UNIVERSITY OF CALIFORNIA, SANTA CRUZ
EE 153 SPRING 2017

Acoustic Echo Cancellation and Adaptive Filters.

Luz Lule Camargo, Tin Thurein

Project Members

Luz Lule Camargo

Undergraduate Electrical Engineer- University of California, Santa Cruz
Luz is interested is interested in analog circuit design. She is also interested in DSP chip design for Deep Learning applications.

Tin Thurein

Undergraduate Electrical Engineer – University of California Santa Cruz
Tin is an undergraduate student who is pursuing Electrical Engineering with a concentration in Electronics and Optics. He has an interest on analog and digital circuit design and simulation, along with digital signal processing.

ABSTRACT

The aim of this project is to use Matlab to implement an Echo-cancellation filter in audio speech signals. Echo suppression and echo cancellation have improved the quality of phone and video communications greatly. We focus on applying the Least mean squares (LMS) algorithm to implement an adaptive filter in MatLab that will cancel out the echo in recorded audio signals. This paper explores the characteristic of the adaptive filter used for acoustic Echo-Cancellation.

INTRODUCTION

Echo is a phenomenon where a delayed and distorted version of an original sound is reflected back to the original source. Echo occurs when speech is reflected of the walls, floor and other neighboring objects that is then picked up by the mic. This phenomenon presents great challenges when trying to communicate over the phone or video chats. In telecommunications, there are two types of echo where the source of the echo could be electrical or acoustic. Electrical echo is due to the impedance mismatch at the hybrids of a Public Switched Telephony Network, (PSTN), exchange where the subscriber two-wire lines[1]. In our project, we are not interested in electrical echo, instead we focus on acoustic echo. One reason for focusing acoustic echo is because acoustic echo cancellation uses DSP algorithms and second reason is the increase in need for acoustic echo cancellation for devices such as hands free communication devices in cars. Acoustic echo cancellation has become essential for the efforts to improve the quality of devices used for telecommunications.

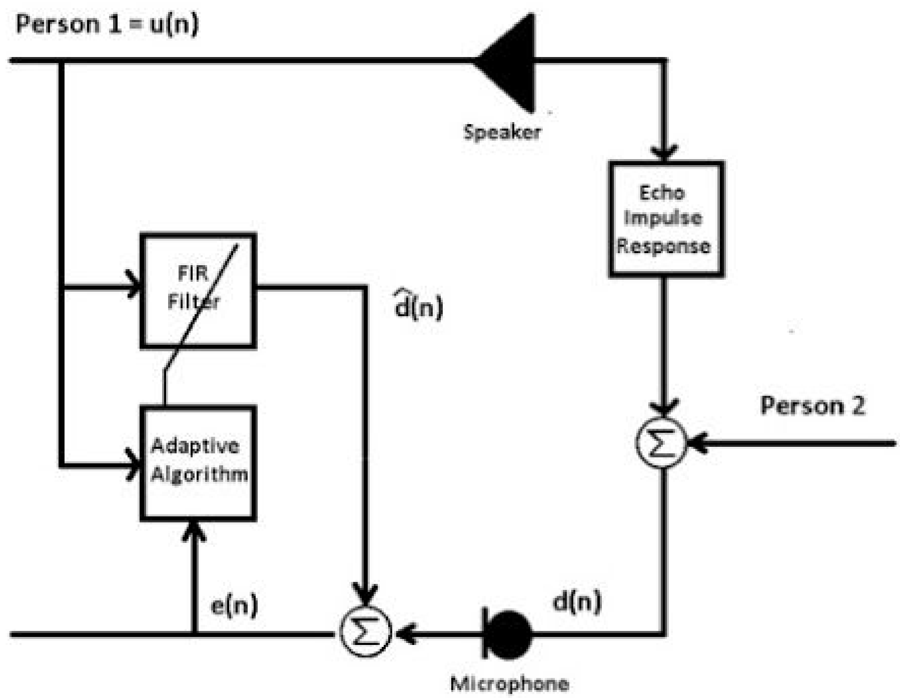

Figure 1. Block diagram of the Echo cancelling System.

