

Audio signal analysis with filter by using FIR filter and detection of signal in noise

EECS451 Project Report

Jianhang Qiu
Department of EECS
qiujiangh@umich.edu

Abstract—This project is mainly about using different windows in Matlab to design FIR filter for signal with different kinds of noise. Compare the results of the signal having noise through the filter with the former result to see whether the designed filter works well. Then the DSP Tool outside class which is detection will be discussed. This part is about how to determine who the speaker according to the voice of database is.

Keywords—FIR Filter, window, Matlab, detection.

I. INTRODUCTION

This project is to use different windows in Matlab to design FIR filter to filter audio signal with noise. The design principle and procedure of FIR filter could be learned. By observing the difference of waveform in frequency domain of the signal before and after the filter, we could find that filter has a lot of importance to signal. The second of part of the project is to design a system about the detection to identify who the speaker is depends on the database.

II. DESIGN PRINCIPLE

A. Get the audio Signal

Download a part of audio signal from the Internet. I choose the song “Hotel California” and incise it only to use the first 20s part. The whole song is too long to analyze. The format of the music is wav. The sampling frequency of the signal is 44100 Hz. By using the ‘wavread’ function, we could get the waveform in time and frequency domain which shows in Fig. 1 and fig. 2. ‘Wavread’ function is ‘[x,fs,bits]=wavread(‘Hotel California.wav’)’. ‘fs’ stands for sampling frequency which is 44100 Hz. ‘bits’ stands for the name of bits per sample which is 16. The waveform of the time domain is a little like a periodic signal which repeats every three second. The reason is that the prelude of this song has nearly the same melody of every small part. Also at the beginning of every part this is always a stress which is the reason why every part of the waveform has larger magnitude at the beginning and then attenuates to small value. Then we could use ‘fft’ to get the waveform in frequency domain. The interval of frequency spectrum is the ration of sampling frequency with length of the signal. With these parameters waveforms of frequency domain could be obtained. The ‘fft’ function returns the discrete Fourier transform (DFT), computed with a fast Fourier transform (FFT) algorithm. Since we have known all the values

of the signal, it is not hard to use the algorithm to get the DFT of it.

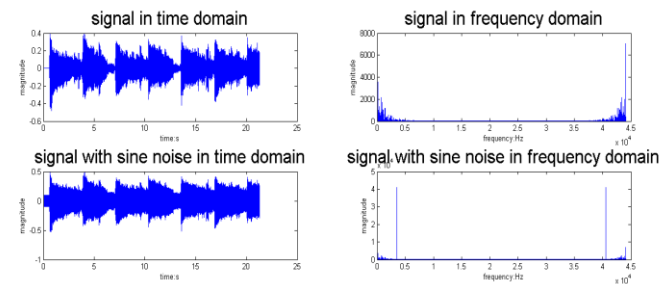


Fig. 1. signal analysis of original signal with and without noise.

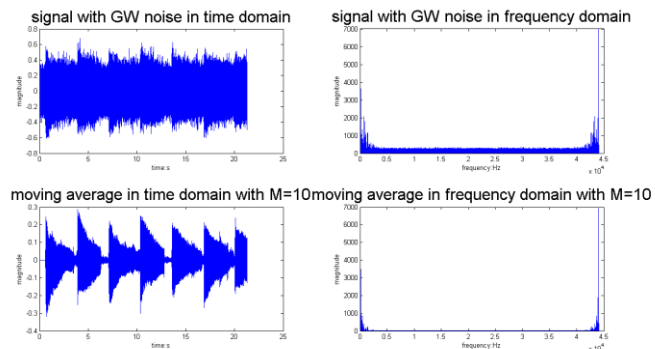


Fig. 2. signal analysis of original with Gaussian noise and moving average.

B. Signal with noise

After the process for the analysis of the original signal, we need to find the characteristic of the signal adding with noise. The first noise function is chosen to be:

$$\text{Noise1} = 0.1 * \sin(3500 * 2 * \pi * t)$$

Then waveform in time and frequency domain with noise is above. Play the signal again after adding the noise we could hear the audio with harsh noise. For time domain we only need to plot the results of signal adding the value of noise. For frequency domain, it only need do the Fourier Transform of the signal with noise. For results analysis in frequency domain it is easy to understand. Since we add the noise whose frequency is 3500 Hz, we could see not only there is some influence in waveforms but also there is an impulse frequency

in 3500 Hz in the figure. For the second noise I choose Gaussian White Noise. Then the sound of the signal after adding the noise has rustle but not very harsh. The figures are shown above. In Matlab it is easy to get Gaussian white noise by using the random function '(randn(m,n))'. Hence the noise function is:

