

EE 8105

DIGITAL SIGNAL PROCESSING

Fall 2016

**PROJECT**

Zubier Abdullah

Student ID - 500419866



**Background**

Speech recognition can be classified into one of two types – **speaker** **recognition**, where the identity of the speaker is verified by comparison of the input speech signal and previously recorded vocal signals from the speaker and **word recognition**, where the system is capable of differentiating individual words spoken. Speech recognition has seen significant improvements in the preceding decades and has been a very fruitful line of research amongst scientists for many years. A successful and robust speech recognition system will allow users greater flexibility in interacting with their devices and recent advances have shown that it allows users to more easily and accessibly use their devices. Speech recognition systems make many tasks much more inexpensive and have a wide variety of practical applications. A simple example of this would be speech recognition systems in place for customer call centers – software will be able to help customers on the phone have their queries satisfied and for more complicated demands, the system can then hand the customer off to human operators.

For many people, especially those who are not used to using technology, using a computer or a smartphone can be a difficult and complicated matter. Having a robust speech recognition system will make it easier for these users to use their computers and devices. For this to be carried out, though, the systems have to be robust enough to be able to tolerate noise.

**Objective**

The objectives of this project are two fold

* Develop a speech recognition in MATAB capable of distinguishing words from different speakers and the same speaker
* Evaluate the noise performance of the system with additive white Gaussian noise being added to the input signal with SNR 50 dB, 10 dB, 1dB, 0.1 dB .

**Methodology**

For the development of this project we will use DSP techniques used in class. The principal knowledge that we require to evaluate this project is that of the DFT, FFT, DC level, sampling theory and most importantly, the cross correlation algorithm and the spectrogram function.

Firstly, we talk about the removal of the dc signal component. This is done from the audio signals because the dc component of the signal is not useful for our purposes during analysis and will interfere with the analysis if the frequency components of the signal are located near the dc component. Thus we remove this from the signal during processing by removing the mean of the signal.

The recordings which are used as samples are all taken from the internet. To ensure that the voices used are as identical as possible, I have taken vocal samples from text to speech software available online. This way, the system can evaluate the words based solely on the differences in sound and not on vocal register. Another thing we must mention is that if the sound samples we have used are recorded in stereo format (the left and the right speakers playing different parts of the song simultaneously), we must convert them to mono for easier calculations in Matlab. This can be done simply through the Matlab program.

For any sort of digital signal processing, the Discrete Fourier Transform is an invaluable tool. The DFT is defined as follows

As we know from our studies, the FFT algorithm is a much more computationally efficient version of the DFT algorithm and it is that what we will use for the purposes of this project. The DFT and the FFT both perform the same function, that is, they convert the time domain representation of the signal into the frequency domain where we can learn much about it.

After doing the DFT/FFT conversions, the signals will be converted from the time domain to the frequency domain. The frequency spectrum is the summation of all of the frequency components within the signal. When we look at the frequency spectrum of different words, we can see that the words are composed of a number of different frequencies, each having separate values for the maximum and minimum of the frequency spectrum When comparing the difference between the two signals, then it is difficult and inadvisable to compare two spectrums with different measurement standards. For a proper comparison, the units must be same for both. To alleviate this problem, we normalize the spectrums to ensure that the measurement standard is the same. Doing this will allow us to calculate the error effectively.

We use the following equation for the linear normalization

After normalization, the spectrums are in the range [0,1]. This only changes the value of the spectrum but does not change the shape of the spectrum in anyway.

After the spectrums have been normalized, we can then proceed to the cross correlation algorithm. As we know, the cross correlation is a measure of how closely two signals are correlated to each other. It is also useful to estimate the shift parameter.

The definition of the cross correlation is

From the equation we can observe that the cross correlation formula is very similar to the convolution sum. We keep one signal fixed and shift the other signal and then multiply the two signals together. Finally, we take a summation of it all.

What we should know is that for two signals which have no time shift, their cross correlation should be maximum. The other conclusion that we should remember is that the position difference between the maximum value position and the middle point position of the cross correlation is the length of the time shift for the two sources.

For the purposes of the development of our algorithm, we will heavily rely on the symmetry of the cross correlation plots of the signals. If two spectrums are identical, then the cross correlation should be identical and also should be symmetric. The system will then compare the level of symmetry for the cross correlation and from that, the system will be able to identify which of the words have more similar spectrum. In essence, it will tell us which two sounds are more likely the same words.