The algorithm we use to create an echo cancellation system required the implementation of an adaptive filter. The filter must match the filter that caused the change of the audio signal as it goes through the speakers, reflects off the room then it's eventually picked up by the mic. In the block diagram in figure 1, we can see that u(n) represents the audio that will be played in the speaker, d(n) is the signal which consist of person 2 speaking as well as the echoed signal. $\widehat{d(n)}$ represents the model of the echo and e(n) = d(n)- $\widehat{d(n)}$. This means that e(n) will give us the sound of person 2 speaking. For the implementation of this algorithm in our project, e(n) gives us the direct sound without the delayed and attenuated audio signals that are introduced in our original recorded signal.

Problem Specification

Our goal is to implement an active echo cancelling filter in matlab, where we can verify the filter's functionalities with different characteristics of sound signals. As a starting point, we implemented a mathematical model of Least Mean Square Algorithm, where it takes the values of the previous sample points, and minimize the error to be zero. With this we were able to output mean error values and display it, however, since this is a non-adaptive filter, it was rather difficult to implement on actual speech signals. Therefore, we shifted our approach to adaptive filter.

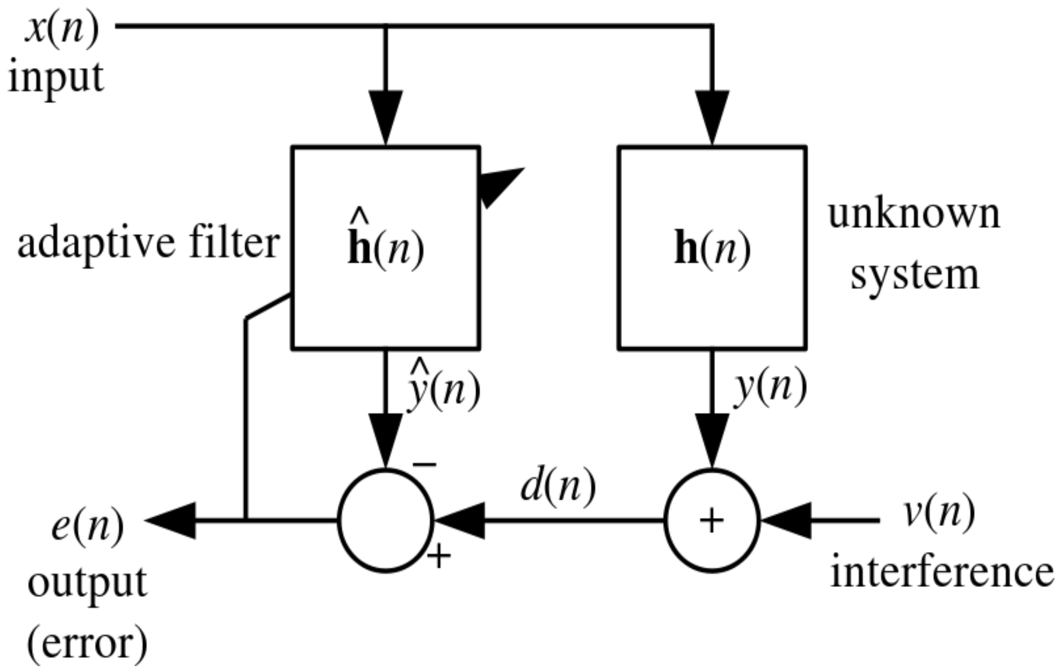

Figure 2. An implementation of Least Mean Square Algorithm

From the Figure 2, we can see that it follows the same process as the echo cancellation system diagram. The problem with the implementation of the LMS algorithm came with first trying to figure out what MATLAB equations would be useful for this project. The problem we were trying to address is disruptive echo in audio signals that were pre-recorded. The aim of this project was to be able to visualize and hear the echo attenuation between the original recorded audio and the processed audio.

Signals

We used both pre recorded echoed signal, and synthesized signal to test how well our filter functions and observe the signal before and after the filter. We also used built-in matlab recorded signal such as “near-end speech” and “far-end speech”, where near-end speech is the speech that goes directly into the mic without any other signals. The Far-end speech is where the echo of the speech signal. Using those and a few other speech signals, we were able to observe the outcome of our filter. In the second stage of our project, we used a signal that we obtained from soundbible.com. The audio sample was just a recording of the word “Hello” and its subsequent echo.

