**Digital Signal Processing**

**BS(CE)-2k20**

**Semester Project Report**



**Submitted By:**

Abdullah Javed (20-CE-035)

Ayesha Ehtisham Ullah (20-CE-038)

**Submitted To:**

Engr. Ali Raza

Created on:

11th June, 2023

**Department of Computer Engineering**

**HITEC University Taxila**

Contents

[**Compression and Filtration of Audio Signals** 4](#_Toc137376740)

[Abstract: 4](#_Toc137376741)

[Introduction: 4](#_Toc137376742)

[Background: 4](#_Toc137376743)

[Objectives: 4](#_Toc137376744)

[Significance: 5](#_Toc137376745)

[Literature Review: 5](#_Toc137376746)

[Fourier Transform: 5](#_Toc137376747)

[Inverse Fourier Transform: 6](#_Toc137376748)

[Chebyshev Filter: 6](#_Toc137376749)

[Methodology: 7](#_Toc137376750)

[Tools Used: 7](#_Toc137376751)

[Major Design Components: 7](#_Toc137376752)

[Source Code: 7](#_Toc137376753)

[Results: 11](#_Toc137376754)

[Output After Compression: 11](#_Toc137376755)

[Output After Noise Cancellation: 12](#_Toc137376756)

[Conclusion: 12](#_Toc137376757)

[Benefits: 12](#_Toc137376758)

[Applications: 12](#_Toc137376759)

Table of Figures

[**Figure 1 Audio Noise FIlter CIrcuit Diagram** 4](#_Toc137376760)

[Figure 2 Fourier Transform 5](#_Toc137376761)

[Figure 3 Inverse Fourier Transform 6](#_Toc137376762)

[Figure 4 Chebyshev Filter 6](#_Toc137376763)

[Figure 5 Audio Compression Function 7](#_Toc137376764)

[Figure 6 Audio Filtration Function 8](#_Toc137376765)

[Figure 7 Plotting Function 9](#_Toc137376766)

[Figure 8 Compressed Waveform 11](#_Toc137376767)

[Figure 9 Compressed Audio File 11](#_Toc137376768)

[Figure 10 Filtered Audio Waveform 12](#_Toc137376769)

[Figure 11 Filtered Audio File 12](#_Toc137376770)

# **Compression and Filtration of Audio Signals**

## Abstract:

In out project of **Digital Signal Processing**, we have designed a software application which performs 2 functions:

* **Audio Compression**
* **Noise Cancellation**

This project is based on the concepts of **Digital Signal processing**. This project uses **IIR Filter** as its main component. The **compression** and **noise filtration** component has been implemented on **Python Programming Language.** It takes the **input signal** and it performs **compression or filtration**, **displays** both the **waveforms** and gives the **output audio**.

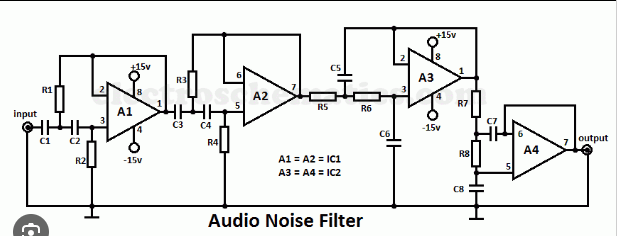
## Introduction:

### Background:

In today’s telecommunication and audio devices, we want to transfer data at high speeds and at the highest possible quality. This can lead to noise and peaking of the audio. Which can make it hard to understand the audio signal. It can also use more data. So, to prevent all these issues, we use filters which can compress the audio so that is takes less space, and is free of any unwanted signal/noise.

### Objectives:

The main objective of the proposed project is to design such a filter which can compress and audio signal or remove noise. Both these features can be used at user’s will.



**Figure 1 Audio Noise FIlter CIrcuit Diagram**

### Significance:

The audio filters are electronic circuits designed to amplify or attenuate a certain range of frequency components. This helps eliminate the unwanted noise from the audio signal and improves the tone of the output audio. Filters play a major role in telecommunication and audio electronics.

