EE310A Course Project
Real-time Audio Filtering Platform

Shivangi Singh
Roll no.: 180726

July 2021

1 Introduction

The aim of this project was to create a real-time audio filtering platform, which takes audio input through the mic, filters it instantly, and then outputs the audio through the speaker. We were supposed to give the user an option between the different ways in which they could specify what their filter looks like, namely: H($e^{j\omega}$), h[n] and LCCDE equation coefficients.

2 Description:

This section holds the description of the steps that were followed for developing each part of the platform.

2.1 Taking audio input:

We take audio input using PyAudio plugin. We have written two different functions, in terms of audio input: one gives us real-time output, and the other saves the audio in a .wav file, so that we can do offline filtering on the input.

2.2 Constructing the filter:

2.2.1 Using LCCDE coefficients:

LCCDE is y[n], written in terms of the past values of y[n] and x[n]. The expression looks like this:

$$a_0 y[n] = -\sum_{k=1}^{K} a_k y[n-k] + \sum_{m=0}^{M} b_m x[n-m]$$

The user is asked to provide a_k, K, b_m and M as inputs. If we take the Z-transform of the above equation and simplify, we get:

$$\frac{Y(z)}{X(z)} = \frac{\sum_{m=0}^{M} b_m z^{-m}}{\sum_{k=0}^{K} a_k z{-k}} = H(z)$$

2.2.2 Using poles and zeros of H(z):

The user is asked to provide the number of zeros, their values, number of poles and their values as inputs. From this, we get

$$H(z) = \frac{\Pi_{i=1}^{N}(z-\phi_i)}{\Pi_{j=1}^{M}(z-\lambda_j)}$$

2.2.3 Using h[n]:

The user is asked to provide the length of $h[n]$ and the value of each impulse as inputs. We use the fundamental formula of Z-transform to calculate $H(z)$ over here:

$$H(z) = \sum_{i=1}^{N} h[i] z^{-i}$$

In this way, we calculate $H(z)$ and return the expression.

2.3 Filtering:

2.3.1 Obtaining DTFT from Z-Transform:

Since the Z-transform is not necessarily finite, we will need the expression for the Discrete-time Fourier Transform of the LTI system. We know that:

$$H(e^{j\omega}) = H(z)|_{z=e^{jw}}$$

2.3.2 Obtaining DFT from DTFT:

Just like the Z-Transform, the system's DTFT is also not necessarily finite. In the real world, we can only deal with causal systems, so we will have to sample the system's DTFT to obtain the Discrete Fourier Transform:

$$H[k] = H(e^{j\omega})|_{\omega=2\pi k/N}$$

In this way, we will create an array of N samples, which will be equivalent to the DFT of the LTI system.

2.3.3 Obtaining h[n] and y[n]:

To obtain h[n], we take the inverse DFT of the DFT H[k] we had obtained earlier. After this our work becomes pretty simple. Our filtered output y[n] will be the convolution of the input x[n] and h[n].

$$y[n] = x[n] * h[n]$$

3 Future Scope:

A future scope in this platform is to introduce zero-phase transfer functions, since we already have the function to save the input audio.