Figure 3, shows the comparison of the outcomes of using different learning parameters μ. We used this graph to determine which μ was best for our system. This μ was used to get rid of the synthetic echo signal that we created. We could hear the improvement in echo detection and cancellation in the synthetic echo signal due to the fact that that signal was of better quality. However, the improvement in echo cancellation can be better observed with the plots of the simple “echo signal” as opposed to the more complex audio signal used in the synthetic echo trial.

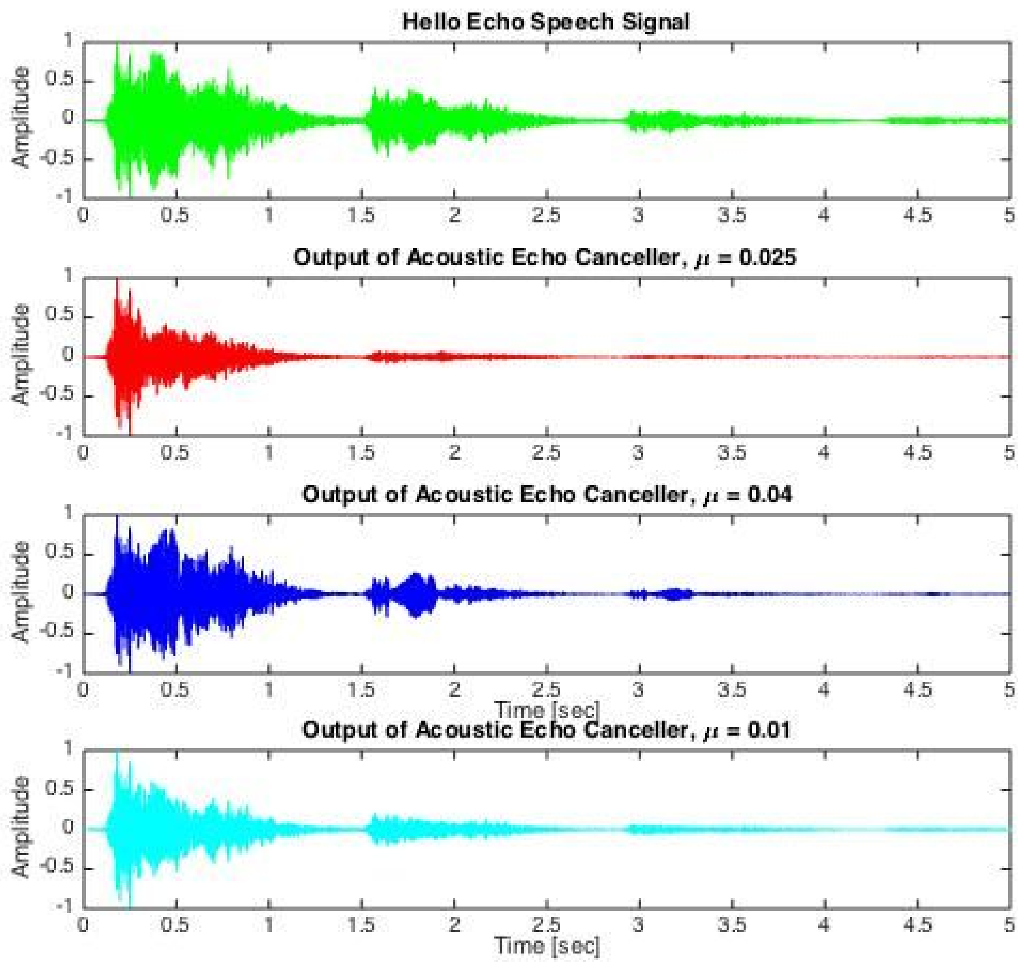

Figure 3. Comparison of learning parameters for adaptive filter.

Approach

As a first step, we implemented a simple non-adaptive filter using least mean square [LMS] FIR method based on mathematical formula described below. It is done by first convoluting the known signal y[n] and estimated impulse response h[n], in order to obtain estimated x[n].

$$\hat{x}[n] = h[n] * y[n] = \sum_{i=0}^{N} h_i y[n-i],$$

Here, the error e[n] is defined as such:

$$e^2[n] = (x[n] - \hat{x}[n])^2 = (x[n] - \sum_{i=0}^{N} h_i y[n-i])^2,$$

Taking derivative of $e^2[n]$ tells us how to adjust h_i.

$$\frac{de^2[n]}{dh_j} = -2\left(x[n] - \sum_{i=0}^{N} h_i y[n-i]\right) y[n-j] = -2e[n]y[n-j].$$

We wish to minimize the $e^2[n]$, therefore we can do so by reducing h_i.

