EE 473

Semester Project Report

IMPLEMENTATION OF SPEECH ENHANCEMENT ALGORITHMS

Submitted by: Egemen Erbayat

Ali Umut Kaypak

Melih Baştürk

04.01.2022

1- Problem Statement

Microphones, noise in the channel, environment, or speech reproduction decrease the quality of speech. When degradation in the speech increase is high, the receiver cannot understand the speech. Implementing state-of-art speech enhancement algorithms to improve the speech which suffers from degradations and noises is the purpose of this project. We mainly focused on two algorithms, which are spectral subtraction method for single-channel and adaptive noise cancelling for two channels.

2- Introduction

The reason for the decrease in the speech quality may be various such as distortion from voice coding algorithms, multi-speakers in the environment, or noise from different sources. Since the degradation sources are various, the different speech enhancement algorithms focus on different speech characteristics and assume different constraints for different cases. Therefore, it is difficult to determine the best algorithm which fits all different speech enhancement scenarios. Although assuming different constraints may be fruitful for a lot of cases, it can cause the algorithm to be fragile for other scenarios. Therefore, we aimed to research different constraints on the speech and speech enhancement techniques for these different constraints. After the research, we implemented the best two algorithms that are suitable for our possible scenarios for single and dual-channel cases.

In this project, we mainly focus on speech enhancement for the single-channel and two channels. We were first going to do research on different methods; after the research, we chose a method that fits single-channel speech enhancement and another method that fits dual-channel speech enhancement. After that, we tried to implement these algorithms in both MATLAB and Python environments. After the implementations, we created a GUI for the user to use these algorithms and to observe results easily.

Firstly, we worked with Short-Term Spectral Magnitude Methods, where a noise bias is estimated during non-speech activity or from the reference channel [1], and then this noise is reduced from the speech. Although the name 'spectral' suggests that the subtraction is made in the frequency domain, there are also various techniques in which the subtraction is implemented in the time domain, for example, 'correlation subtraction' [1]. We made our implementation in the frequency domain. Next part, we will give more details about the implementation and mathematical background of spectrum subtraction method. The mathematical background in this method can be found in [2]. Spectral subtraction offers computational efficiency [3]. Therefore it can be reasonable to choose these types of methods when CPU usage is an important constraint.

Our second method is the LMS adaptive filter, one of the Adaptive Noise Cancelling algorithms, which are widely used in other areas which require noise cancellations [1]. In this type of algorithms, there exist two types of signals: a primary signal which is noisy speech in our case, and a reference signal which is correlated somehow to noise in the speech. We will again give more information about this method and its implementation in the next section. Also, more information on adaptive noise canceling can be found in [4]. A complex noise canceled based algorithm which suggests using Neural Networks for noise canceling may be investigated for later projects [5].

3-Methodology and Results

There are a lot of speech enhancement methods and techniques for noise reduction. Even though these methods have many different approaches, they can be classified according to some main features. Some of these classes are "Short-Term Spectral Amplitude Techniques", "Speech Modeling and Wiener Filtering", "Adaptive Noise Cancelling", and "Systems Based on Fundamental Frequency Tracking". We have considered two of these classes: "Short-Term Spectral Amplitude Techniques" and "Adaptive Noise Cancelling". We have implemented the spectral subtraction method from short-term spectral amplitude techniques, and we worked with the LMS adaptive filter from adaptive noise cancelling techniques.

We tried to implement these algorithms in both MATLAB and Python. In MATLAB, we made some simulations by adding Additive White Gaussian Noise (AWGN) at different signal-to-noise ratios and observed SNR enhancement at each SNR level for single-channel speech. In Python, we implemented both algorithms and created an interface to select which method will be used, which sound file will be enhanced, to play input and output sounds and similar features. Before going deep into these parts, we should mention the mathematical background of our speech enhancement methods.

3.1- Short-Term Spectral Amplitude (STSA) Techniques

In these techniques, noise reduction can be achieved by subtracting an estimated noise power spectral density(PSD) from the PSD of speech signal. All of these methods perform noise reduction steps on spectral magnitude. This process can be summarized as follows: firstly, short term amplitude and phase of the noisy spectral speech signal is obtained, then estimated noise magnitude spectrum is subtracted from the obtained amplitude of speech signal, and finally, inverse Fourier transform is taken by using the phase of noisy speech signal and magnitude of subtraction result signal. However, the primary process is similar for most of the approaches, and there are again different approaches and formations to reduce noise in STSA. One of them is spectral subtraction. Firstly, we chose this method to implement and to reduce the noise. At this method, speech is modeled as a random process, and noise is assumed to be a wide-band stationary additive noise. Therefore, we first recorded some noise free speech and added some wide-band stationary

additive noise such as Additive White Gaussian Noise (AWGN). Mathematically this process also can be written as follows:

$$\boldsymbol{y(n) = s(n) + d(n)}$$

y(n): noisy speech signal, s(n): original speech signal and d(n) noise. Here noise process d(n) is assumed to be uncorrelated. Then we can write this equation in terms of power spectral densities (PSD) as follow:

$$\Gamma_y(\omega) = \Gamma_s(\omega) + \Gamma_d(\omega),$$

From this equation, we know the received signal and its PSD, and also we can estimate the PSD of noise because we have assumed that noise is wide-band stationary additive noise. Thus, we can obtain PSD of the original speech signal as follow:

$$\hat{\Gamma}_s(\omega) = \Gamma_y(\omega) - \hat{\Gamma}_d(\omega).$$

However, this method is not appropriate for realistic conditions where the received and original speech signals are not stationary. Therefore, we used 25 ms hamming windows (see Figure3.1) to obtain locally stationary signals over short time ranges.

