THE UNIVERSITY OF TEXAS AT ARLINGTON

COMPUTER SCIENCE AND ENGINEERING

MATLAB Filter Design Project

SIGNAL PROCESSING

Submitted by,

Servando Olvera

1001909287

Date 12/01/2023

Description

In this project, you will be designing a filter to remove unwanted noise from an audio file. You will accomplish this with a Butterworth lowpass filter, which will be designed using MATLAB.

Audio Sample:

The project includes an audio sample that consists of a conversation in the presence of noise. The noise is thunder and a high-frequency tone.

The project should be completed in the following steps:

1. Audio Frequency Analysis

   Use the DFT, spectrogram, or other analysis technique to determine the wanted and unwanted frequency ranges for the audio sample. Keep in mind that unwanted frequencies may overlap with wanted frequencies.

2. Filter Design

   Use MATLAB filter design commands to design a digital filter that will reduce the noise from the audio sample. You will need to experiment to see what filter parameters result in the best final result. You can use a lowpass, highpass, or bandpass filter.

3. Filter Implementation

   Use MATLAB filtering commands to apply the filter designed above to the audio sample.

4. Plots

   At a minimum, create the following plots and include in your report:

   a. Any plots used for audio frequency analysis.

   b. Plots of the DFT of the audio before and after filtering. In these plots, the zero frequency should be in the center of the plot and the horizontal axis should show frequency in Hz.

   c. A plot of the magnitude of the frequency response of the filter you designed. The horizontal axis should display digital frequency in rad/s.

Problem

The problem this project aims to solve is the removal of unwanted noise from a given audio file. The audio contains both desired signals, a conversation. and unwanted noise elements, such as thunder and a high-frequency tone.

Goal: The goal of this project is to design a digital filter whose implementation effectively attenuates or removes unwanted noise while preserving the clarity of the conversation.

Key Challenges:

1. Identifying and Isolating Unwanted Frequencies:
   Differentiating between the frequencies associated with the conversation and those associated with the unwanted noise can be challenging, especially if they overlap in certain ranges.

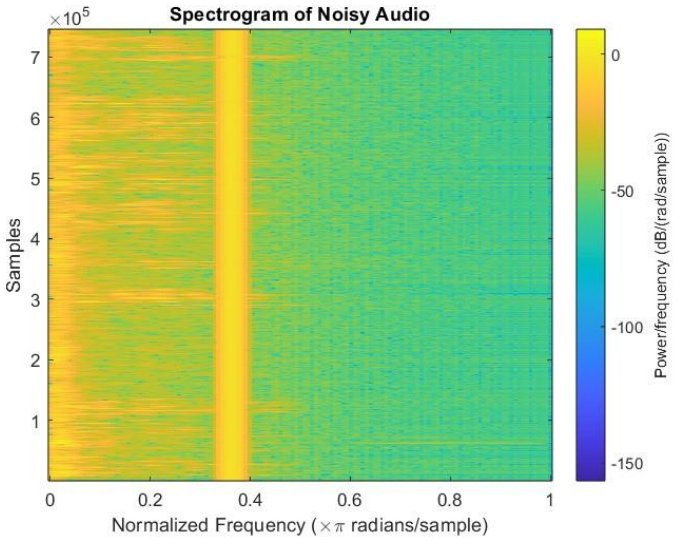
2. Balancing Noise Reduction and Signal Preservation:
   Designing the filter to remove the unwanted noise without significantly changing or the desired conversation can be difficult. Finding the right filter parameters (ω_P, ω_S, δ_P, δ_S) to obtain a balance between noise reduction and signal clarity can prove to be challenging.

3. Choosing the Filter Type and Order:
   Deciding whether to use a lowpass, highpass, or bandpass filter, as well as determining the appropriate parameters for it involves certain costs in the output. Higher filter orders might provide better noise reduction but might also distort the conversation.

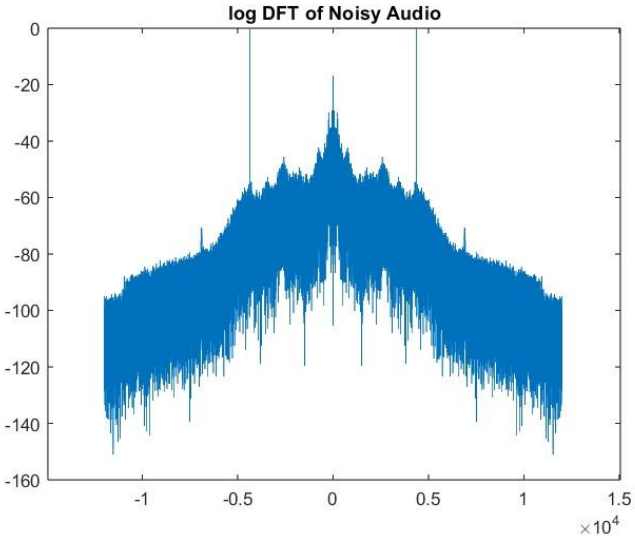
Approach

My approach to this project was based on the code already provided for us, and the lecture associated with this project. The “MATLAB Filter Demo” code and its respective lecture, where we walked through an example of a digital lowpass filter, were of great aid when developing this project. However, with the audio sample provided a lowpass filter would only remove the high-pitch noise and not the thunder noises, since the those are at lower frequencies. This could be observed on the figure “Spectogram of Noisy Audio”. In my approach I attempted to implement a passband digital filter, this way I could simultaneously target both unwanted low and high frequencies, but this proved to be inadequate since it distorted the conversation significantly, so I decided to stick with the lowpass filter. Then, as how it was done in the example, I used the figures “Spectogram of Noisy Audio” and “log DFT of Noisy Audio” to determine the *Butterworth filter order and cutoff frequency* function parameters (ω_P, ω_S, δ_P, δ_S). From there, the obtained filter order and cutoff frequency variables were directly fed into the *Butterworth filter* function and the rest of the code worked its logic displaying figures showing the before and after filtering of the audio sample.

