REPORT ON
DIGITAL SIGNAL PROCESSING
ASSIGNMENT 1
AUDIO EQUALIZER

Xiong ChenYu
U1521516C
EEE

Oct. 20, 2017

Contents

List of Tables

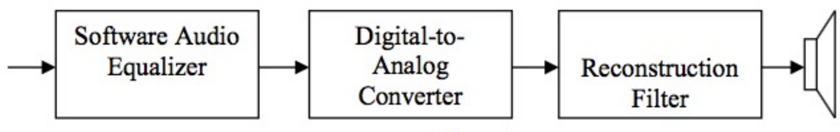
List of Figures

1 Introduction

1.1 Background

In the information ages everything is store in digital, such as the picture, the video and the sound. To play the sound from the digital to the corresponding real analog signal, we need the signal system to do such things. First we need Software Audio Equalizer to enhance the sound and use a digital to analog converter to convert the digital to analog with the minimum noise added in Figure 1 And the last step we need a reconstruction filter to filter out the high frequency noise and connect to speaker.

Figure 1: Audio System



1.2 Motivation

This semester I take the EE4413 Digital Signal Design, So I learn different ways to build IIR and FIR filters design to build low/high pass band filter. Compare to IIR the FIR have linear phase which means FIR is good to design the sound system without distortion[1].

I decide to build a FIR system to enhance and reduce this 8 bands to get better music effect as in Table 1.

Table 1: Bands

Band #	Low	High Frequency(Hz)
1	0	200
2	200	400
3	400	800
4	800	1600
5	1600	3200
6	3200	6400
7	6400	12300
8	12300	22000

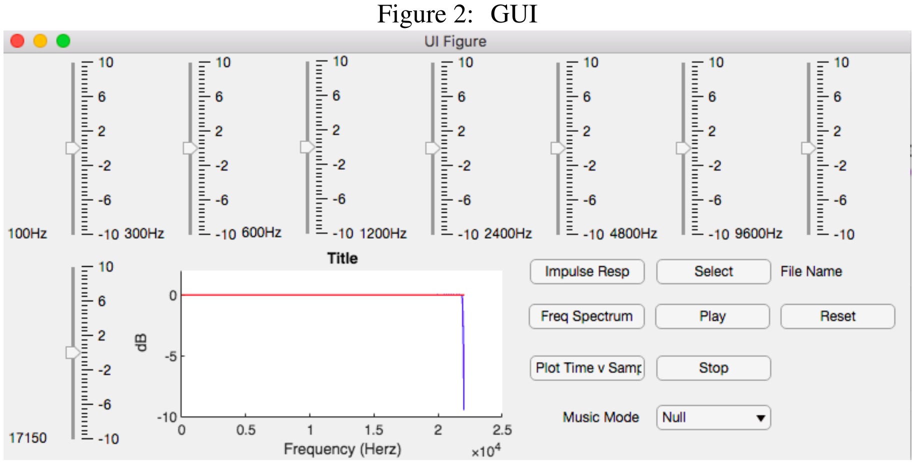
1.3 Objective

The objective is to design and simulate an audio equalizer using Mat lab by using 512-tap FIR filter structure.

2 Design and system construction

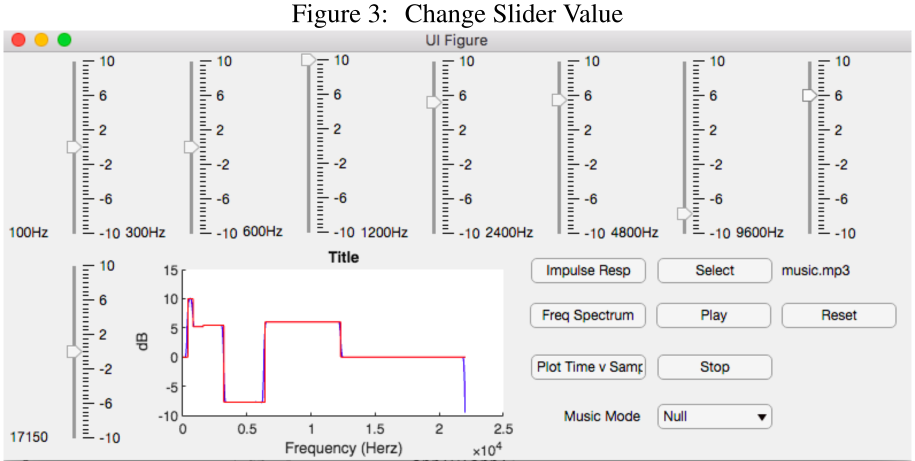
2.1 GUI design

Figure 2: GUI



For the graphic user interface, I use 8 sliders to control the different frequency bands. You can find it at Figure 2.