To understand this, we look at the cross correlations of two words with a reference word Hello. The first word is Hello and the second word is Girl.



Figure 1: Cross correlation of ‘hello’ with itself and ‘hello’ with ‘girl’

We can see that there is a much greater level of symmetry in the figure on the left than with the second figure. This is because the cross correlation of two words which are similar will yield a more symmetrical plot than the cross correlation of two words which are not similar.

We can observe this much more clearly when we look at the cross correlations of the words House and Hello.



Figure 2: Cross correlation of ‘hello’ with itself and ‘hello’ with ‘house’

We can observe for the cross-correlation curves that they are symmetric about the maximum point, which lies in the middle. For the speech recognition program, we need to find the location of this maximum value for comparison. To establish whether the cross-correlation is symmetrical or not, we calculate the difference between the values to the right of the maximum and from the left of the maximum. We find the absolute value of this difference and find the mean square error of this value.

The mean squared error is calculated by the formula

By calculating the mean squared error, we can then numerically quantify which of the signals are more symmetrical. This can be deduced because the more symmetrical signal will have less error.

In this project, we will be relying on the Spectrogram function. This is a time frequency plotting which contains the power density distribution. From Matlab, we can easily get this by defining the parameters – sampling frequency, the length of the STFT and the length and type of window. For the purposes of this project, we will be using the Hanning window.



Figure 3: Spectrogram of ‘Hello’

This is the spectrogram of the word ‘Hello’ which we are using as a reference word. We can see that much of the power is concentrated in the lower frequencies.

Using the Hanning window, we will be breaking up the input sequences into a series of smaller parts, on which we will perform the Fast Fourier Transform on. This is advantageous to us because using a window function and ‘moving it’ along the length of the input signal will prevent spectrum leakage and create a smoother spectrum. To ensure greater reliability in our spectrum, we will make sure there is overlap in the window function.

Thus the entire process of the cross correlation speech recognition algorithm can be summarized below

* Take set of input reference words (For our project, we have used 5 different words).
* If the input reference word is recorded in stereo, then convert it into mono for easier calculations.
* Choose a test word. All of the other words will be compared with this test word.
* Remove the dc component from all of the audio signals
* Compute the spectrogram for each of the words, including reference and test words.
* Transpose the absolute values of the spectrograms and then sum them together. Use the linear normalization formula to normalize each of these values.
* Compute the cross-correlation using the xcorr function in Matlab for all reference words and the test word.
* Calculate the frequency shifts of the cross-correlations
* Calculate the level of symmetry by calculating the mean squared error between the left and right sides of the symmetrical cross-correlations.
* By finding the lowest value of the error, the system can identify which of the reference words is the best match for the test word.

Finally the experiment will be repeated for differing words set as the test word and for different levels of noise added to the system. The experiment will be repeated for three noise levels in total. No noise added, 1 dB noise added and 2 dB noise added.

Signal to noise ratio (SNR) is a measure used in science and engineering to denote the level of desired signal to background noise. It is defined as the ratio of signal power to noise power, with a ratio less than 1 meaning that the power of the noise is higher than the power of the signal and vice versa. It is often expressed in dB.

Additive White Gaussian Noise is a basic noise model used in information theory. It is used to mimic the effect of many natural processes. AWGN is called so because it is added to any noise already present in the system, because it has a uniform power throughout the frequency band and because it has a Gaussian Distribution in the time domain.

We will observe the spectrum of the AWGN and how it changes the spectrum of the input word in the following diagram.



Figure 4: Effect of Various SNR added to the input word

Here we can see how the differing SNRs affect the input word. What must be remembered is that the lower the SNR, the greater the amount of noise added.

Let us look at the noise profile of only the AWGN.

We can see that the noise distribution is uniformly added.

