);
pause
soundsc(sp(v_ind), fs);

%TITLE: AddNoise.m
function nsp = AddNoise(sp, snr);
%AddNoise.m created by Dr. Uche Abanulo; %Spring 2009
%inputs:
%sp = Speech
%SNR = signal to noise ratio in decibels (dB)%
%output:
%nsp = noisy speech with a signal to noise ratio of SNR dB
% Generate Additive White Guassian Noise (AWGN) - zero mean and unity variance)
len = length(sp);
ns = randn(1, len); % generating the noise signal
ns = (ns - mean(ns))/std(ns);%standardizing the signal Scale noise
sc_ns = (ns*max(sp))./(max(ns)*(10^(snr/10)));
%Generate Noisy speech
nsp = sp + sc_ns';
```

3. To obtain the spectrum of our noisy speech graph

```
%This program looks at the fft of voiced and unvoiced portions of speech
%Reseting the Program
clear all;
close all;
clc;
%Reading the file
[speech, fs] = wavread('noisy_speech.wav');
speech1 = AddNoise(speech, 5);
%Converting the x axis from sample numbers to time
x = 1:length(speech);
t = x./fs;
```

```
%Plotting the file
figure;
subplot(2,1,1);
plot(t, speech);
xlabel('Time (seconds)');
ylabel('Amplitude');
title('Waveform for a Female Speech Signal with a Sampling Frequency of 16KHZ');
axis([0 max(t) -1 1]);
subplot(2,1,2);
plot(t, speech1);
xlabel('Time (seconds)');
ylabel('Amplitude');
axis([0 max(t) -1 1]);
%Displaying voiced and unvoiced speech segments
%Detecting voiced and unvoiced segments based on energy
%Computing the frame-by-frame energy of the signal
f_size = 1000;
len = length(speech);
num_F = floor(len/(f_size));
beg =1;
en = f_size;
for i = 1:num_F
        speech_frame = speech(beg:en);
        speech_frame1 = speech1(beg:en);
        theta = sum(abs(speech_frame))/length(speech_frame);
        energ(beg:en) = theta;
        beg = beg + f_size;
        en = en + f_size;
        figure;
        subplot(2,2,1);
plot(speech_frame);
xlabel('Time (milliseconds)');
ylabel('Amplitude');
title('One frame from a Female Speech Signal');
subplot(2,2,2);
plot(speech_frame1);
xlabel('Time (milliseconds)');
ylabel('Amplitude');
title('One frame from a Noisy Female Speech Signal');
subplot(2,2,3);
N = 1028;
X = abs(fft(speech_frame,N));
X = fftshift(X);
F = [-N/2:N/2-1]/N;
plot(F,X),
xlabel('frequency / f s')
```

```
grid;
title('Approximate Spectrum of a speech frame with the FFT');
subplot(2,2,4);
N = 1028;
X = abs(fft(speech_frame1,N));
X = fftshift(X);
F = [-N/2:N/2-1]/N;
plot(F,X),
xlabel('frequency / f s')
grid;
title('Approximate Spectrum of a Noisy speech frame with the FFT');
pause
end
```

4. To automatically label voiced segments based on the spectrum.

```
 %This program looks at the fft of voiced and unvoiced portions of speech
%Reseting the Program
clear all;
close all;
clc;

%Reading the file
[speech, fs] = wavread('ssh2.wav');
speech1 = AddNoise(speech, 5);
%Converting the x axis from sample numbers to time
x = 1:length(speech);
t = x./fs;

%****************************************************
%Plotting the file
%****************************************************
figure;
subplot(2,1,1);
plot(t, speech);
xlabel('Time (seconds)');
ylabel('Amplitude');
title('Waveform for a Male Speech Signal with a Sampling Frequenvy of 16KHZ');
axis([0 max(t) -1 1]);
subplot(2,1,2);
plot(t, speech1);
xlabel('Time (seconds)');
ylabel('Amplitude');
axis([0 max(t) -1 1]);
%****************************************************
```

```
%*****************************************************
%Displaying voiced and unvoiced speech segments
%*****************************************************
%Detecting voiced and unvoiced segments based on energy