$$\text{Noise2} = 0.1 * \text{randn}(1, N)$$

N is the length of the signal. Then analyze the waveform in time and frequency like before. From the figure we could find it has larger difference than the sinc frequency noise with the original signal figure. Both noises have the same amplitude. Since the function above is to get the random Gaussian noise and it cannot be controlled. At last the waveforms in time and frequency of the moving average for the original signal with M=10 are given. From the figures we could find that they are really nearly the same with the original signal waveforms. Because we choose M=10 here. The waveforms of original signal do not have many large ups and downs. Hence if we make M not be very large we will get the very similar figures as before. However, when we choose M to be large enough such as 50 we find the magnitude to be smaller than before obviously.

III. FIR FILTER DESIGN

We have two different noises and we need to design two different filters. The frequency of the first noise is 3500 Hz. According to this frequency we have got the design parameters. The cut-off frequency can be chose arbitrarily only need to be close to frequency of noise. Then after designing the bandstop filter by using Barthanwin window, we could also use the 'freqz' function to get the frequency response of the filter. Also the impulse response of the filter could be obtained by 'ideal_lp' function. At last 'freqz_m(h_bs,1)' can be used to get the frequency characteristic. All figures for the parameters of the filter are list below. The band pass cut-off db value I choose is 35db. Hence we could find the lowest value is -35db when $w/\pi=0.15$. The linear magnitude characteristic has the only difference of unit with the former one. The results are the same as other figures and they are both obtained by the 'freqz_m(h_bs,1)' function in Matlab to get the frequency characteristic.

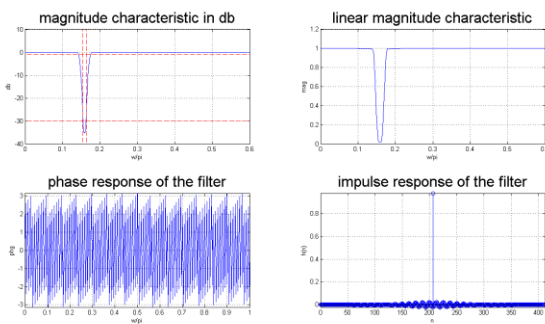


Fig. 3. magnitude characteristic in unit of db.

For the Gaussian White noise another filter needs to be designed. I designed three kinds of filter which are lowpass, highpass and bandpass to see which one is the best one. Both

lowpass and highpass are designed by 'fir1' in matlab with 'kaiserord' to get the order and cut-off frequency. For bandpass we could use '[b, a]=butter(n, Wn, 'ftype', 's')' to get the order and cut-off frequency of filter by butter. In the next part analysis of the signal with Gaussian noise after filter will be given.

IV. NOISE SIGNAL ANALYSIS AFTER FILTER

Since we have already finished designing the filter we need to use the filter make the noise clear and to see whether the filter works well. For the sinc noise we use the function below:

```
>> f_y1=filter(h_bs,1,y1); >>f_Y1=abs(fft(f_y1));
```

The comparison of waveforms in time and frequency domain of original signal and noise after filter are below:

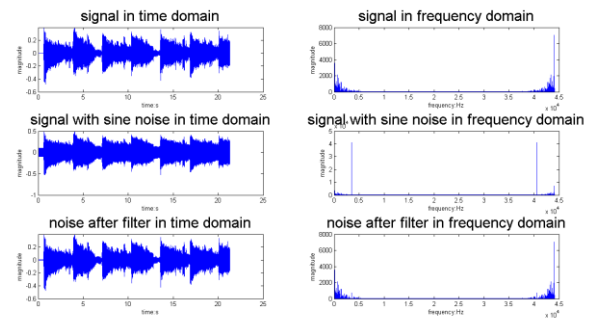


Fig. 4. Comparison of signal before and after filter

From the figures we could find the magnitude of signal is larger than before when adding noise in time domain. After using the filter the magnitude of the signal has attenuated a little and the waveform of signal after filter is nearly the same as the original signal which shows the filter works well. We could also see there is an impulse at 3500 Hz in the figure of signal with noise in frequency domain obviously. However, the impulse in the frequency domain figure of signal with noise after filter has disappeared. The waveform is also nearly the same as the original signal in frequency domain. If we play the signal after filter we could hear the audio signal clearly without the harsh noise. All these have proved that the design of the filter is successful.