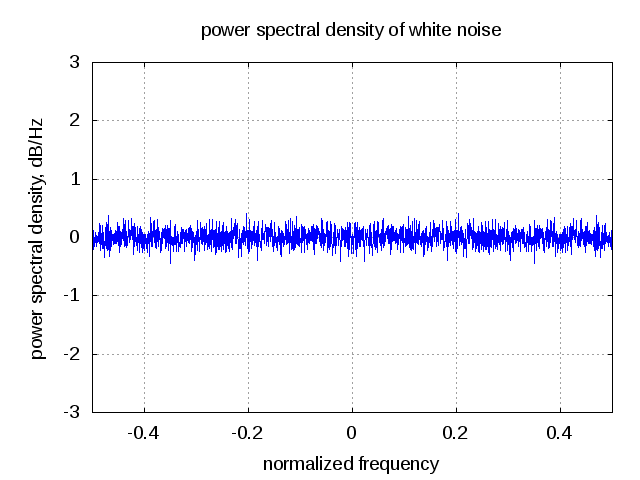


Figure 5: Power Spectral density of white noise (AWGN)

In the following figure we can see the spectra of the reference words as well as the test word. In this instance, the test word was chosen as the first reference word (Hello).

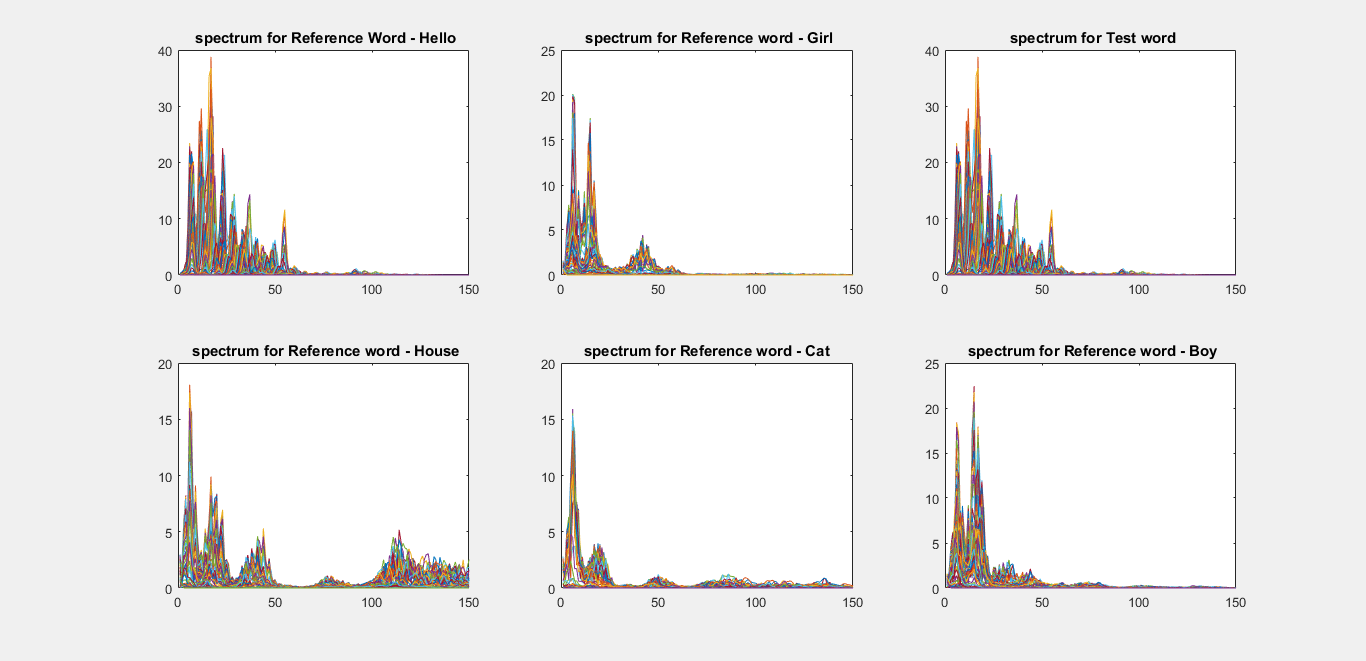


Figure 6: Spectrogram output of the various reference and input words

The units for the x and y axis are taken as number of samples and magnitude respectively. From this figure, we can see there is quite a bit of differences between the spectra of the different words. We can also see that the test word and the first reference word have identical spectra, as it should be.



Figure 7: Normalized spectra for reference and input words

This above is the normalized spectra for each of the reference words. We are plotting this is because it is much more legible than the previous figure. Here we have performed a number of transpose and sum operations to reduce our original double matrix into a single row or column matrix. This allows us to plot and do calculations more easily.