%Computing the frame-by-frame energy of the signal
f_size = 1000;
len = length(speech);
num_F = floor(len/(f_size));
beg =1;
en = f_size;
for i = 1:num_F
        speech_frame = speech(beg:en);
        speech_frame1 = speech1(beg:en);
        theta = sum(abs(speech_frame))/length(speech_frame);
        energ(beg:en) = theta;
        beg = beg + f_size;
        en = en + f_size;
        figure;
subplot(2,2,1);
plot(speech_frame);
xlabel('Time (seconds)');
ylabel('Amplitude');
title('One frame from a Male Speech Signal');
subplot(2,2,2);
plot(speech_frame1);
xlabel('Time (seconds)');
ylabel('Amplitude');
title('One frame from a Noisy Male Speech Signal');
subplot(2,2,3);
N = 1028;
Xa = abs(fft(speech_frame,N));
Xa = fftshift(Xa);
F = [-N/2:N/2-1]/N;
sigpart(1:499)=Xa(530:length(Xa));
%Segment speech into frames
if max(sigpart)>25,
mrkr(beg:en) = 1; %marker for voiced
elseif max(sigpart)>4 && max(sigpart)<25,
   mrkr(beg:en) = 0.5; %marker for silence
else
mrkr(beg:en) = 0; %marker for unvoiced
end
plot(F,Xa);
xlabel('frequency / f s')
grid;
```

```
title('Approximate Spectrum of a speech frame with the FFT');
subplot(2,2,4);
N = 1028;
X = abs(fft(speech_frame1,N));
X = fftshift(X);
F = [-N/2:N/2-1]/N;
plot(F,X),
xlabel('frequency / f s')
grid;
title('Approximate Spectrum of a Noisy speech frame with the FFT');
end
figure;
plot(speech1); hold on; plot(mrkr, 'r');
title('Speech and scaled energy: 0 = sil: 0.5 = UV: 1 = voiced');
xlabel('Sample'); ylabel('Amplitude');
legend('Speech', 'marker');
grid;

sil_ind = find(mrkr == 0);
uv_ind = find(mrkr == 0.3);
v_ind =find(mrkr == 1);
pause
soundsc(speech(v_ind), fs);
```

5. To apply the Moving Average Filter

```
function Out = speechfilter(sp)
%speechfilter.m created by Ozino, Abby, and Brian; %Fall 2012
%inputs:
%sp = Speech
%output:
%Out = Filtered speech using a 10-pt moving average filter
 a=1;b=[1/10 1/10 1/10 1/10 1/10 1/10 1/10 1/10 1/10 1/10];
x=sp;
Out=filter(b,a,x);
close all; plot(Out);  title ('Filtered speech using a 10-pt moving filter');
xlabel('Sample'); ylabel('Amplitude');figure; plot(sp); title ('Noisy speech with 5dB of
white noise');
xlabel('Sample'); ylabel('Amplitude');
wavwrite(Out,10000,'filtered_speech');
```

6. Applying Moving Average Filter for Problem 3

```
%this script detects voice speech in an utterance
%need a wave file called shout in current dir.
clear all; close all; clc;
[sp, fs] = wavread('filtered_speech'); %read the speech file
sp = sp(1:80000);
```

```
sp = sp/max(sp); % normalizing the speech the speech signal
sp = AddNoise(sp, 5);

%Segment speech into frames
fr_sz = floor(10/1000*fs); %frame size of 10ms
len = length(sp); %length of the speech
n_fr  = floor(len/fr_sz); %number of frames

beg = 1; enn = fr_sz; %initializing frames

% sil_thresh = .0005;
% uv_thresh = .005;
sil_thresh = .005;
uv_thresh = .05;

for i = 1:n_fr,
 spf = sp(beg:enn);
en = sum(spf.^2)/fr_sz;
if en <= sil_thresh,
mrkr(beg:enn) = 0;%marker for silence
elseif en <=uv_thresh,
mrkr(beg:enn) = 0.5; %marker for unvoiced
else
mrkr(beg:enn) = 1; %marker for voiced
end

 beg = enn + 1;
 enn = enn + fr_sz;

end

figure;
plot(sp); hold on; plot(mrkr, 'r');
title('Filtered Speech and scaled energy: 0 = sil: 0.5 = UV: 1 = voiced');
xlabel('Sample'); ylabel('Amplitude');
legend('Speech', 'marker');
grid;

sil_ind = find(mrkr == 0);
uv_ind = find(mrkr == 0.3);
v_ind =find(mrkr == 1);
pause
soundsc(sp(v_ind), fs);
```