Audio Filtering using Butterworth Filter

Khang Nguyen

University of Texas at Arlington

Abstract

This report is to present our audio filtering technique using the Butterworth filter to denoise noisy speech audio. The ultimate goal of removing unwanted noise from the noisy audio input is to obtain a clear and audible output in order to enhance users' experiences. In this report, we use the Butterworth filter since it is a well-known signal processing filter and designed to have a system frequency response to be flat as much as possible within the passband region. Specifically, the desired (or actual) speech is limited by the ending point of the passband frequency, and the unwanted noise is limited by the starting point of the stopband frequency. The filter design technique will be explained in this report, and the implementation of this entire process is conducted using MATLAB built-in functions.

1 Introduction

Humans can detect sound in the range of from 20 Hz to 20 kHz of frequency [6]. Nevertheless, many sounds co-exist in the human hearing range, such as overlapping speech from different people being heard or environmental noises from many factors. By nature, species like us, humans, can be able to filter unwanted noises and focus on the primary input automatically; this phenomenon is known as the *cocktail party effect* [1, 5], in which the partygoer's brain concentrates on one's speech while filtering out a range of other stimuli. Hence, he or she can focus on a single conversation in a noisy room. Unfortunately, microphones on embedded systems or robots do not have the ability to differentiate input signals mixed with environmental noises as humans do [4]. Therefore, filtering such unwanted noises can be helpful for computers to provide us with a better sound quality even when the noisy signal input is recorded [3, 2]. In order to filter out the unwanted noises programmatically, we use the Discrete Fourier transform and the Butterworth filter with appropriate parameters to design a filter as desired (Sec. 4).

2 Audio Problem & Challenges

2.1 Problem

Given a segment of a noisy audio file (speech with noises), filter the unwanted noise by designing a filter with appropriate parameters to obtain clean, audible speech.

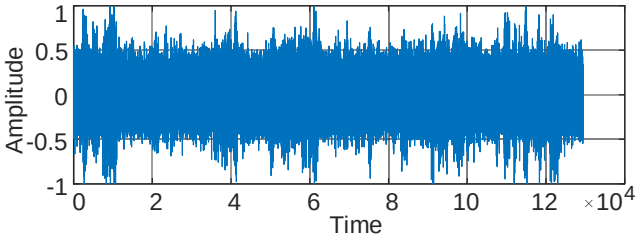

Figure 1: Noisy Audio Waveform

Reading from the given .wav file and reconstructing the audio signal in the form of waveform, we obtain Fig. 1. As we can see from Fig. 1, the noises and the speech are fused simultaneously with the duration of the file.

2.2 Challenges

As the problem is stated, we start to design such a filter to remove the unwanted noises from the audio. However, implementing the filter for the audio file is difficult due to the following challenges:

- Choosing the endpoint of the passband, ω_p, and the beginning point of the stopband, ω_s, needs to be precise. Otherwise, the filtered audio would still contain some audible-recognizable noises, or the speech would be lost.
- Identifying the appropriate minimum attenuation for the noise to attenuate sufficiently is challenging due to the complexity of the precise procedure of audio frequency analysis.
- Calculating the cut-off frequency, Ω_c, and the order, N, for the Butterworth filter is complicated since the precision in establishing mathematical equations is needed.

3 Audio Frequency Analysis

In this section, we analyze the audio input frequencies using the Discrete Fourier Transform (DFT) and the magnitude in decibels (dB).

3.1 Analyzing Frequency using DFT

First of all, to analyze the frequencies of the input audio, we apply the Discrete Fourier Transform to the input audio using the N-point DFT formula as follows:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi}{N}nk} \tag{1}$$

where $X[k]$ is the encode of the amplitude and phase of a sinusoidal wave with frequency $\frac{k}{N}$ cycles per time unit and $x[n]$ are values of input signal at n^{th} time.