When your voice becomes too loud it will cause your microphone to peak which means your audio will become distorted. To prevent your audio from peaking, you’ll want to use the compressor audio filter that when you start to yell, your audio will be lowered preventing your audio from peaking.

## Literature Review:

We have studied and implemented the following concepts for the development of our project:

### Fourier Transform:

In [physics](https://en.wikipedia.org/wiki/Physics) and [mathematics](https://en.wikipedia.org/wiki/Mathematics), the **Fourier transform** (**FT**) is a [transform](https://en.wikipedia.org/wiki/Integral_transform) that converts a [function](https://en.wikipedia.org/wiki/Function_(mathematics)) into a form that describes the frequencies present in the original function. The output of the transform is a [complex](https://en.wikipedia.org/wiki/Complex_number)-valued function of frequency. The term *Fourier transform* refers to both this complex-valued function and the [mathematical operation](https://en.wikipedia.org/wiki/Operation_(mathematics)). When a distinction needs to be made the Fourier transform is sometimes called the [frequency domain](https://en.wikipedia.org/wiki/Frequency_domain) representation of the original function. The Fourier transform is analogous to decomposing the [sound](https://en.wikipedia.org/wiki/Sound) of a musical [chord](https://en.wikipedia.org/wiki/Chord_(music)) into terms of the [intensity](https://en.wikipedia.org/wiki/Sound_intensity) of its constituent [pitches](https://en.wikipedia.org/wiki/Pitch_(music)).