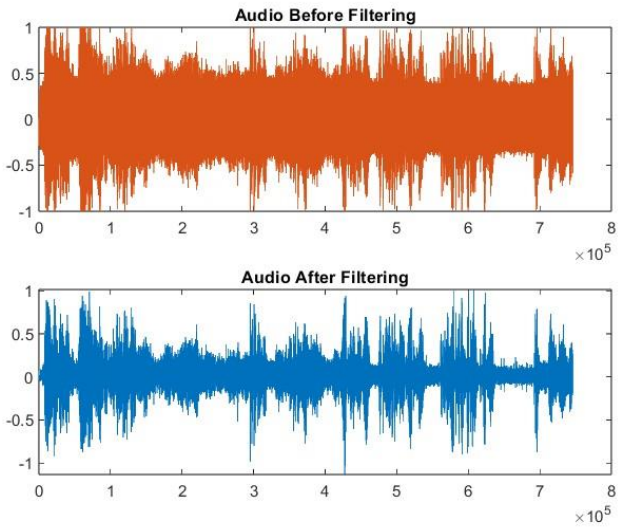
Results



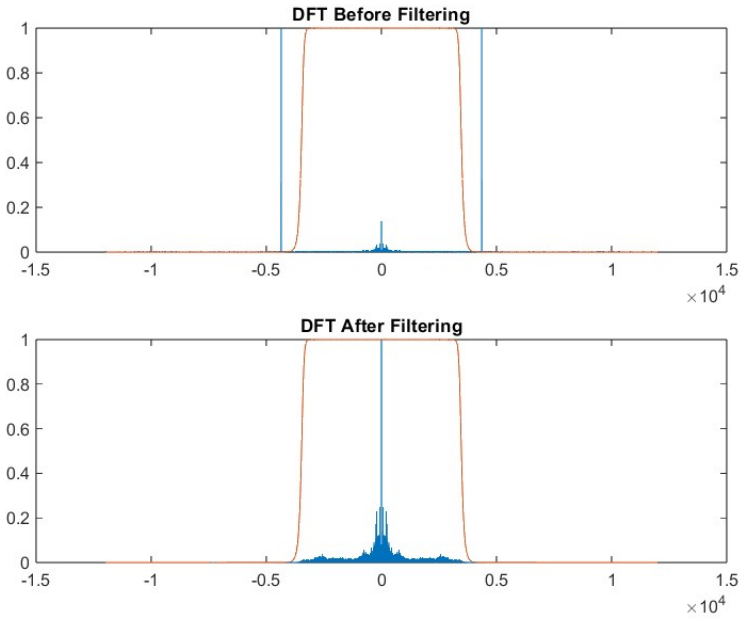
In the figure above it is clear that past 0.5 there is nothing to be acknowledged, no useful data on that region. Furthermore, is it safe to assume that the solid yellow line is likely to be the high-pitch noise, thus $\omega_S < 3.4$, and since it is safe to assume that the yellow region on the left is the conversation to be filtered, that would entail that $\omega_P > 0.25$.



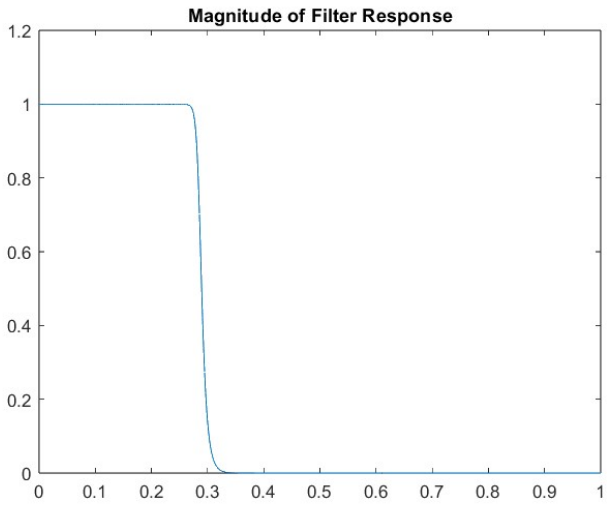
Looking at the DFT of the sample audio, the high-pitch noise appears to have a massive impact on it. However, messing around with the code to find a relatively functional value for δ_S it appears that anything above an attenuation of 55 dB would distort the audio quite significantly, so $\delta_S < 55$. Furthermore, the values used for δ_P was the standard 1 dB ripple.



The figure above shows the audio before and after filtering. It can be observed that after the audio was filtered its amplitude was not as pronounced as before, entailing that indeed some noise was removed from the audio.



The figure above shows the audio's DFT before and after filtering. It can be observed that after filtering its frequency components/peaks were more pronounced or more noticeable, entailing that the conversation frequencies were 'enhanced'.



From the figure above, it is evident that the filter responded accordingly to the chosen values for ω_P and ω_S. Past the respective passband, at around 0.28, the filter begins to attenuate the undesired frequencies, showing that the filter's cutoff frequency was set accordingly.