The following figures show the cross correlations of the test word with the various reference words. Running the program we obtain the following outputs.



Figure 8: Cross correlation of Test word(Hello) and Reference words



Figure 9: Cross correlation of Test word(Hello) and Reference words

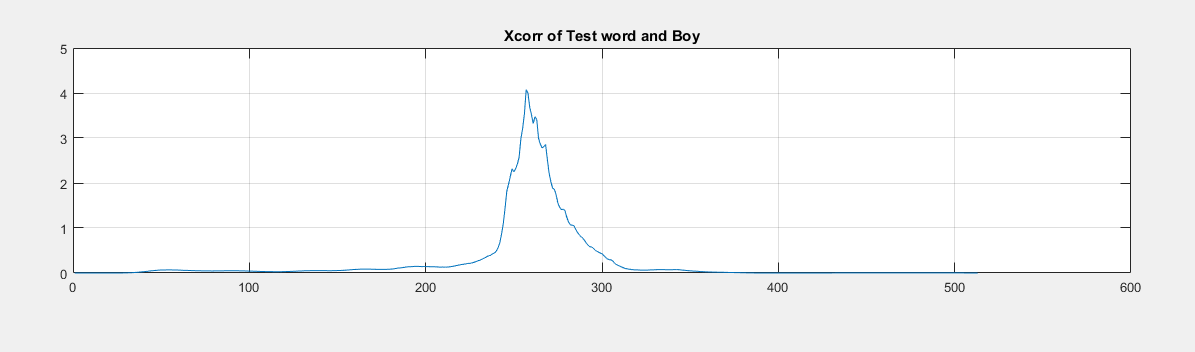


Figure 10: Cross correlation of Test word(Hello) and Reference words

Looking at these figures, we can see that the cross correlation of two similar words will yield a cross correlation spectrum similar to the test word and hello. We can see a high degree of symmetry in that plot.

With this, we can compute the mean squared error. The system will identify the closest matching word by finding the lowest mean squared error of the cross correlation.

We will also repeat the experiment by adding varying levels of noise to the input signal.

**Results**

The following table will be our results table.

|  |  |  |  |
| --- | --- | --- | --- |
| Test word | Same speaker success | Different speaker success | SNR |
| Hello | Yes | No | No noise added |
| House | No | No | No noise added |
| Boy | Yes | Yes | No noise added |
| Girl | Yes | No | No noise added |
| Cat | Yes | Yes | No noise added |
| Hello | Yes | No | 50 dB |
| House | No | No | 50 dB |
| Boy | Yes | Yes | 50 dB |
| Girl | Yes | No | 50 dB |
| Cat | Yes | Yes | 50 dB |
| Hello | No | No | 10 dB |
| House | Yes | No | 10dB |
| Boy | No | No | 10 dB |
| Girl | No | No | 10 dB |
| Cat | No | No | 10 dB |
| Hello | No | No | 1 dB |
| House | Yes | Yes | 1 dB |
| Boy | No | No | 1 dB |
| Girl | No | No | 1 dB |
| Cat | No | No | 1 dB |
| Hello | No | No | 0.1 dB |
| House | Yes | Yes | 0.1 dB |
| Boy | No | No | 0.1 dB |
| Girl | No | No | 0.1 dB |
| Cat | No | No | 0.1 dB |

What must first be clarified is that the SNR column represents the SNR of the input signal (Test word) against the amount of noise power. Thus the ratio of the input power to the noise power is 50dB or 316.22. 12/50

The following is the real estimate of the power difference at various SNRs.

|  |  |
| --- | --- |
| Power in dB | Real power ratio |
| 50 dB | 316.22 |
| 10 dB | 3.16 |
| 1 dB | 1.120 |
| 0.1 | 1.0166 |

**Discussions**

From our results we can see that the system is not very robust and noise tolerant. This is greatly limits the functionality of the system as we have mentioned that for good performance the system must be able to tolerate noise and be able to identify the spoken words accurately in less than ideal situations.