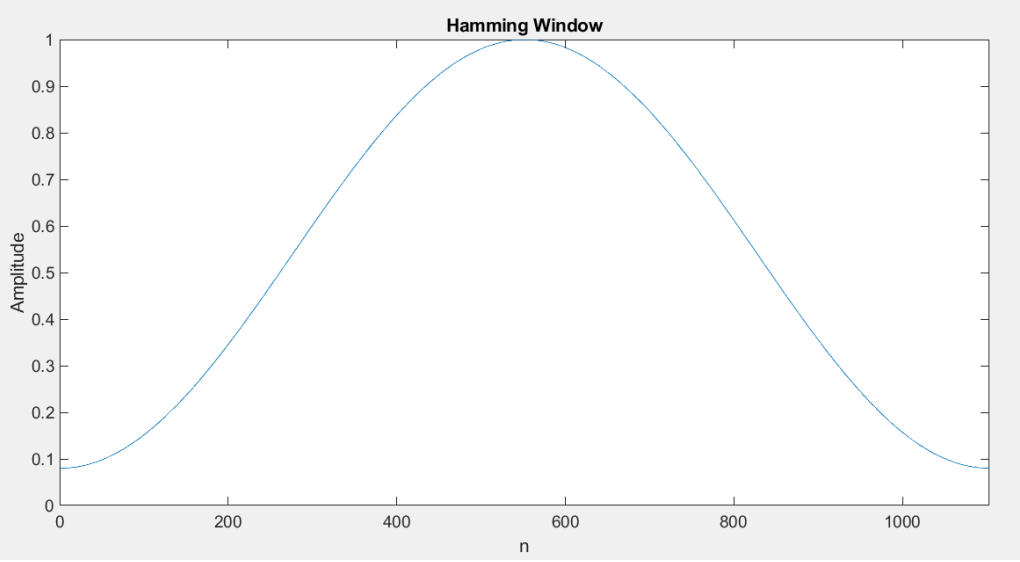

Figure3.1: Our Hamming window used in the implementation

In the implementation of this algorithm, for reference noise, we took the first window of noisy speech. Then, we took the Fast Fourier Transform of all noisy speech and reference noise using windows with 25ms. After taking FFT of both noisy speech and noise, we kept the magnitude and phase of these signals separately. We subtracted the magnitude square of noise spectrum with a β factor from the magnitude square of speech spectrum:

$$|\boldsymbol{S_m(w)}|^2 = |\boldsymbol{Y_m(w)}|^2 - \boldsymbol{\beta} \times |\widehat{\boldsymbol{N}_m}(\boldsymbol{w})|^2$$

Then we obtained the FFT of the noise-free spectrum by taking the square root of the subtraction result and multiplying it with the phase of noise speech. Finally, we obtain noise-free speech by taking the real part of the inverse Fast Fourier Transform (IFFT) of the noise-free spectrum. After making this noise reduction for each window, we used a Butterworth lowpass filter with cutoff frequency 2500 Hz, which is determined according to speech frequency interval, and order 4 to get rid of musical noise artifact, which is a general problem in speech noise reduction algorithms as a result of spectral subtraction [6]. All of these operations were made frame by frame because speech is not a stationary process, and therefore we should look locally stationary over short intervals. Also, we used the 50% overlap-add for hamming window method, which is used to break long signals into smaller parts, in this algorithm. In the end, by adding overlapped windows, we obtained total output speech. The Block diagram in Figure3.2 summarizes our main implementation steps. The " Results " section mentions how well this method works and which level we can reach in SNR enhancement, etc.

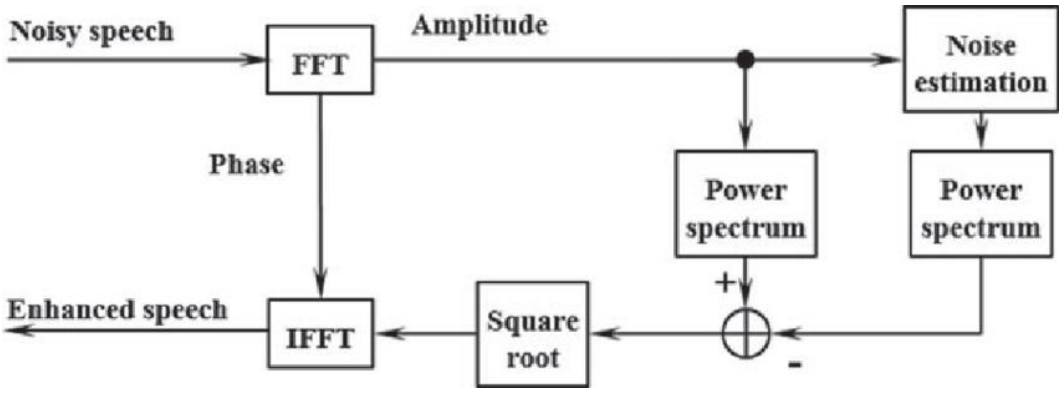

Figure 3.2: Spectral Subtraction Method. Source: [1]

3.2- Adaptive Noise Cancelling (ANC)

Our second method is LMS adaptive noise cancelling. Firstly we want to mention generally adaptive noise canceling (ANC) methods. In addition to speech enhancement, these methods are also used for electrocardiography, elimination of periodic interference and adaptive antenna theory[1]. ANC techniques reduce unwanted background noise by using a noise-cancelling method. ANC algorithms were developed based on two channels. One of them is primary channel takes a noisy signal source. The other is a reference source that only takes noise as an input. The flow diagram is shown in Figure 3.3. In this block diagram, reference noise d_2 may be correlated with d_1, however not the noise-free signal s.

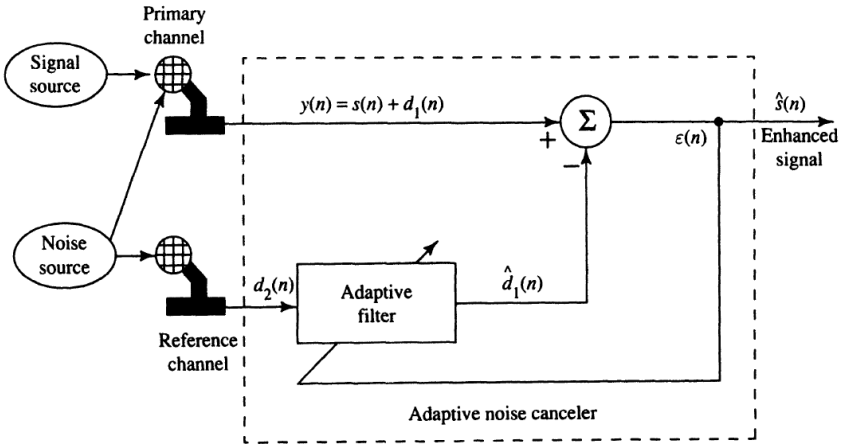

Figure 3.3: Flow Diagram of Adaptive Noise Canceling. Source: [1]

Adaptive noise cancelling methods use an adaptive filter on reference noise, as seen in Figure3.4, to estimate noise. Then this estimated noise is subtracted from the noisy signal. Coefficients of adaptive filter are continuously changing such that minimizes mean square energy of noise difference, which can be written as d= d_1- $\widehat{d_1}$(n). However, we can never know noise in the noisy speech signal. Therefore we cannot minimize MSE of d. Therefore, instead of minimizing MSE of d, there is no problem with considering MSE of the estimated signal because s and d_2 are uncorrelated. This method is also known as least mean square error (LMS).