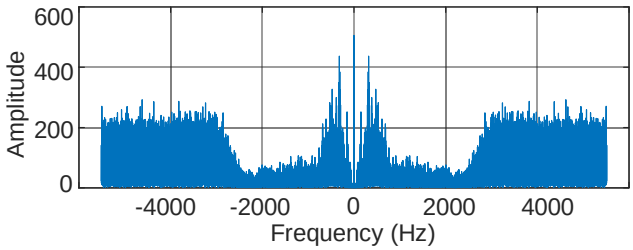

Figure 2: DFT of Noisy Audio

As can be seen in Fig. 2 after applying Eq. 1, the low-frequency audio (the speech frequency) is in the middle of the figure; meanwhile, the significant high-frequency audio (the noise frequency) are on the two sides of the figure. Based on this frequency, we aim to cut the noise out of the audio with a suitable frequency while having minimal impact on the speech frequencies.

3.2 Analyzing Magnitude in Decibels

We normalize Fig. 2 into a decibels scale with the maximum value of the DFT being 0 dB. The reason for doing this is that we can hence easily read lower peaks based on the baseline at 0 dB. To convert the DFT plot into the normalized log plot, we first convert the magnitude of the original DFT to dB scale as follows:

$$X_{db} = 20 \cdot log_{10}(X) \tag{2}$$

where X is an array of absolute values in the DFT, and X_db is an array of corresponding values of the DFT in dB scale.

Yet, for the X_db array, we deduct the magnitude of the maximum value in the X_db array from every element in the array by using the following equation:

$$X_{db} - max(|X_{db}|) \tag{3}$$

Applying Eq. 2 and Eq. 3 to the DFT, we obtain:

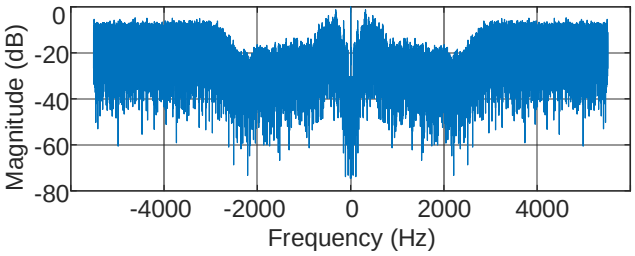

Figure 3: Normalized Log of Noisy Audio

4 Filter Design Technique

In this section, we find the most suitable parameters, including the endpoint of the passband frequency (ω_p), the start point of the stopband frequency (ω_s), the cut-off frequency (Ω_c), and the order (N) for the Butterworth filter, and implement the Butterworth filter.

4.1 Identifying Passband and Stopband Frequencies

From Fig. 2, we can see that the speech (low) frequencies change quickly to the high (noise) frequencies, which is considered the transition band, from 1990 Hz to 3375 Hz. Hence, we choose the endpoint of passband and start point of stopband frequencies as follows:

$$\begin{cases} \omega_p = 1990 \text{ (Hz)} \\ \omega_s = 3375 \text{ (Hz)} \end{cases}.$$

Therefore, the passband frequency is from 0 Hz to 1990 Hz, the transition band frequency is from 1990 Hz to 3375 Hz, and the stopband frequency is from 3375 Hz to infinity, as illustrated in Fig. 4:

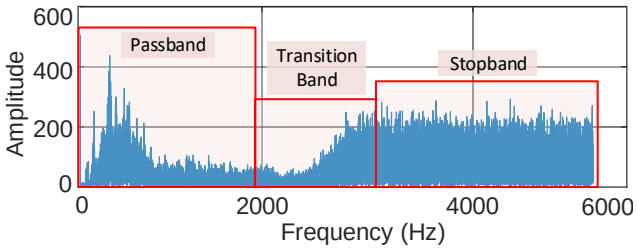

Figure 4: Passband, Transition Band, and Stopband of Noisy Audio

Note that if we choose ω_p and ω_s too close to each other, the order for the Butterworth filter will be extremely large, and this would cause computational complexity.

4.2 Identifying Stopband Attenuation

From Fig. 3, we can see that the attenuation for the noise to attenuate sufficiently in the stopband region is relatively large. In this particular case, we choose the minimum attenuation to be 69 dB, as shown in Fig. 5.