Another observation that we can make from the results is that the system has difficulty differentiating more complicated words such as house from simpler words. Even in situations with no noise or high SNR, the system misidentifies the word house. It however, correctly identifies the word ‘house’ at lower SNRs, although this is because for higher levels of noise, the system finds that the lowest value of error is from the word house, regardless of the word being used as the input.

Taking this into account, the system matches our expected outcomes to a certain extent. We expected the system to be accurate 100% of the time for no noise added but from the performed experiments, we can see that the system correctly identifies the words 80% of the time if the input and reference word are the same but only 40% of the time if the speakers are different from no noise.

We can also see that the system is extremely susceptible to noise. This was expected but not to such a high regard. We can see that the system misidentifies the reference word almost constantly as soon as the SNR dips below 50 dB.

Thus the system and the algorithm based on the cross correlation of the input signal and the reference words is extremely noise intolerant and not robust. However, it does work for situations with no noise.

One of the limitations of the algorithm is that, if input word is not any of the reference words, it will misattribute it as one of the reference words.

Thus compared to our expected outcomes, the system is able to identify the input word from the reference word only some of the time. It is most accurate when the speaker is the same and accuracy decreases if the speaker changes and if noise is added. The system is extremely susceptible to noise.

**Conclusion**

We have successfully designed a speech recognition system with the use of an algorithm based on cross-correlation. However, as the results indicate, the system is extremely susceptible to noise and to misidentification if the reference word and the input word is not spoken by the same speaker. Thus the system is not robust enough to be used for any commercial purposes. However, it is a good way to begin understanding how speech recognition systems work.

In practice, developed speech recognition systems are much more complicated and rely heavily on statistical analysis. The most dominant models used are the Hidden Markov Model and the use of neural networks.

**References**

1. D.Ning, *“Developing an Isolated Word Recognition System in MATLAB.”* MATLAB Digest. Retrieved from www.mathworks.com

2. A. Davletcharova, S. Sugathan, B. Abraham and A.P. James, *“Detection and Analysis of Emotion from Speech Signals”,* Procedia Computer Science (58)

3. Yang T. (2012), *“The Algorithms of Speech Recognition, Programming and Simulating in MATLAB”* (Bachelor’s Thesis), University of Gavle, Gavle, Sweden.

4. Adarsh K.P, A.R. Deepak, Diwakar R. and Karthik R. (2007), *“Implementation of a voice based Biometric System”*, Visveswaraya Technological University, Belagum, Bangalore, India

5. Sankar, K. (2012, March 25) Thermal Noise and AWGN, retrieved from <http://www.dsplog.com/2012/03/25/thermal-noise-awgn/>

**6.** Proakis, John G., and Dimitris G. Manolakis *Digital Signal Processing*. Upper Saddle River, NJ: Pearson Prentice Hall, 2007. Print.   
7. Vocal samples obtained from <http://www.text2speech.org/>

Appendix

Matlab code

%%

clc

clear all

close all

%the system will identify what word was said from a group of reference

%words

[Hello1,Fs1] = audioread('Hello\_4.wav');

[Girl1,Fs2] = audioread('Girl.wav') ;

[Test1,Fs3] = audioread('Cat2.wav');

[House1,Fs4] = audioread('House\_1.wav');

[Cat1,Fs5] = audioread('Cat.wav');

[Boy1,Fs6] = audioread('Boy.wav');

%%

% Test2 = Test1;

Test1 = awgn(Test1,1);

%This is where we add AWGN to the test word

% Test3 = awgn(Test2,10);

% Test4 = awgn(Test2,1);

% Test5 = awgn(Test2,0.1);

%since the Girl sound file is mono, we do not need to convert it.

% sound(Hello1)

% pause

% sound(Hello1,Fs1)

% pause

% sound(Girl1,Fs2)

%%

H1 = Hello1 - mean(Hello1);

G1 = Girl1 - mean(Girl1);

Ho1 = House1 - mean(House1);

T1 = Test1 - mean(Test1);

C1 = Cat1 - mean(Cat1);

B1 = Boy1 - mean(Boy1);

%%

% T1 = awgn(T1,0)

%sound(Hello1) %Removal of the dc component changes the quality of the voice sample