We have implemented LMS adaptive noise cancelling method. The flow diagram of the LMS adaptive noise cancelling method is given in Figure3.4. In this method, we do not need a priori information of noise. However, we need a reference noise that is correlated with noise in the speech signal. Because of this need, we have made recordings with two devices in a noisy place, such as the Bogazici University cafeteria. One of these two devices is located near the talker, the primary channel, and the other is located two meters further from the talker, which is the reference channel, such that only noise in space is recorded.

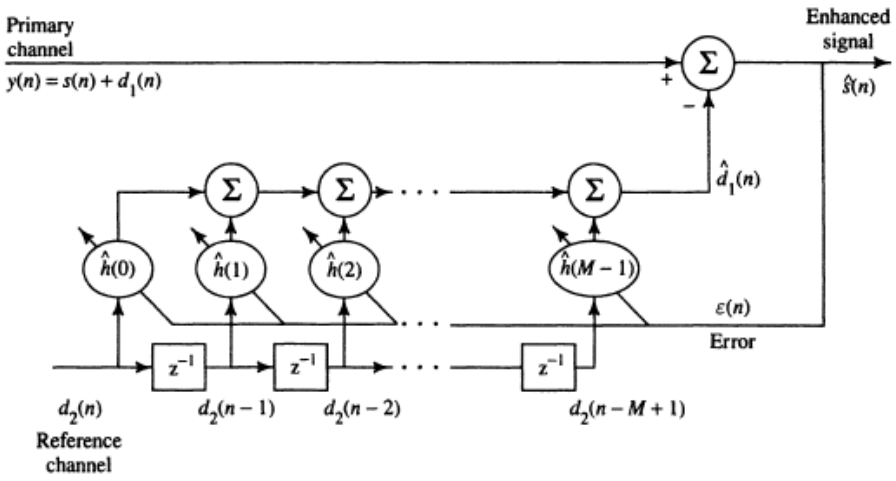

Figure 3.4: Flow Diagram of LMS Adaptive Noise Canceling. Source: [1]

Now we can go into the algorithm. There is an adaptive filter, which is an FIR filter with the estimated coefficients $\hat{h}$(i) where I=0, 1, 2, …, M-1 with the order M to obtain estimated noise, which then subtracted from signal comes from the primary channel. This filter uses filter coefficients at a moment ago to adjust filter parameters at present to achieve optimal filter operation. This estimated error can be expressed as follow:

$$\hat{d}_1(n) = \sum_{i=0}^{M-1} \hat{h}(i) d_2(n-i).$$

After that, we tried to minimize the mean square error between the original noise (d_1(n)) and estimated noise ($\widehat{d_1}$(n)). However, as previously mentioned, we do not have any information about noise in the speech signals; therefore, we cannot minimize this error. Instead of this, the energy of the estimated speech signal is minimized by exploiting the fact that s and d_2 are uncorrelated.

In the implementation of this method, firstly, our algorithm takes the two signals: noisy speech signal and reference noise, which are recorded by two different devices at the same time and space, as mentioned before. Similar to the spectral subtraction method, we again consider a windowed version of noisy speech signal and reference noise because they are not stationary in real life. Again we used 25 ms hamming windows. In each window, we shifted reference noise and multiplied it with adaptive filter coefficients to obtain estimated noise ($\widehat{d_1}$(n)). After that, we subtracted estimated noise from the noisy speech signal to get a noise-free speech window. Also, at each window, our algorithm updates adaptive filter coefficients with a learning parameter μ. Here order M and learning parameter μ are taken from the user according to the noise power, noise type and etc. We update the filter coefficients according to this equation:

$$\mathbf{h_{n+1}} = \mathbf{h_n} + \mathbf{2} \times \boldsymbol{\mu} \times \left(\widehat{\mathbf{s(n)}} \times \widehat{\boldsymbol{d_1}}(\boldsymbol{n}) \right)$$

The mathematical background of this equation can be summarized as follow:

1) We aim to obtain adaptive filter coefficients that minimize the noise difference between estimated noise and noise in a speech signal. This idea can be written as:
$$\mathbf{h_{n+1}} = \mathbf{h_n} + \mathbf{2} \times \boldsymbol{\mu} \times \mathbf{E}[\widehat{\mathbf{s(n)}} \times \widehat{\mathbf{d_1(n)}}]$$
However, the expected value of $\widehat{\mathbf{s(n)}} \times \widehat{\mathbf{d_1(n)}}$ is not known. Therefore, this is not useful.
2) Instead of this, we can take sample mean:
$$\mathbf{h_{n+1}} = \mathbf{h_n} + \mathbf{2} \times \boldsymbol{\mu} \times \frac{\mathbf{1}}{\mathbf{L}} \sum_{\mathbf{0}}^{\mathbf{L-1}} \widehat{\mathbf{s(n-l)}} \times \widehat{\mathbf{d_1(n-l)}}$$
3) After that, we used L = 1, and we obtained:
$$\mathbf{h_{n+1}} = \mathbf{h_n} + \mathbf{2} \times \boldsymbol{\mu} \times \left(\widehat{\mathbf{s(n)}} \times \widehat{\boldsymbol{d_1}}(\boldsymbol{n}) \right)$$

Similar to spectral subtraction method, all of these operations were made frame by frame because speech is not a stationary process and therefore, we should look locally stationary over short intervals. In addition, we used 50% overlap-add for hamming window method, which is used to break long signals into smaller parts, in this algorithm. In the end, by adding overlapped windows, we obtained total output speech.

After the implementation steps, we test this algorithm with some recordings, which are taken by two devices. We will mention more detailed results of both algorithms in the results section. Before the results, we want to say a little about our interface.

4- Interface User Manuel

We created an interface to ease use and see changes in sounds. While designing the interface, we used Python 3.8 release. In the interface, firstly user sees a window like in Figure4.1. On this page, the user can choose channel type according to speech.