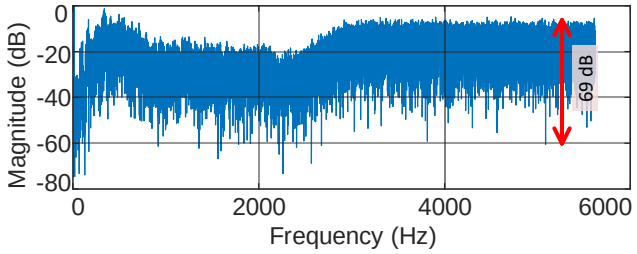

Figure 5: Attenuation in Stopband

We also assume no more than 1dB attenuation in the passband. Hence, we have:

$$\begin{cases} 1 - \delta_p \geq -1 \text{ (dB)} \\ \delta_s = -69 \text{ (dB)} \end{cases} . \tag{4}$$

4.3 Finding Cutoff Frequency & Order

In order to find the cutoff frequency and the order for the Butterworth filter, we do some mathematical operations using impulse invariance.

From Eq. 4, we rewrite these specifications in terms of requirements on the transfer function or frequency response, the magnitude of the frequency response as follows:

$$\begin{cases} 20 \cdot log_{10}|H(e^{j1990})| \geq -1 \\ 20 \cdot log_{10}|H(e^{j3375})| \leq -69 \end{cases} .$$

Rewrite the system of equation in terms of the system frequency response only, we get:

$$\begin{cases} |H(e^{j1990})| \geq 10^{-0.05} \\ |H(e^{j3375})| \leq 10^{-3.45} \end{cases} .$$

Also, we have the system of passband edge frequency equation and stopband edge frequency equation:

$$\begin{cases} \frac{1}{1+\left(\frac{\omega_p/T}{\Omega_c}\right)^{2N}} = (1-\delta_p)^2 \\ \frac{1}{1+\left(\frac{\omega_s/T}{\Omega_c}\right)^{2N}} = \delta_s^2 \end{cases} . \tag{5}$$

where ω_p is the passband cutoff frequency, ω_s is the stopband cutoff frequency, T is a design parameter, δ_p is the passband attenuation, δ_s is the stopband attenuation, Ω_c is the cutoff frequency, and N is the order for the Butterworth filter.

Algebraic manipulating on Eq. 5, we obtain:

$$\begin{cases} 1 + \left(\frac{\omega_p/T}{\Omega_c}\right)^{2N} = \frac{1}{(1-\delta_p)^2} \\ 1 + \left(\frac{\omega_s/T}{\Omega_c}\right)^{2N} = \frac{1}{\delta_s^2} \end{cases} . \tag{6}$$

Plugging in the values we had into Eq, 6, we get:

$$\begin{cases} 1 + \left(\frac{1990/T}{\Omega_c}\right)^{2N} = 10^{0.1} \\ 1 + \left(\frac{3375/T}{\Omega_c}\right)^{2N} = 10^{6.9} \end{cases} .$$

We have two equations and two unknown variables; hence, it is sufficient to find Ω_c and N; thus, we have:

$$\begin{cases} \Omega_c = 1927.5334 \text{ (Hz)} \\ N = 16.3168 \approx 17 \end{cases} .$$

4.4 Butterworth Filter Response

After obtaining the cutoff frequency (Ω_c) and the order (N), we plot the logarithmic gain of the frequency response of the analog filter using the following equation:

$$|H_a(s)| = 20 \cdot log_{10}\left(\sqrt{\frac{1}{1+\left(\frac{j\Omega}{j\Omega_c}\right)^{2N}}}\right) \tag{7}$$

Using Eq. 7 to plot the gain plot, we obtain the digital Butterworth filter response plot over time domain:

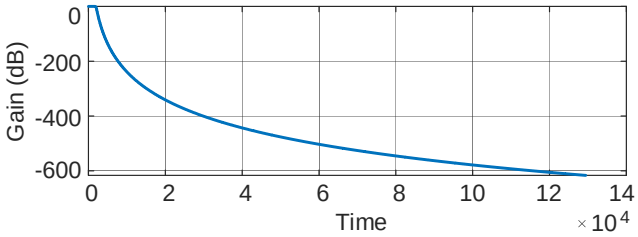

Figure 6: Digital Butterworth Filter Response

As we can see from Fig. 6, there is a flat region of the system frequency response at the beginning of the digital Butterworth filter response.