%%

s1 = spectrogram(H1,hanning(512),380);

s2 = spectrogram(G1,hanning(512),380);

s3 = spectrogram(T1,hanning(512),380); %THIS IS TEST WORD

s4 = spectrogram(Ho1,hanning(512),380);

s5 = spectrogram(Cat1,hanning(512),380);

s6 = spectrogram(B1,hanning(512),380);

ar1 = transpose(abs(s1));

ar2 = transpose(abs(s2));

ar3 = transpose(abs(s3));

ar4 = transpose(abs(s4));

ar5 = transpose(abs(s5));

ar6 = transpose(abs(s6));

a11 = sum(ar1);

a21 = sum(ar2);

a31 = sum(ar3);

a41 = sum(ar4);

a51 = sum(ar5);

a61 = sum(ar6);

a1\_norm = (a11-min(a11))/(max(a11)-min(a11));

a2\_norm = (a21-min(a21))/(max(a21)-min(a21));

a3\_norm = (a31-min(a31))/(max(a31)-min(a31));

a4\_norm = (a41-min(41))/(max(a41)-min(a41));

a5\_norm = (a51-min(a51))/(max(a51)-min(a51));

a6\_norm = (a61-min(a61))/(max(a61)-min(a61));

FA1 = transpose(a1\_norm); %Reference signal 1

FA2 = transpose(a2\_norm);%Reference signal 2

FA3 = transpose(a3\_norm); %Test signal

FA4 = transpose(a4\_norm);

FA5 = transpose(a5\_norm);

FA6 = transpose(a6\_norm);

%%

[x1,lag1] = xcorr(FA3,FA1); %This is for s1 and s3

[mx1,indice1] = max(x1);

Frequency\_shift1 = lag1(indice1);

[x2,lag2] = xcorr(FA3,FA2); %This is for s2 and s3

[mx2,indice2] = max(x2);

Frequency\_shift2 = lag2(indice2);

[x3,lag3] = xcorr(FA3,FA4); %This is for s3 and s4

[mx3,indice3] = max(x3);

Frequency\_shift3 = lag3(indice3);

[x4,lag4] = xcorr(FA3,FA5); %This is for s3 and s5

[mx4,indice4] = max(x4);

Frequency\_shift4 = lag4(indice4);

[x5,lag5] = xcorr(FA3,FA6); %This is for s5 and s3

[mx5,indice5] = max(x5);

Frequency\_shift5 = lag5(indice5);

figure(1)

subplot(2,3,1)

plot(abs(s1))

grid on

xlim([0 150])

title(' spectrum for Reference Word - Hello')

subplot(2,3,2)

plot(abs(s2))

grid on

xlim([0 150])

title(' spectrum for Reference word - Girl')

subplot(2,3,3)

plot(abs(s3))

grid on

xlim([0 150])

title(' spectrum for Test word')

subplot(2,3,4)

plot(abs(s4))

grid on

xlim([0 150])

title(' spectrum for Reference word - House')

subplot(2,3,5)

plot(abs(s5))

grid on

xlim([0 150])

title(' spectrum for Reference word - Cat ')

subplot(2,3,6)

grid on

plot(abs(s6))

xlim([0 150])

title(' spectrum for Reference word - Boy')

figure (2);

subplot(2,3,1)

plot(FA1)

grid on

xlim([0 150])

title('Normalized spectrum for Reference Word - Hello')

subplot(2,3,2)

plot(FA2)

grid on

xlim([0 150])

title('Normalized spectrum for Reference word - Girl')

subplot(2,3,3)

plot(FA3)

grid on

xlim([0 150])

title('Normalized spectrum for Test word')

subplot(2,3,4)

plot(FA4)

grid on

xlim([0 150])

title('Normalized spectrum for Reference word - House')

subplot(2,3,5)

plot(FA5)

grid on

xlim([0 150])

title('Normalized spectrum for Reference word - Cat ')

subplot(2,3,6)

grid on

plot(FA6)

xlim([0 150])

title(' Normalized spectrum for Reference word - Boy')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure (3);

subplot(2,1,1)

plot(x1)

title('Xcorr of Test word and Hello')

grid on

subplot(2,1,2)

plot(x2)

title('Xcorr of Test word and Girl')

grid on

figure(4)

subplot(2,1,1)

plot(x3)

title('Xcorr of Test word and House')

grid on

subplot(2,1,2)

plot(x4)

title('Xcorr of Test word and Cat')

grid on

figure(5)