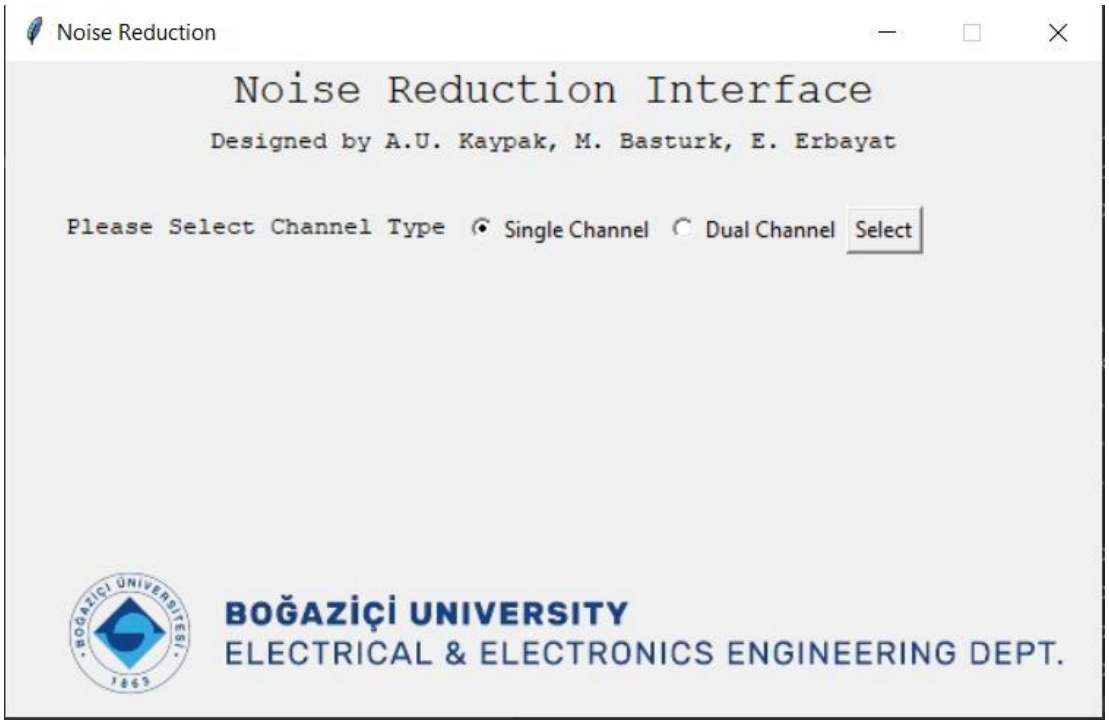

Figure4.1: Opening Page of our Interface

If the user selects to use single-channel, where spectral subtraction algorithm is used, the page in Figure4.2 is opened. On this page, the user should choose a speech file and an output path. In addition to these, the user should write the output file name without its extension, the default extension is wav. Also, the user can specify the β value, which is the coefficient of subtracted noise, and it should be positive. After all these parameters are specified, the user can execute the program for single-channel speech. For single-channel, the program executes the spectral subtraction method.

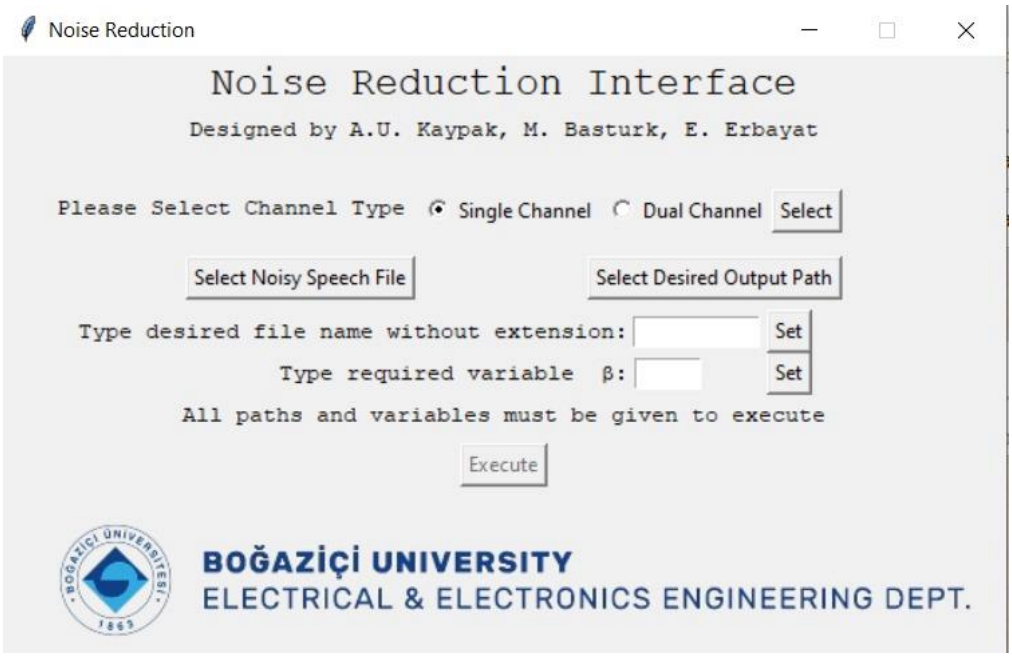

Figure4.2: Single Channel Page

After the execution, the page in Figure4.3 will be opened to play noisy speech signal and noise-reduced output speech.

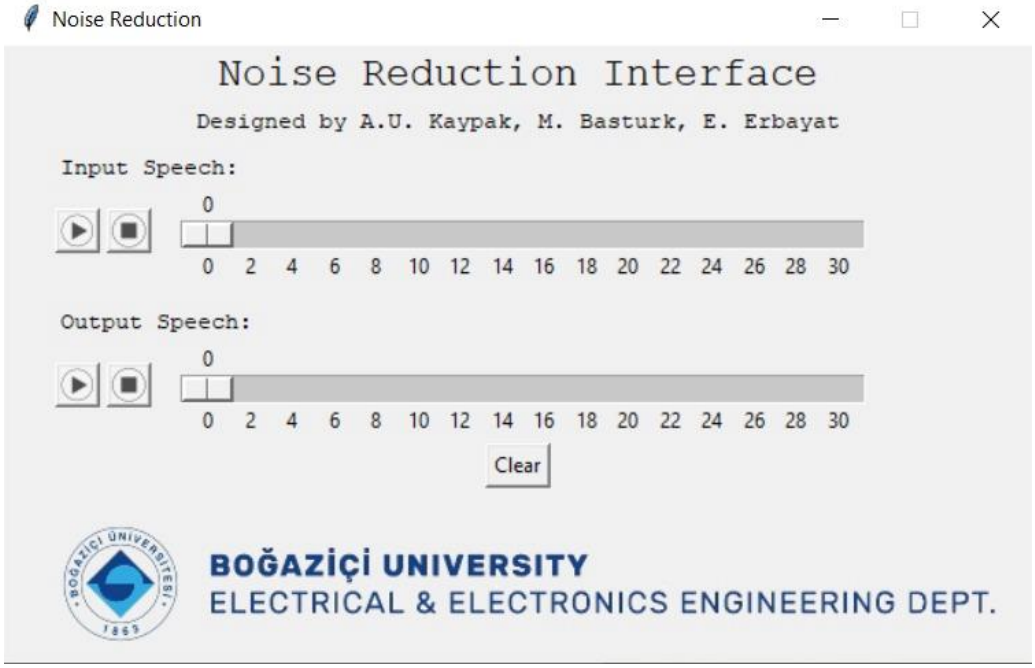

Figure4.3: Play Window

If the user chooses a dual-channel, which uses adaptive noise canceling algorithm, the page in Figure4.4 will be opened on the opening page. Unlike the single-channel case, the user should choose a reference noise file. Also, there are two parameters which are μ and order of the adaptive filter. μ should be between 0-1, and order should be a positive integer number. After execution again page at Figure4.3 will be opened.