To obtain h[n], we first "train" with known signal x[n] and observe the output y[n], where $H(z) = X(x)/Y(z)$. While it is easy to implement in Matlab, it wasn't quite useful for our application since this is non-adaptive filter. Therefore, we attempted to implement an active filter.

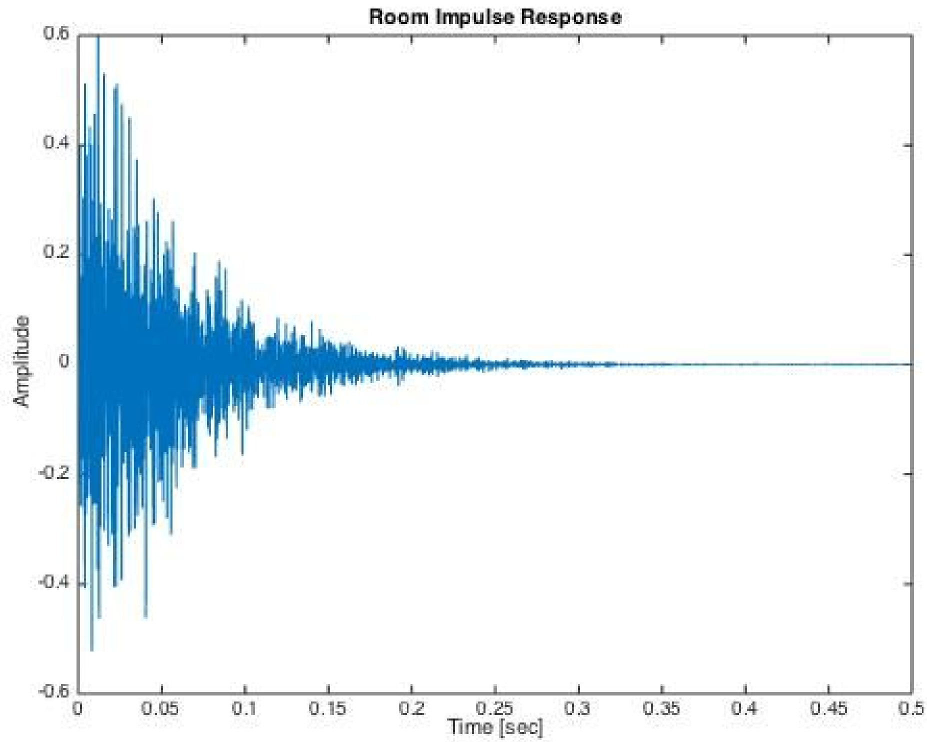

Figure 4: Room Impulse Response in Matlab

To implement an active echo cancelling filter, we needed to find the room impulse response. The room impulse response tells us how the sound is being reproduced in the room with arbitrary dimension and size. Fortunately, Matlab already has built in room impulse response generator that we can use to obtain how the sound is being reproduced. As we can see from the figure above, we have amplitude with Time in the x-axis. What this tells us is that the sound bounces back from the wall, with lower amplitude after a slight delay. A better illustration can be seen below.