For the Gaussian White noise we have designed three kinds of filter. The task is to choose the best one. Waveforms of the signal before and after each filter are given below:

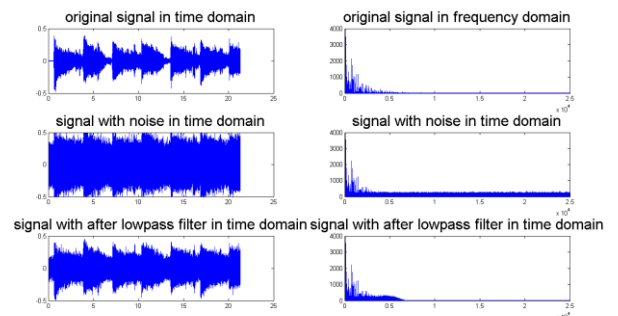


Fig. 5. Comparison of signal with noise and signal after lowpass filter.

From fig5 we could see after adding the noise to the signal the magnitude for the waveform of the signal is much larger than that of original signal. After using the lowpass filter we could find that most part of the noise frequency has been eliminated to the value of the original signal except in the low frequency region. It retains most of the waveform of the original signal. However, as said before, there is still some interference in the low frequency region since it is lowpass filter. Hence, if we play the signal after through the filter we could hear the rustle sound is lighter than before but still cannot as good as the original signal.

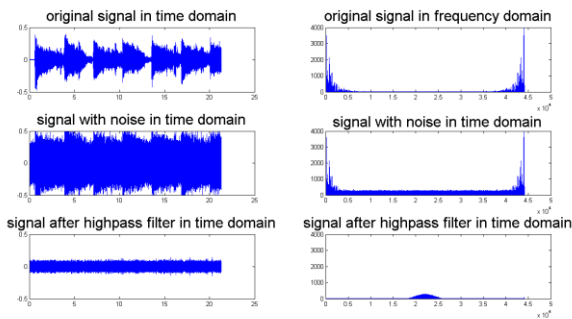


Fig. 6. Comparison of signal with noise and signal after highpass filter.

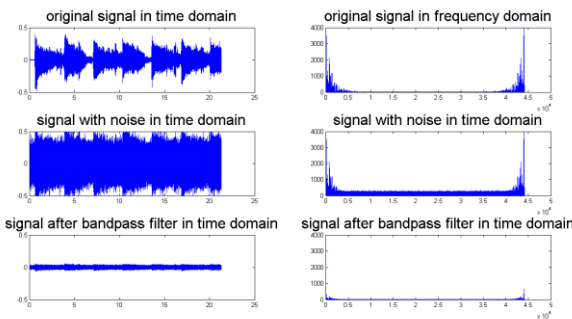


Fig. 7. Comparison of signal with noise and signal after bandpass filter.

From fig6 and fig7 we could see that the highpass and bandpass filters are not good for the signal. After the signal going through the highpass filter the only left is some high frequency noise. The filter not only filters some noise but also with the some part of the original signal. If we play the signal again we could here nearly nothing for highpass filter. Hence we could not get the signal with this filter. The reason is also fit for the bandpass. It eliminates some part of the original signal at low frequency. The sound of signal for bandpass filter is much more obscure than before. So the best filter for it is the lowpass filter.

V. DETECTION OF SIGNAL

A. Principles of speaker recognition

In this part the detection of signal which is how to distinguish different people's voice will be designed. The goal is to

determine who the speaker according to the voice of database is.

● Extraction of feature

We could use the MFCC method in Matlab to get the parameters.[1] It is a method using variation of human ear's bandwidths with frequency. The sampled signals can capture all frequencies up to 5 kHz, which cover most energy of sounds that are generated by humans. The feature can be obtained by extracting the important characteristics of speech in the Mel-frequency scale. The procedure of extraction is below:

1. Frame blocking, windowing and fast Fourier transform of the signal
2. Get the square of magnitude of frequency spectrum to get the power spectrum.
3. Let the power spectrum through a lot of triangle filter to get the log power of every filter.
4. Use discrete cosine transforms (DCT) to get the MFCC coefficients. The coefficient is always in the interval of 20~30. Here we choose it to be 20. [2]

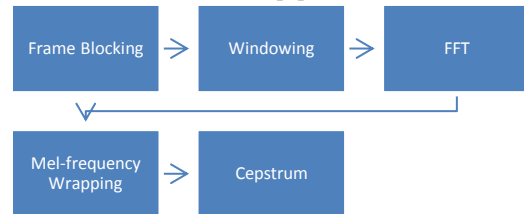


Fig. 8. Block diagram of the MFCC

● Clustering the training vectors

We have got each speaker's voice to be represented as specific codebook of different speaker. The algorithm here is called LBG for clustering a set of L training vectors into a set of M codebook vectors. The procedure of algorithm is below:

1. Take the centroid of the entire set of training vectors to be the first 1 vector codebook B1.
2. Double the size of the codebook by splitting each current codebook according to the rule:

$$\begin{cases} B_m^+ = B_m(1 + \varepsilon) \\ B_m^- = B_m(1 - \varepsilon) \end{cases} [3]$$

Here $\varepsilon = 0.01$

3. According to the codebook classify all training vector. Calculate the distortion D and the relative distortion by the equations below. If relative distortion is smaller than the coefficient ε the procedure ends. The codebook current is the double size of the codebook. Then go procedure 5. Otherwise, go procedure 4.

$$distortion: D^{(n)} = \sum_{k=1}^K \min d(X_k, B) [4]$$

$$relative\ distortion: \left| \frac{D^{(n-1)} - D^n}{D^n} \right| [4]$$

4. Compute the new centroid to get new codebook then go procedure 3.
5. Iteration 2, 3 and 4 until a codebook size of M is designed. D0=10000.[2]

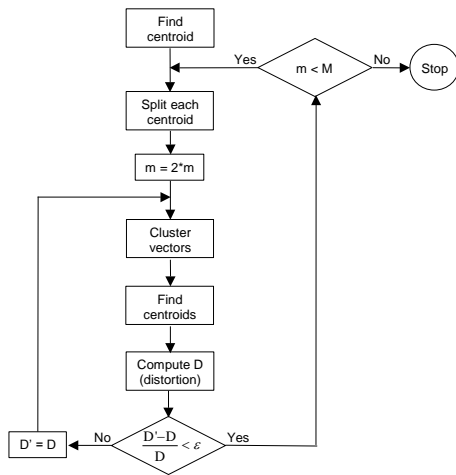


Fig. 9. Flow diagram of the LBG algorithm [5]

● **Identification of speakers' voice for VQ**

Let the training vector of speaker to be $\{X_1, \dots, X_T\}$. Every codebook has M codons. Compute the average distortion D of the speaker and take a value of ϵ .

$$D = 1/T \sum_{j=1}^T \min \left[d \left(x_j, B_m \right) \right] \quad [6]$$

If D is smaller than ϵ then it is the former speaker.

B. Voice analysis

We have all the methods of the voice identification and we need to accomplish it in Matlab. I choose 10 different speakers' voice including English and Chinese. The format is 'wav'. I put them in the same file. Then record the same speakers' voice of longer time and put them in another file. This is to test and verify whether the system could work well. The result is good. Of the 10 group of data, the system could recognize 9. The error rate is 10%. However this test is not very accurate to represent recognizing voices because we only tested 10 speakers' voices with a little time of each voice record. If the system has a large number of data, it may not have such high accurate result. Then some analysis of the system will be given.

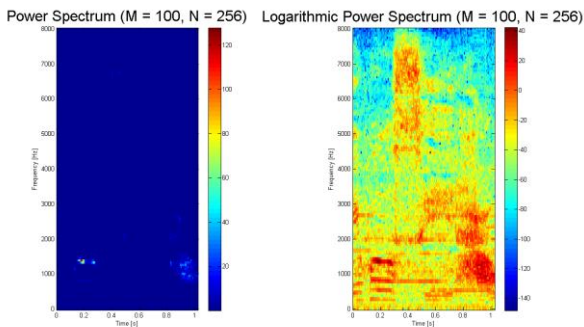


Fig. 10. Power spectrum of the first signal s1

Fig10 shows that the areas containing the highest level of energy is the color of red. Hence from the plot we could find that the red area is all of the time and most of the energy is concentrated in the frequencies between 50 Hz and 2500 Hz.

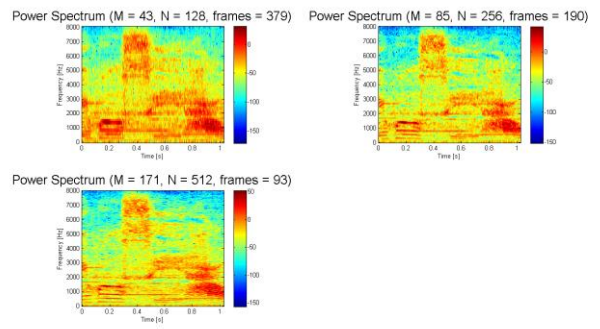


Fig. 11. Power spectrum of the first signal s1 with different N and M

Fig11 shows that when $N=128$ the time has a high resolution and poor frequency resolution. Each frame lasts a very short period of time. When $N=256$ there is a compromise between time resolution and frequency resolution. When $N=512$ the frequency resolution is very high. The results also fit what we have learned in class.