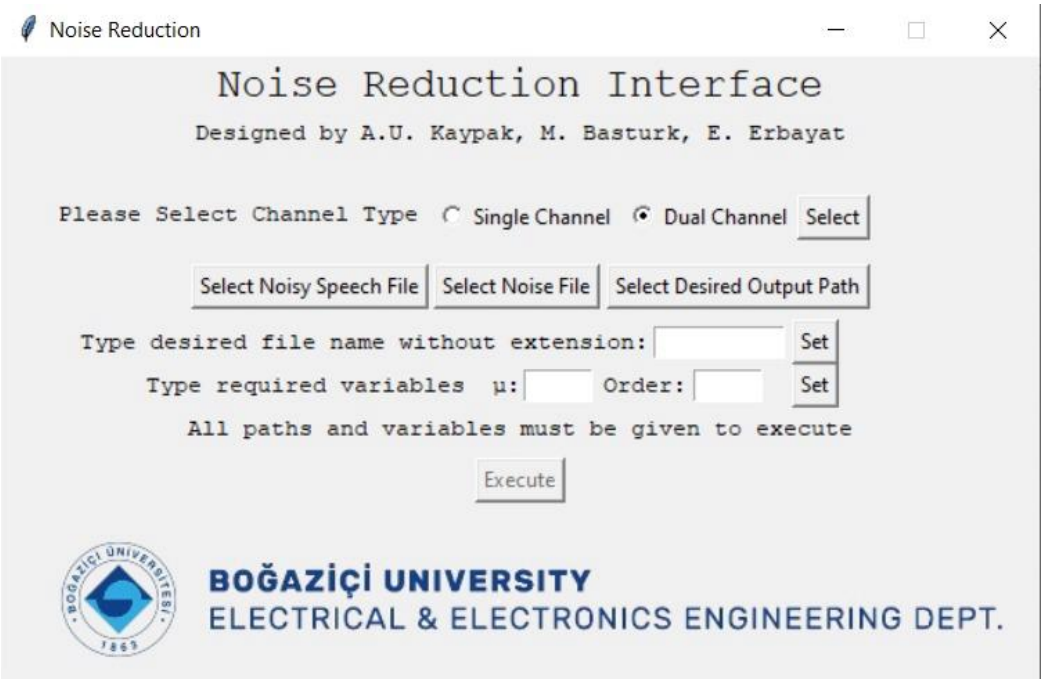

Figure4.4: Dual Channel Page

5- Results

To test our algorithms, we recorded different speech with and without noise. We do not need an extra noise record in spectral subtraction because this algorithm takes the first window of noisy speech signal as reference noise. On the other hand, we need a noise file as a reference noise for adaptive noise cancelling, which should be at least a little correlated with the original noise. These reference noises were recorded by using another device at the same time and space.

We have observed that both methods effectively enhance speech enhancement by using noise reduction. Also, we observe that adaptive noise cancelling is a better method for speech enhancement. We added our original speech files and enhanced speech files in project documents.

In addition to sound files, we have made simulations with β=10 in MATLAB for a single channel spectral subtraction algorithm at different signal-to-noise ratio values from -20 dB to 20 dB. For example, in Figure 6.1, you can see noise-free original speech signal, noisy signal with 5 dB SNR level and enhanced signal waveforms. From this figure, we can easily observe that noise effects and speech enhancement effects the speech signal. Also, in Figure6.2-3-4, you can see spectrogram graphs of these signals, respectively. Again, we can clearly see noise and spectral subtraction effects on the speech signal from these spectrograms.