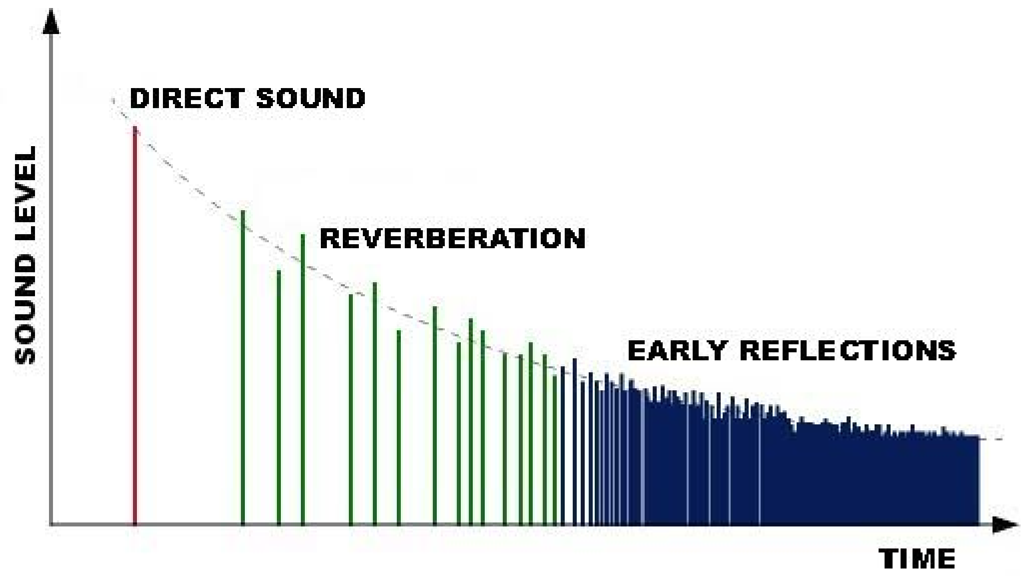

Figure 5: A better illustration of Room Impulse Response

Here, we can see that the direct sound is the original speech signal and follows by the echos of itself. For the adaptive filter, we used built-in Frequency Domain Adaptive filter (FDAF) from Matlab. This filter is great for identifying long impulse response, as well as having reduced filter length guarantees less computationally intensive and fast convergence. However, it has some trade-off such as increasing in latency, since it samples previous value.

Results and Analysis

While we saw great improvement in cancelling and suppressing echoed signal with our filter, we also saw some tradeoffs in the signals. Namely reduced in audio quality as well as distortion to the original signal after the filter. Using the original echoed signal and echo-removed signal, we can compare and contrast our filter's parameter by means of Echo Return Loss Enhancements or ERLE for short. It is a measurement of how much echo is attenuated in comparison to original signal.

Results

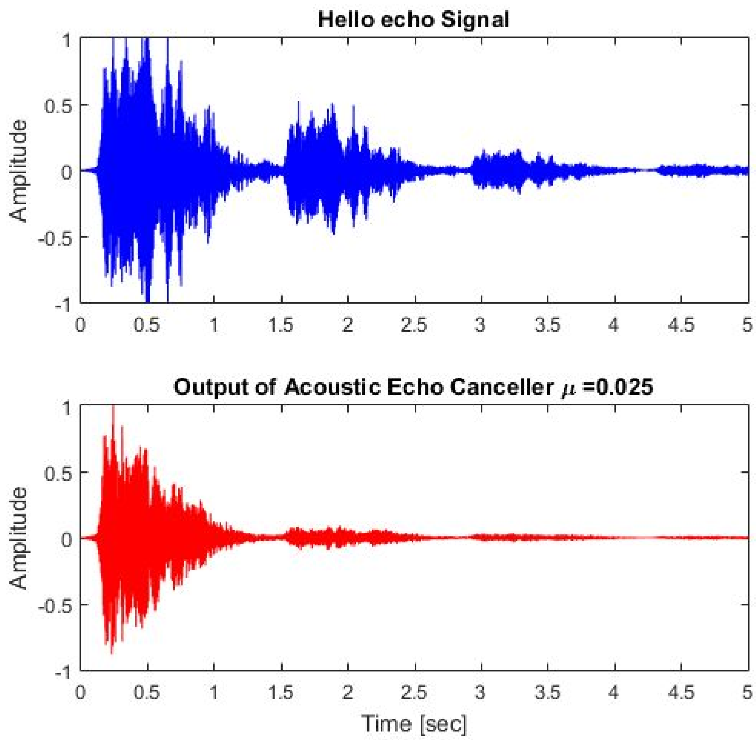

Figure 6: An illustration of original signal and after removing echo

As we can see, the echo presented in the original signal is almost canceled out by the filter, however, there is some residual echo left in the signal as we would be able to hear it when we play the signal on matlab. We also found that only a certain set values of μ works well for this type of filter.

As part of learning experience, we tried creating synthetic echoed signals by superimposing the signal with delayed version of itself. Tweaking around with different values of amplitude and delay, we noted that our filter works best if our delay is relatively low (~50 mS). This way we were able to get back the similar but somewhat distorted version of the original signal.