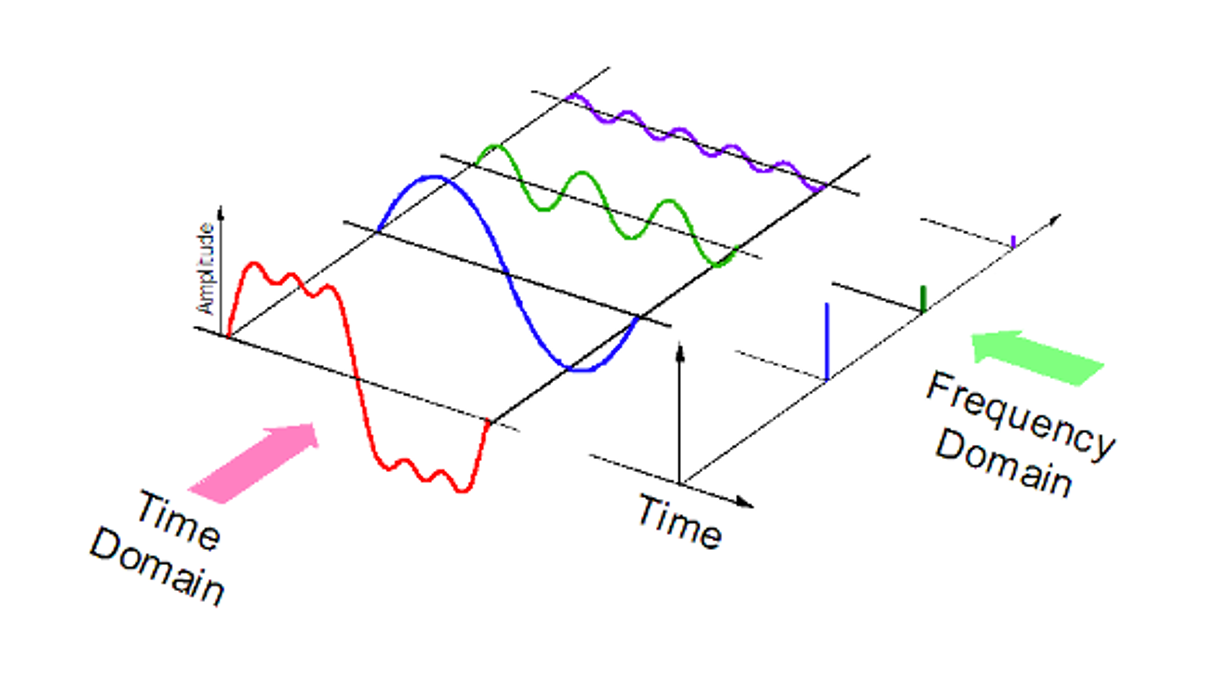


Figure 2 Fourier Transform

### Inverse Fourier Transform:

In [mathematics](https://en.wikipedia.org/wiki/Mathematics), the **Fourier inversion theorem** says that for many types of functions it is possible to recover a function from its [Fourier transform](https://en.wikipedia.org/wiki/Fourier_transform). Intuitively it may be viewed as the statement that if we know all [frequency](https://en.wikipedia.org/wiki/Frequency#Frequency_of_waves) and [phase](https://en.wikipedia.org/wiki/Phase_(waves)) information about a wave then we may reconstruct the original wave precisely.

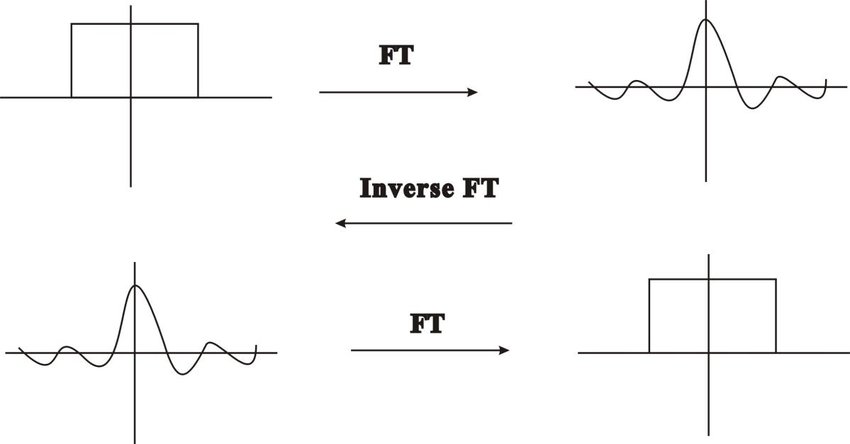


Figure 3 Inverse Fourier Transform

### Chebyshev Filter:

**Chebyshev filters** are [analog](https://en.wikipedia.org/wiki/Analog_filter) or [digital](https://en.wikipedia.org/wiki/Digital_filter) filters that have a steeper [roll-off](https://en.wikipedia.org/wiki/Roll-off) than [Butterworth filters](https://en.wikipedia.org/wiki/Butterworth_filter), and have either [passband](https://en.wikipedia.org/wiki/Passband) [ripple](https://en.wikipedia.org/wiki/Ripple_(filters)) (type I) or [stopband](https://en.wikipedia.org/wiki/Stopband) ripple (type II). Chebyshev filters have the property that they minimize the error between the idealized and the actual filter characteristic over the operating frequency range of the filter,[[1]](https://en.wikipedia.org/wiki/Chebyshev_filter#cite_note-Daniels1974-1)[[2]](https://en.wikipedia.org/wiki/Chebyshev_filter#cite_note-Lutovac2001-2) but they achieve this with ripples in the passband. This type of filter is named after [Pafnuty Chebyshev](https://en.wikipedia.org/wiki/Pafnuty_Chebyshev" \o "Pafnuty Chebyshev) because its mathematical characteristics are derived from [Chebyshev polynomials](https://en.wikipedia.org/wiki/Chebyshev_polynomials). Type I Chebyshev filters are usually referred to as "Chebyshev filters", while type II filters are usually called "inverse Chebyshev filters".Because of the passband ripple inherent in Chebyshev filters, filters with a smoother response in the passband but a more irregular response in the stopband are preferred for certain applications.