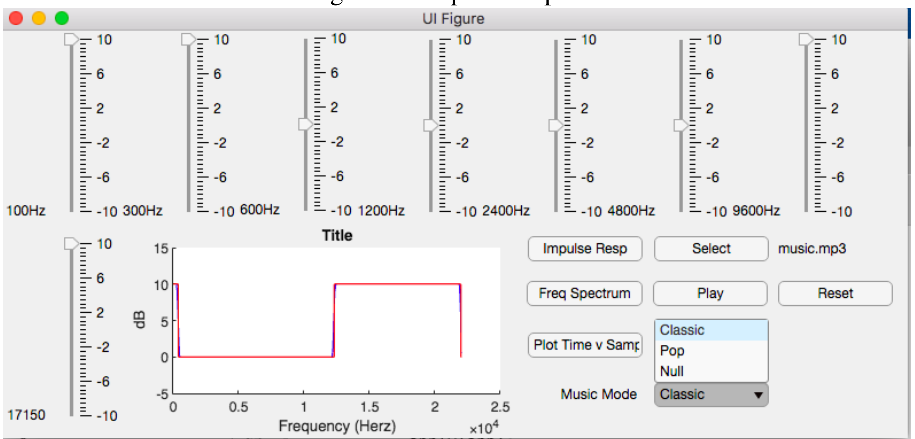
Figure 3: Change Slider Value



The sliders is used to control the different frequency range from -10 DB to 10 DB, and the label show the mid frequency of the band. Every time when I change the frequency the FIR filter will be automatic regenerate. Like in Figure 3.

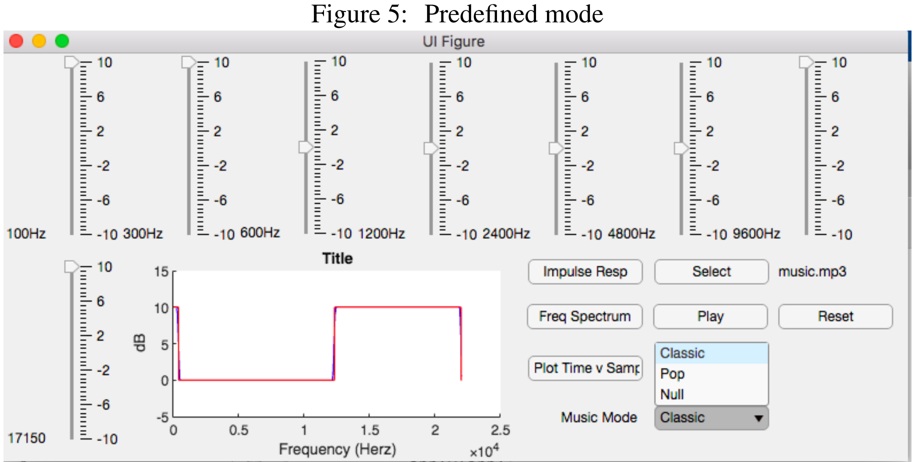
In the middle there is a 2-D graph to display the required response and spectrum. The default graph it the frequency response of the FIR filter and the ideal frequency

Figure 4: Impulse response



response. If you want to see the impulse response in time domain just click the Impulse Resp button. Then you can get it as in Figure 4.

Figure 5: Predefined mode



On the right side are some control buttons and a select box. For the select box below there are some predefined filter such as pop music(increase between 500Hz and 4000 Hz). You can choose it and the specified filter will be applied. As in Figure 5.

After I create the desired FIR filter. I can use the select button to select a MP3 audio file. Show in Figure 6. The audio file then will be convert to 2 dimension matrix which stand for the left and right Channel. Then I can use play button to enjoy the enhanced music.

I can stop it and analysis the filter. First is to plot the 2 frequency spectrum of the origin and filtered data as in Figure 7. And also the first 10 second data samples of the input and output. As in Figure 8.