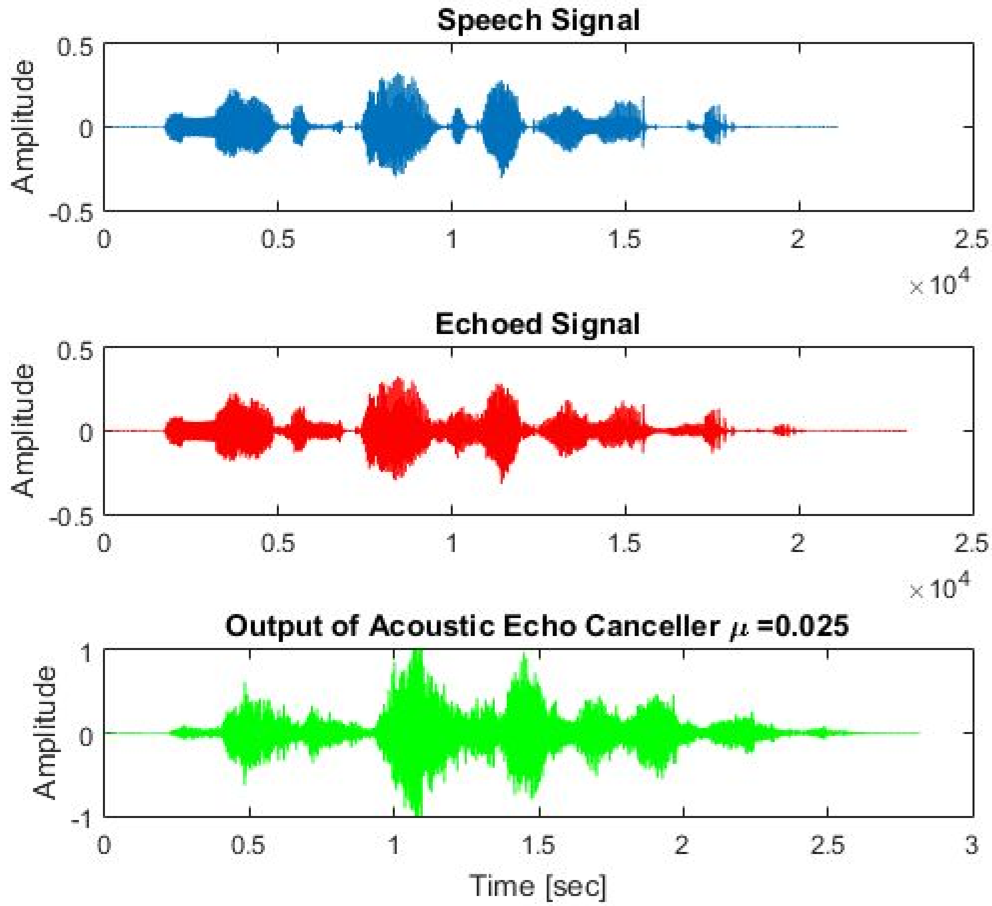

Figure : The synthetic echo creation and removal

It can be seen that the processed signal is distorted after the filter.