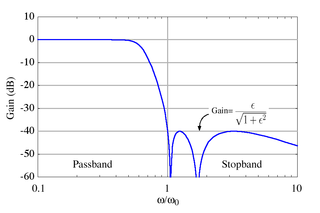


Figure 4 Chebyshev Filter

## Methodology:

### Tools Used:

The following tools have been used for the development of this project:

* Python Programming Language
* Pycharm Community IDE
* Librarie such as **numpy, scipy, soundfile, matplotlib.**

### Major Design Components:

Following are the major design areas of this project:

* Fourier Transform
* Inverse Fourier Transform
* Shifting
* Scaling
* Chebyshev Filter

### Source Code:

#### Audio Compression:

Following is the algorithm of this function:

* Take the input signal
* Compute its Fourier Transform
* Compress it by the **Compression Factor** given by the user
* Compute its Inverse Fourier Transform
* Return the Compressed Signal

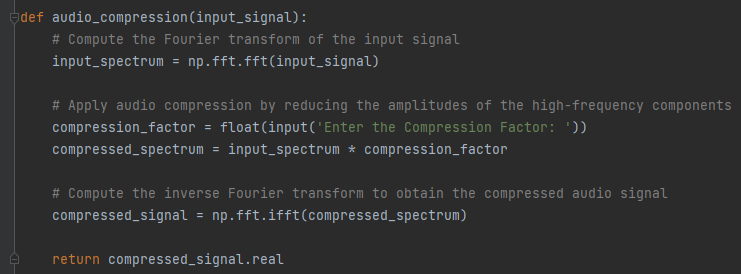


Figure 5 Audio Compression Function

#### Noise Cancellation:

Following is the algorithm of this function:

* Take the input signal
* Compute its Fourier Transform
* Input the Filter Order
* Input the Cut-off Frequency
* Use Chebyshev function to filter the audio
* Compute the Inverse Fourier Transform of the filtered signal
* Return the output signal.

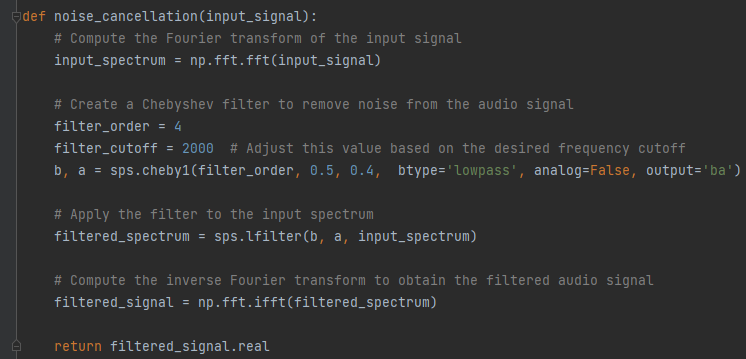


Figure 6 Audio Filtration Function

#### Plotting:

The algorithm of the plotting function is as follows:

* Calculate the time
* Plot the Input signal with its time and amplitude
* Plot the Output signal with its time and amplitude.
* Use Subplot for plotting both the waveforms to the same window.

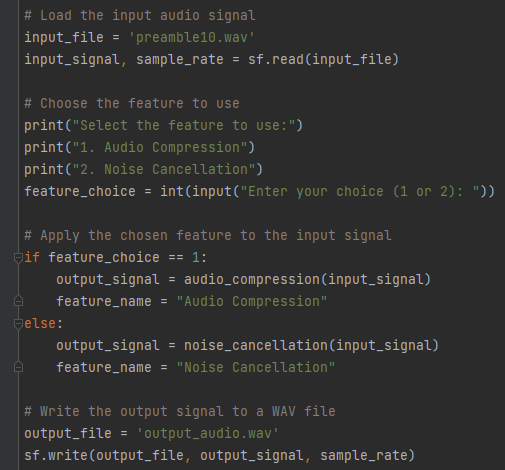


Figure 7 Plotting Function

#### General Working:

Following is the algorithm for the general working of the code:

* Load the input audio file in .wav format
* Take user input for which function to use
* Function call on the basis of user’s choice
* Store the output file obtained from the functions in the computer’s storage.