grid on

subplot(2,1,1)

plot(x5)

title('Xcorr of Test word and Boy')

grid on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if indice1<length(x1)/2

q1 = 1:(indice1 -1);

p1 = indice1 + length(q1): -1 : indice1 + 1;

length(p1);

length(q1);

x1\_left = x1(q1);

min(x1\_left);

x1\_right = x1(p1);

min(x1\_right);

error1 = mean((abs(x1\_right - x1\_left)).^2);

else

q1 = 1 + Frequency\_shift1\*2: indice1 - 1;

p1 = length(x1): -1: indice1 +1;

length(p1);

length(q1);

x1\_left = x1(q1);

x1\_right = x1(p1);

error1 = mean((abs(x1\_right - x1\_left)).^2);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if indice2<length(x2)/2

q2 = 1:(indice2 -1);

p2 = indice2 + length(q2): -1 : indice2 + 1;

length(p2);

length(q2);

x2\_left = x2(q2);

min(x2\_left);

x2\_right = x2(p2);

min(x2\_right);

error2 = mean((abs(x2\_right - x2\_left)).^2);

else

q2 = 1 + Frequency\_shift2\*2: indice2 - 1;

p2 = length(x2): -1: indice2 +1;

length(p2);

length(q2);

x2\_left = x2(q2);

x2\_right = x2(p2);

error2 = mean((abs(x2\_right - x2\_left)).^2);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if indice3<length(x3)/2

q3 = 1:(indice3 -1);

p3 = indice3 + length(q3): -1 : indice3 + 1;

length(p3);

length(q3);

x3\_left = x3(q3);

min(x3\_left);

x3\_right = x3(p3);

min(x3\_right);

error3 = mean((abs(x3\_right - x3\_left)).^2);

else

q3 = 1 + Frequency\_shift3\*2: indice3 - 1;

p3 = length(x3): -1: indice3 +1;

length(p3);

length(q3);

x3\_left = x3(q3);

x3\_right = x3(p3);

error3 = mean((abs(x3\_right - x3\_left)).^2);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if indice4<length(x4)/2

q4 = 1:(indice4 -1);

p4 = indice4 + length(q4): -1 : indice4 + 1;

length(p4);

length(q4);

x4\_left = x4(q4);

min(x4\_left);

x4\_right = x4(p4);

min(x4\_right);

error4 = mean((abs(x4\_right - x4\_left)).^2);

else

q4 = 1 + Frequency\_shift4\*2: indice4 - 1;

p4 = length(x4): -1: indice4 +1;

length(p4);

length(q4);

x4\_left = x4(q4);

x4\_right = x4(p4);

error4 = mean((abs(x4\_right - x4\_left)).^2);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if indice5<length(x5)/2

q5 = 1:(indice5 -1);

p5 = indice5 + length(q5): -1 : indice5 + 1;

length(p5);

length(q5);

x5\_left = x5(q5);

min(x5\_left);

x5\_right = x5(p5);

min(x5\_right);

error5 = mean((abs(x5\_right - x5\_left)).^2);

else

q5 = 1 + Frequency\_shift5\*2: indice5 - 1;

p5 = length(x5): -1: indice5 +1;

length(p5);

length(q5);

x5\_left = x5(q5);

x5\_right = x5(p5);

error5 = mean((abs(x5\_right - x5\_left)).^2);

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ans = min([error1,error2,error3,error4,error5]);

errormat = [error1,error2,error3,error4,error5]

if ans == error1

display('the test word is the closest match to the first word - Hello')

elseif ans == error2

display('The test word is the closest match to the word - Girl')

elseif ans == error3

display('The test word is the closest match to the word - House')

elseif ans == error4

display('The test word is the closest match to the word - Cat')

elseif ans == error5

display('The test word is the closest match to the word - Boy')

end

% figure(6)

% subplot(5,1,1)

% plot(Test1)

% title('SNR 50')

% grid on

% subplot(5,1,2)

% plot(Test2)

% title('No AWGN')

% grid on

% subplot(5,1,3)

% plot(Test3)

% title('SNR 10')

% grid on

% subplot(5,1,4)

% plot(Test4)

% title('SNR 1')

% subplot(5,1,5)

% grid on

% plot(Test5)

% title('SNR 0.1')

% grid on