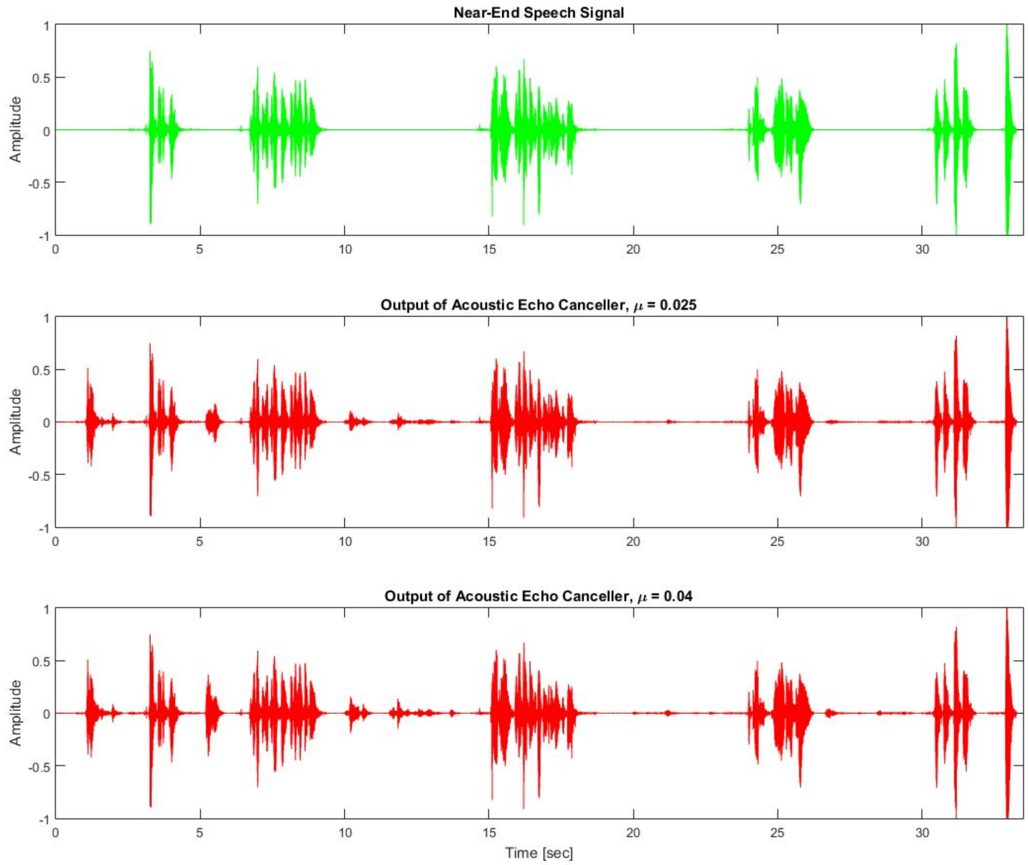

Figure : The output of Echo Canceller with different values of μ

This is a great example of how the echo is attenuated before and after the filter. We can see that the audio quality is equally the same as the original signal yet, the echo is removed from the signal. We believe this has to do with how Matlab generator generates room impulse response namely, its parameters such as delay and amplitude. It is quite obvious that certain characteristic of echoed signals may be similar to one another but how echos are presented in our tested speech signals won't be the same, hence difference in how our filter performs.