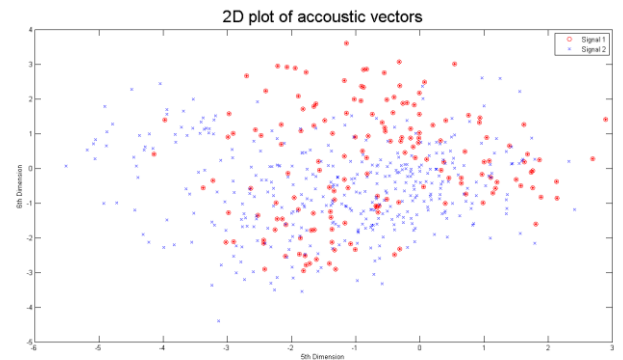


Fig. 12. 2D plot of acoustic vectors of two signals.

From Fig12 we could see that most of the areas of the two signals overlap. However, there are still some areas that can be used to identify the different speakers of the two signals. For example in some top areas we could find the first signal. For some left areas can could only see the second signal. From the results we can see even though there are some areas that two signals overlap we can still identify who the speakers are by those areas only have one signal. Furthermore it is only a 2-D plot. In fact it has 20 dimensions. [7]

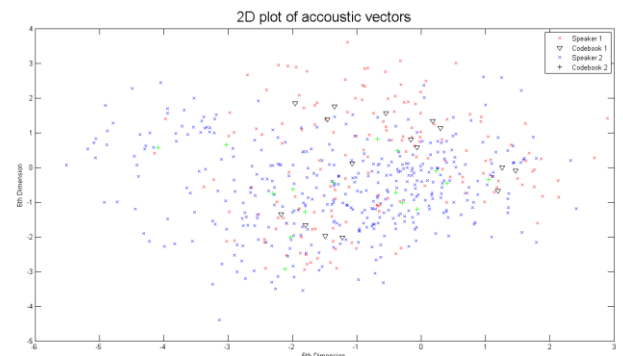


Fig. 13. The resulting VQ codewords after function vqlbg.

For fig13 it shows the resulting VQ codewords with the signal of two different speakers. The clusters can be identified not very difficult. Still this is a 2-D plot.

Speaker 1 matches with speaker 1
Speaker 2 matches with speaker 2
Speaker 3 matches with speaker 3
Speaker 4 matches with speaker 4
Speaker 5 matches with speaker 5
Speaker 6 matches with speaker 7
Speaker 7 matches with speaker 7
Speaker 8 matches with speaker 8
Speaker 9 matches with speaker 9
Speaker 10 matches with speaker 10

Fig. 14. The result of the identification

VI. CONCLUSION

This project has two parts. The first part is about the signal and noise analysis. After adding sinc frequency noise and Gaussian White noise into the signal and doing analysis to see the difference with the original signal. Design the special filter according to the kinds of noise. The results show that the filter designed by Barthannwin window has a good function to filter the sinc frequency noise. For Gaussian White noise three different filters which are lowpass, highpass and bandpass are given. However, only the lowpass filter can make the noise to an acceptability level. For the second part the detection of signal in noise has been analyzed. With the design of system before the results of voices identification show the system works well. The error rate is 10%. However our database is not large enough to show the actual accuracy of the system. For improvement it can add more data into the system. In summary it is a good system to identify different speakers'

voice. For four DSP Tools used in this project, three are from in class which are: **a. Moving average of linear systems.** **b. Filter design.** **c. Frequency domain analysis.** For DSP Tools out of class **Detection of signal in noise**

VII. REFERENCE

- [1] Muda, Lindasalwa, Mumtaj Begam, and I. Elamvazuthi. "Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques." arXiv preprint arXiv:1003.4083 (2010).
- [2] Marchetto, Enrico, Federico Avanzini, and Federico Flego. "An automatic speaker recognition system for intelligence applications." Proceedings of European Signal Processing Conference (EUSIPCO'09). 2009.
- [3] Reynolds, Douglas A. "Automatic speaker recognition using Gaussian mixture speaker models." The Lincoln Laboratory Journal. 1995.
- [4] Reynolds, Douglas. "An overview of automatic speaker recognition." Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)(S. 4072-4075). 2002.
- [5] L.R. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, N.J., 1993.
- [6] Soong, Frank K., et al. "Report: A vector quantization approach to speaker recognition." AT&T technical journal 66.2 (1987): 14-26.
- [7] Reynolds, Douglas A., Thomas F. Quatieri, and Robert B. Dunn. "Speaker verification using adapted Gaussian mixture models." Digital signal processing 10.1 (2000): 19-41.