## Results:

### Output After Compression:

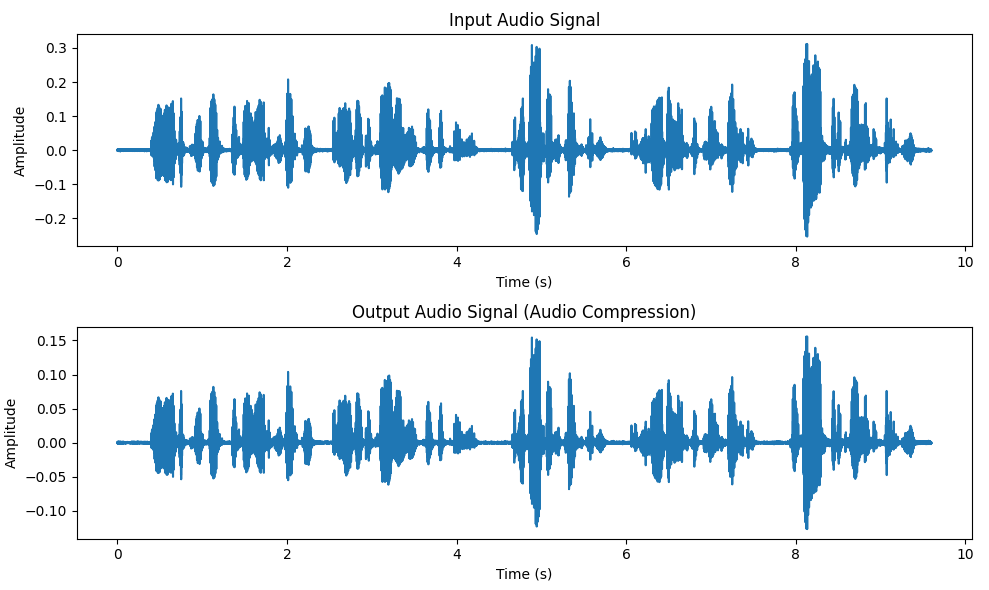


Figure 8 Compressed Waveform

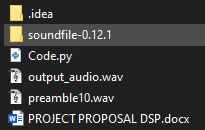


Figure 9 Compressed Audio File

### Output After Noise Cancellation:

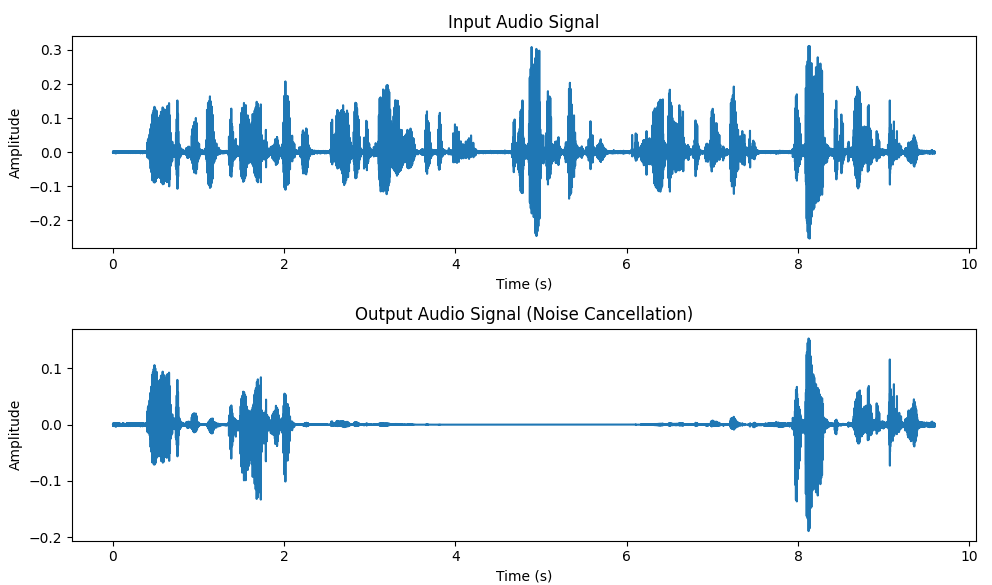


Figure 10 Filtered Audio Waveform

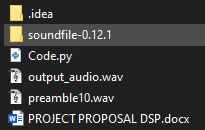


Figure 11 Filtered Audio File

## Conclusion:

### Benefits:

* You can automatically remove unwanted material from the audio
* You don’t need any other software for this purpose
* Saves time & memory

### Applications:

* Music Studio
* Audio Players
* Mobile Communication
* Amplifiers

Hence, the project is based on how we can easily implement any filter and compress the audio or intercept the audio