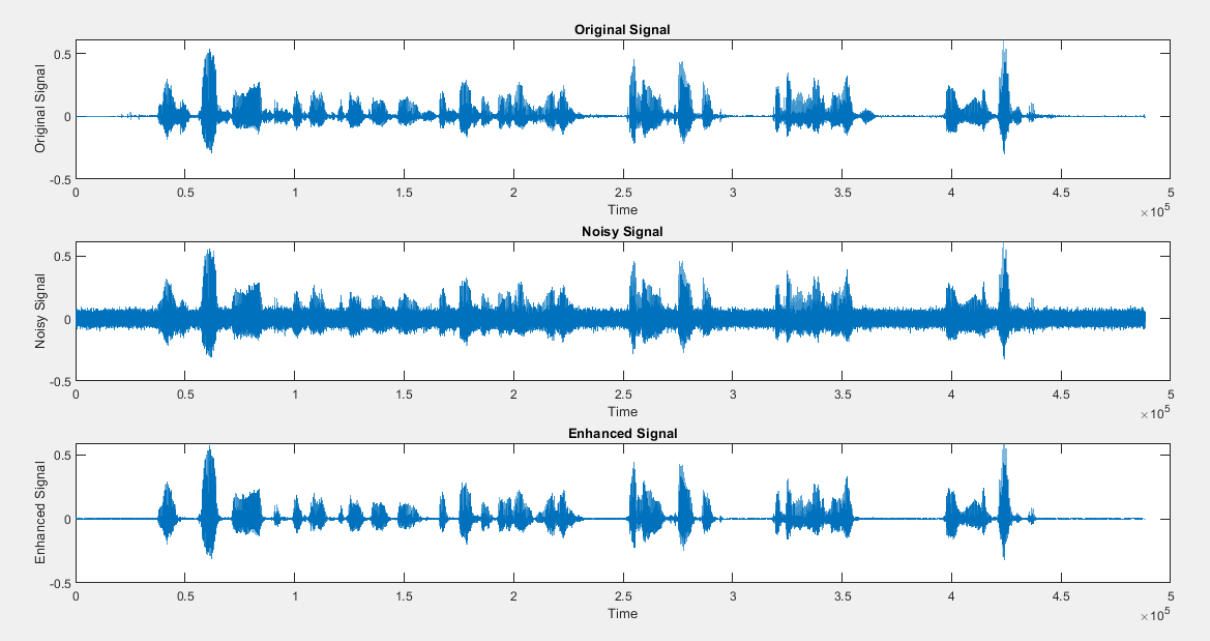

Figure6.1 Waveforms of noise free speech, noisy speech and enhanced speech

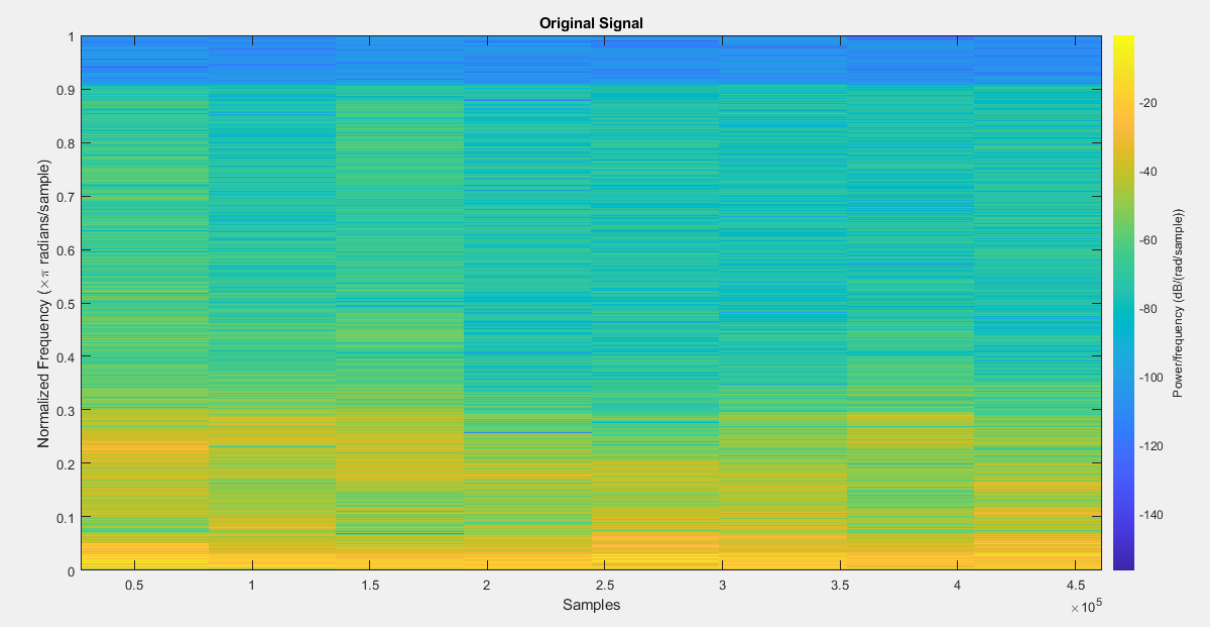

Figure6.2 Spectrogram of noise free speech signal

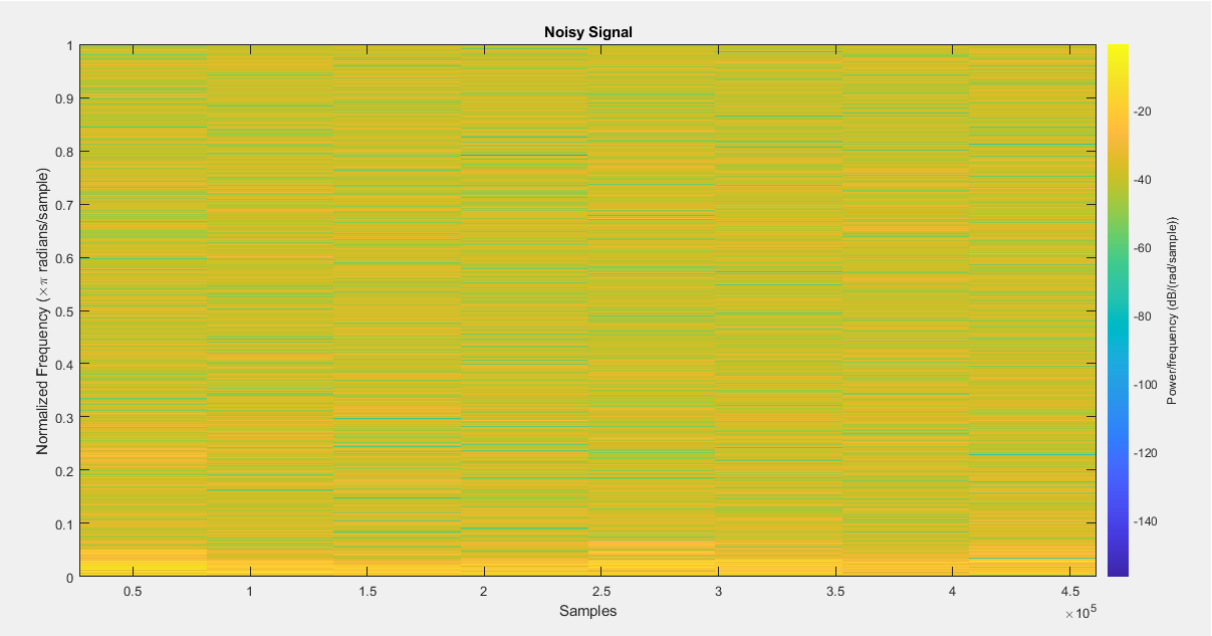

Figure6.3 Spectrogram of noisy signal

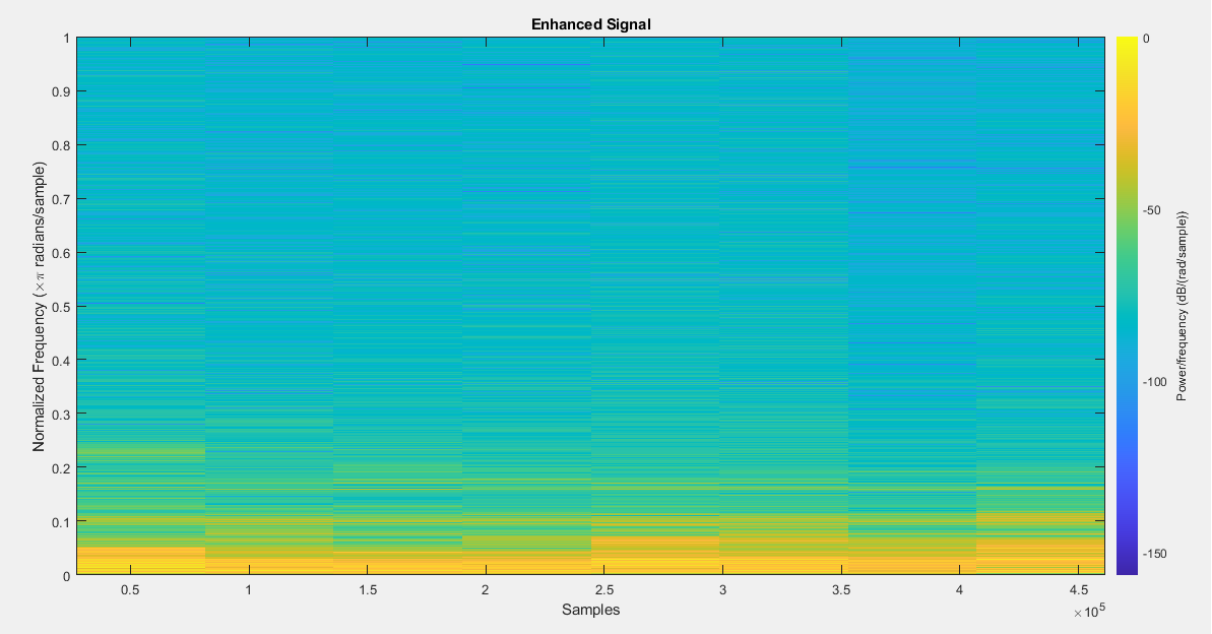

Figure6.4 Spectrogram of enhanced speech signal

If we return simulation with different SNR levels, in this simulation, we added additive white Gaussian noise (AWGN) according to SNR levels to noise-free speech signal. And we looked at SNR levels of enhanced speech. We ran our code 50 times to observe smooth curves and took the average value. Results