4.5 Implementing Butterworth Filter

As the cutoff frequency (Ω_c) and the order (N) for the Butterworth filter are found, we use `butter(N, Wn)` command, where W_n is specified as a percentage of $F_s/2$ and can be expressed in terms of cutoff frequency and sampling frequency (F_s):

$$W_n = \frac{\Omega_c}{F_s/2}$$

Thus, $W_n = 0.3497$ as $\Omega_c = 1927.5334$ and $F_s = 11025$

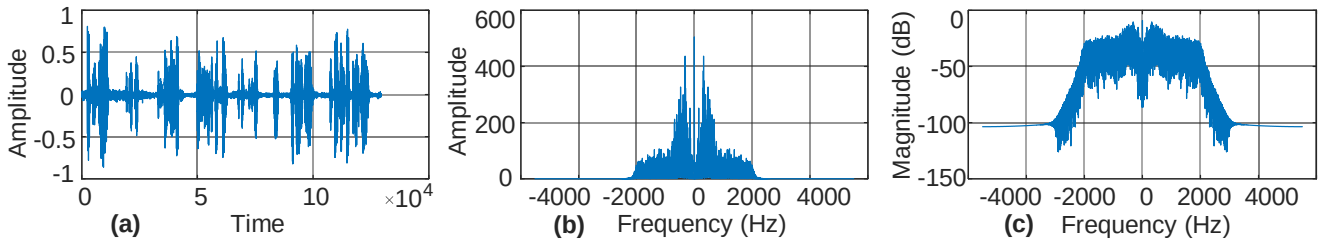

Figure 7: (a) Filtered Audio Waveform, (b) DFT of Filtered Audio, (c) Normalized of Filtered Audio

5 Results

After applying the Butterworth filter for the noisy audio, we obtained the filtered audio file. The filtered audio file can be analyzed in the same way we examined the given noisy audio file, such as the waveform, the DFT, and the normalized log of the denoised audio. In comparison to Fig. 1, Fig. 7 (a) is filtered out the audible noises significantly, and the remaining waveform is believed to be the speech waveform. For Fig. 7 (b) and (c), it is obvious that the high (noise) frequencies that we wanted to cut off in Fig. 2 and Fig. 3 was filtered out in the filtered audio, and the remaining parts are the low (speech) frequencies that we would like to keep.

The filtered audio can be heard to verify how well the filter is designed. Since it is audible, we can hear the conversation between two men. The conversation can be transcripted as:

- How soon you can land?
- I can't tell.
- You can tell me. I'm a doctor.
- No, I mean I'm just not sure.
- Well, can't you take a guess?
- Well, not for another two hours.
- You can't take a guess for another two hours?

This is the iconic scene of the conversation between `Rumack` and `Captain Oveur` in the `Airplane!` movie in 1980.

6 Conclusion

In conclusion, the Butterworth filter works well on a noisy audio input when it is well-designed. However, some manual frequency analyses require a more insightful perspective on the given data. In this case, we first have to understand how to differentiate the dissimilarities between the speech frequencies and the noise frequencies based on both the audio file and the Discrete Fourier Transform procedure. With that being done, we can define the passband and stopband frequencies and attenuations properly and calculate the Butterworth filter's required parameters. In case of the filtered audio is altered inadequately, we have to re-choose the more appropriate parameters in order to get better audio quality. Nevertheless, thanks to MATLAB's built-in functions, we do not have to go through the lengthy calculations again in later iterations.

References

[1] Barry Arons. A review of the cocktail party effect. *Journal of the American Voice I/O Society*, 12(7):35–50, 1992.

[2] Jacob Benesty, Jingdong Chen, and Yiteng Huang. *Microphone array signal processing*, volume 1. Springer Science & Business Media, 2008.

[3] Jie Huang, Noboru Ohnishi, and Noboru Sugie. Building ears for robots: sound localization and separation. *Artificial Life and Robotics*, 1(4):157–163, 1997.

[4] David McFarland, Tom Bösser, and Tom Bosser. *Intelligent behavior in animals and robots*. Mit Press, 1993.

[5] Irwin Pollack and James M Pickett. Cocktail party effect. *The Journal of the Acoustical Society of America*, 29(11):1262–1262, 1957.

[6] Dale Purves, George J Augustine, David Fitzpatrick, Lawrence C Katz, Anthony-Samuel LaMantia, James O McNamara, and S Mark Williams. Circuits within the basal ganglia system. In *Neuroscience. 2nd edition*. Sinauer Associates, 2001.