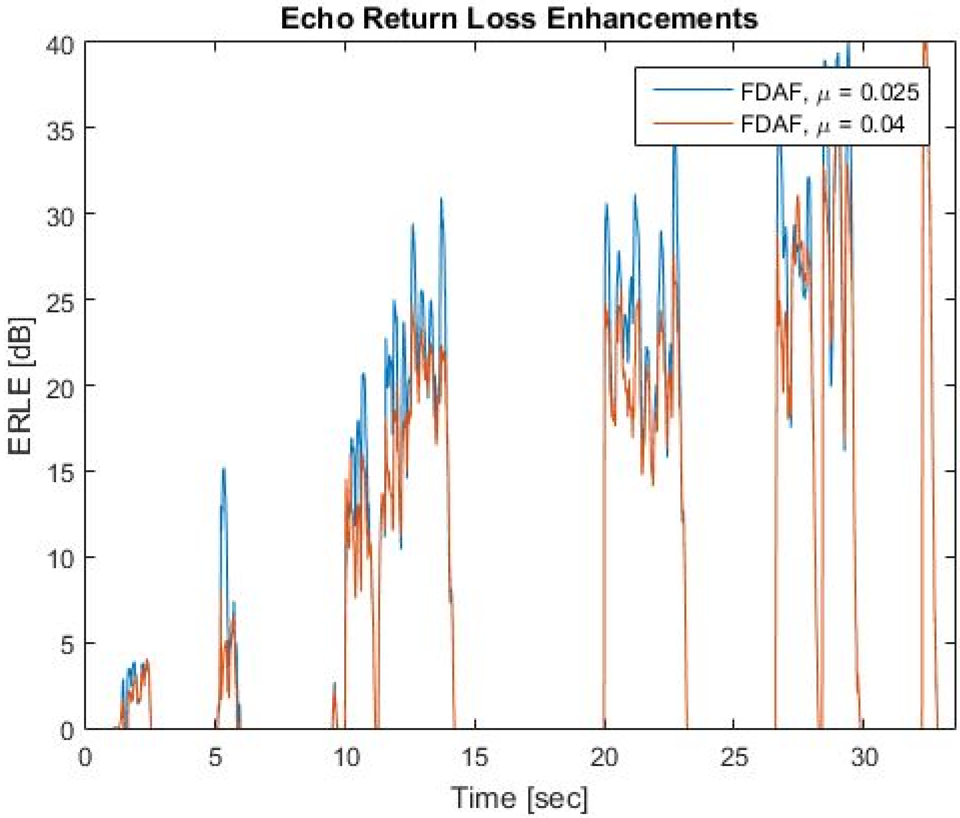

Figure : ERLE indicates how much the echo is attenuated

From the figure above, we can see that the echo is greatly attenuated and the speech signal sounds much better without any reverbing. With our implementation of filter, we can see that the best ERLE we can get is around 35 dB of echo attenuating.

Analysis

After iterating the filter with different values of μ, where μ is defined as the learning rate of the filter, we found that μ = 0.025 seems to be the best for our filter. This means that only a very specific value of learning rate works well for a given signal. This also has to do with how we convolute with the room impulse response: we simply assume that all our speech signals comes with same room with same room impulse response, which is not true. As an example, we used a long delayed echo speech signal, that says “Hello”, with decaying amplitude. This is a great example because we can clearly hear that the echos after the first direct speech decays rather quickly. However, there is a reduced in sound quality and therefore, it doesn’t sound quite the same after the filter-- which is one of the tradeoff of using FIR echo cancelling filter.

As stated previously, the quality of the audio signal is reduced once it is put through the filter. We might need to put this signal to another filter that would enhance the frequencies in the audio signals that correspond to speech, meaning in the range of 300-3400Hz. As for the signal in our second trial, we also need a filter to get rid of the white noise so we could improve the overall quality of the original audio signal.

Development

This project gives us an insight not only to implementation of filters using Matlab, but also how these filters work and the mathematical theory behind why they work. It's a rewarding learning experience, where we learned quite a bit of synthesizing our own echoed signals. As a further development, we would like to have our filters automatically adjust the learning rate in real-time, which would be a really neat application to real-world scenario.

Conclusions

We learned that built-in Matlab tools are really handy when it comes to Digital Signal Processing. It has great visualization tools as well as a wide variety of filters to choose from. We have learned a few important parameters on signal processing filters such as learning parameter or step size, μ. Larger step size doesn't guarantee the better performance of the echo cancellation and only a finite set value of μ works well for given filter. Using these information, we were able to get Echo Return Loss Enhancements (ERLE), the measurement of echo attenuation, i.e. filter's performance. Also the most effective filters can suppress the echos in speech signal up to 35 dB or a factor of up to 65 times which is quite significant. We also learned how to synthesize our own echoed signal by first creating a matrix with impulse functions and doing convolution. We also learned how to implement filters using fvtool in Matlab, as well as plotting and comparing before and after processing signals.

References

[1] Douglas, Scott C. "Acoustic Echo Cancellation(ACE)." MathWorks®. 2017.

https://www.mathworks.com/help/audio/examples/acoustic-echo-cancellation-aec.html#zmw57dd0e927

Introduction to MATLAB's basic equations for echo cancellation and adaptive filter models.

[2] "Echo Cancellation Using Adaptive Filters In MATlab Code"
http://vaaiibhav.me/echo-cancellation-using-lms-matlab-code/

We used this as an essential guide to implementing the LMS algorithm using MATLAB. It outlined the five steps that we mentioned in our presentation slides.

[3] Eriksson, Kalle F. Jonsson, Axel. Ahlander Mans. " Algorithms in Signal Processors EITN80 Adaptive Echo Canceller." March 2017.
http://www.eit.lth.se/fileadmin/eit/courses/etin80/2017/reports/adaptive-echo-canceller.pdf

This document provided us with the fundamental theoretical background to carry out this project.

[4]"Echo Sounds" May 2017.
http://soundbible.com/tags-echo.html

We used this website to get the "Hello echo" audio sample. We used this audio sample to test different learning parameters.

[4]Dowla, Farid. "Adaptive Filters, C/D, A/D:Review , Quantization Errors." [Lecture slides]. May 15, 2017.

Fundamental theoretical background to carry out this project.