can be seen in Figure6.5. From these plots, we can say that our algorithm nearly supplies 18dB enhancement at -20dB input SNR level. As input SNR value increases, SNR enhancement decreases, as expected. These results show us that our spectral subtraction algorithm perfectly works.

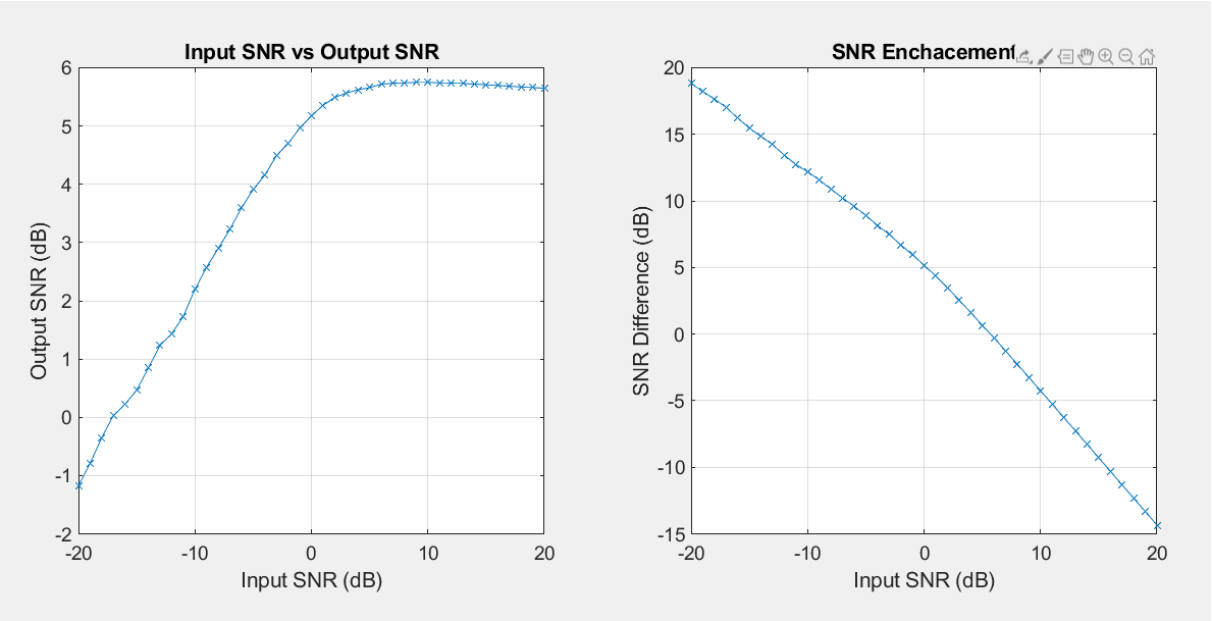

Figure6.5: Input and output SNR comparison

On the other hand, in Figure6.6, you can see the noisy speech signal and the enhanced signal by using an adaptive filter algorithm. We can easily observe speech enhancement effects on the speech signal from this figure. Also, in Figure6.7-8, you can see spectrogram graphs of these signals, respectively.

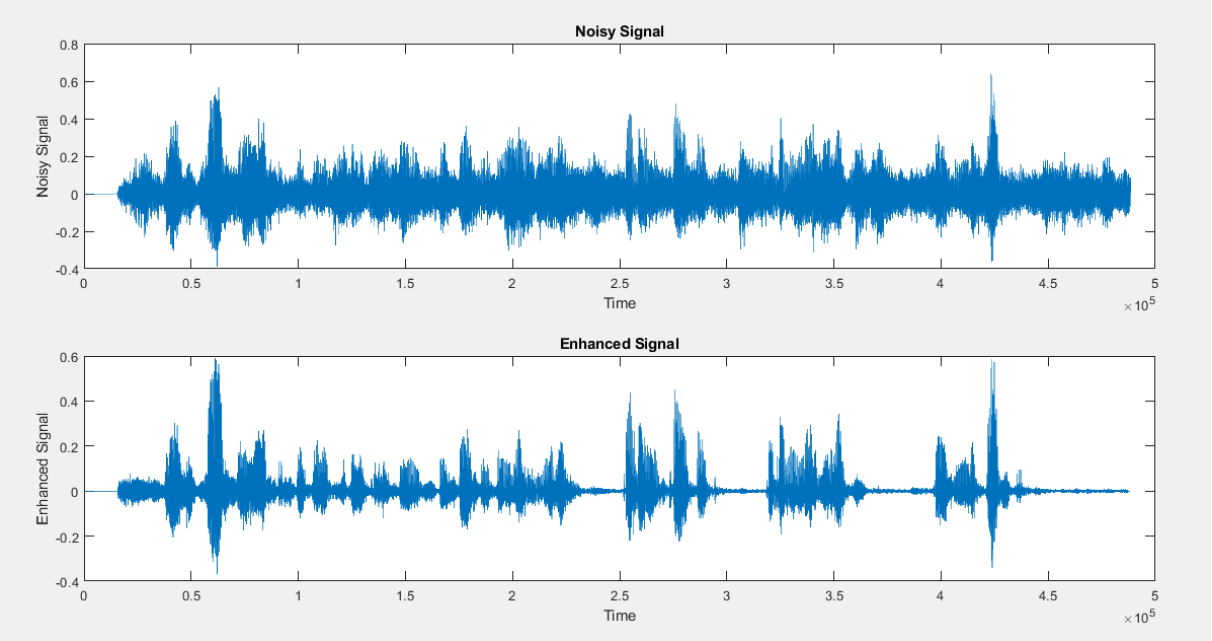

Figure6.6 Waveforms of noisy speech and enhanced speech

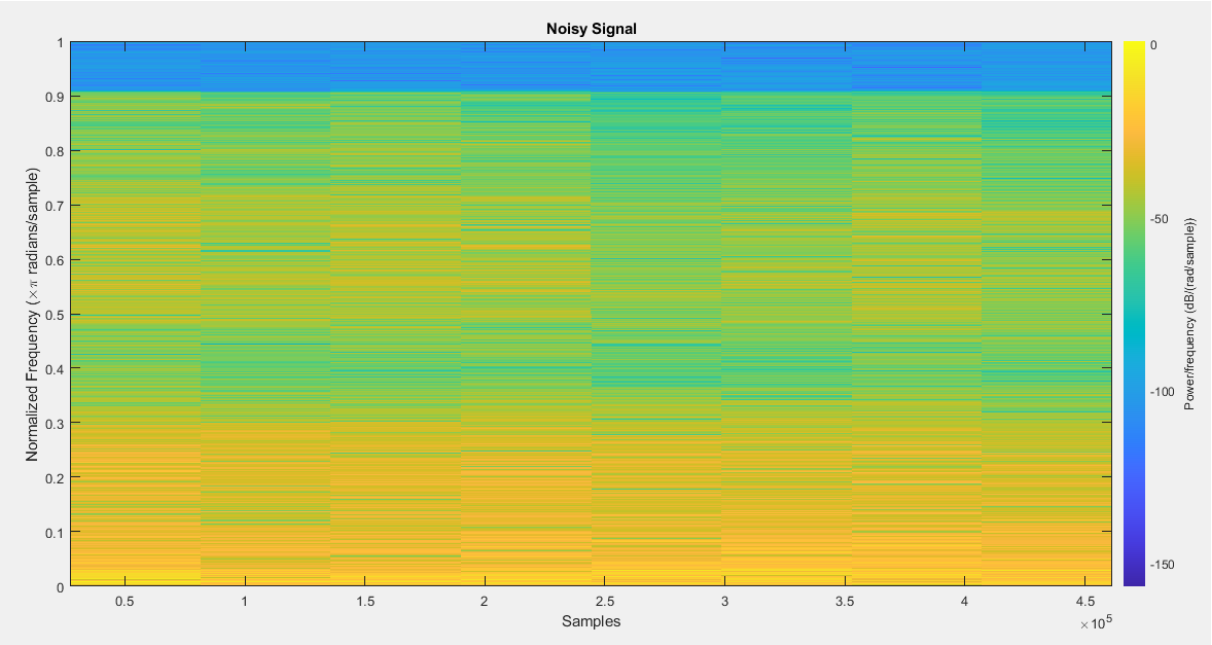

Figure6.7 Spectrogram of noisy signal

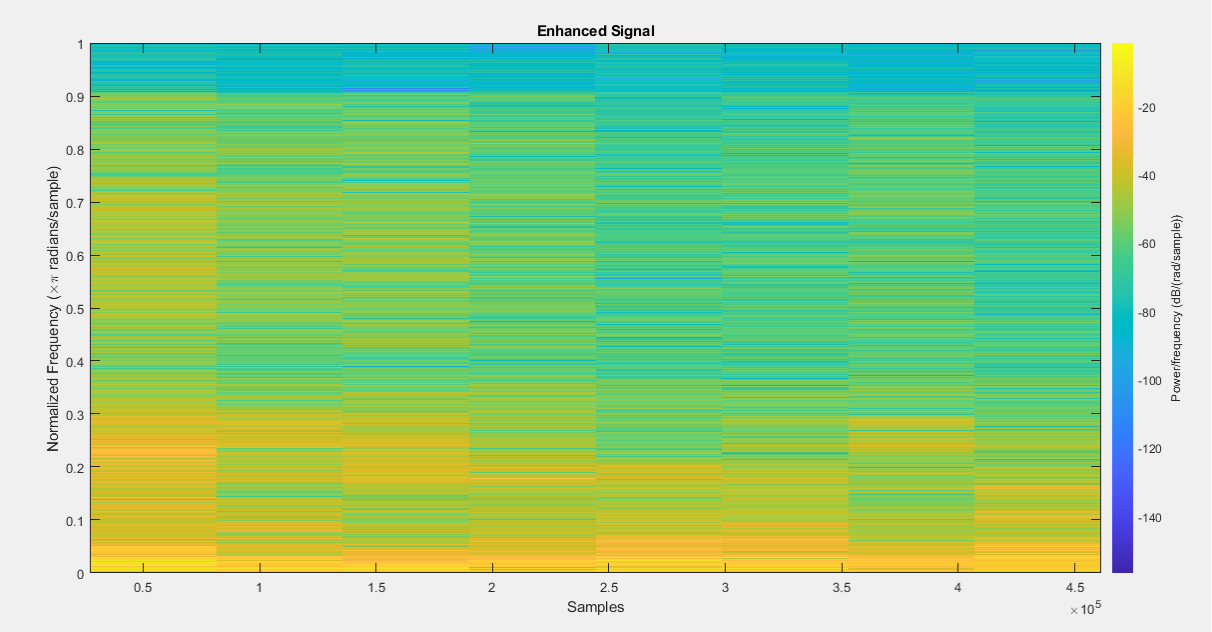

Figure6.8 Spectrogram of enhanced speech signal

6- References:

[1] Deller Jr, John R., John G. Proakis, and John H. Hansen. Discrete time processing of speech signals. Prentice Hall PTR, 1993.

[2] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete Fourier transform," in *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235-238, June 1977, doi: 10.1109/TASSP.1977.1162950.

[3]S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," in *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 2, pp. 113-120, April 1979, doi: 10.1109/TASSP.1979.1163209.]

[4] B. Widrow *et al*., "Adaptive noise cancelling: Principles and applications," in *Proceedings of the IEEE*, vol. 63, no. 12, pp. 1692-1716, Dec. 1975, doi: 10.1109/PROC.1975.10036.

[5] W. G. Knecht, M. E. Schenkel and G. S. Moschytz, "Neural network filters for speech enhancement," in *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 6, pp. 433-438, Nov. 1995, doi: 10.1109/89.482210.

[6] Lukin, Alexey and Jeremy G. Todd. "Suppression of Musical Noise Artifacts in Audio Noise Reduction by Adaptive 2-D Filtering." Journal of The Audio Engineering Society (2007)