Figure 6: Load Digital Data

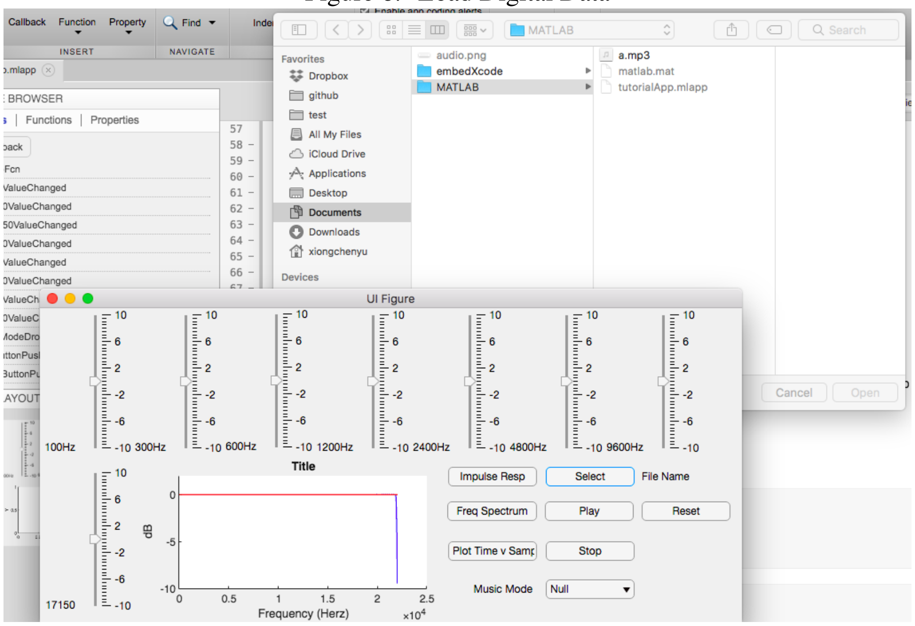


Figure 7: Frequency Spectrum

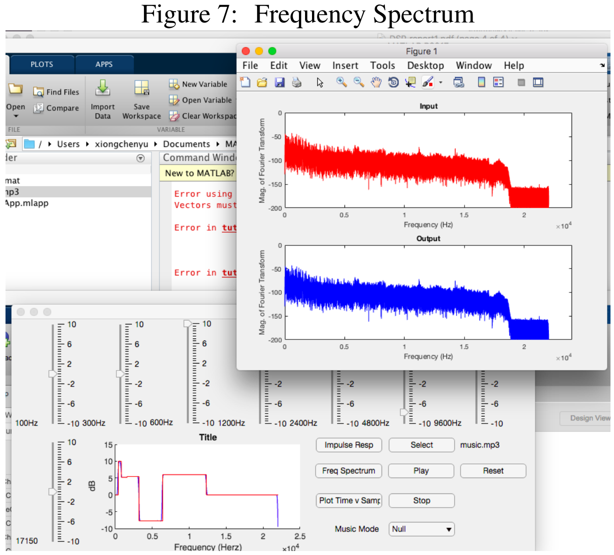
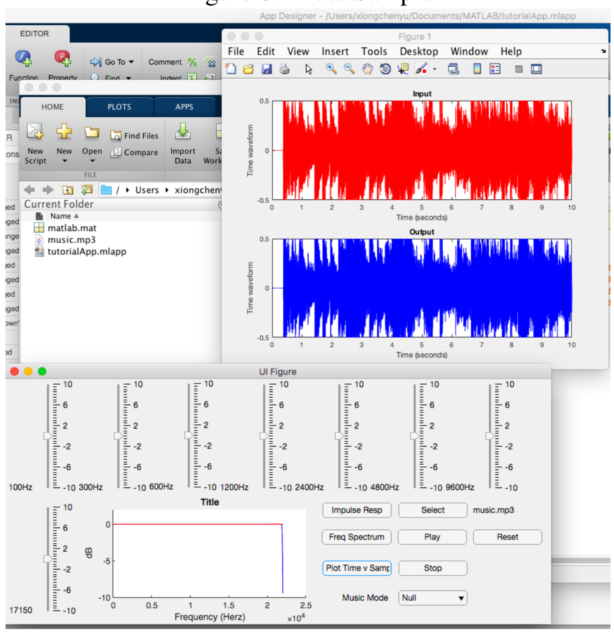


Figure 8: Data Sample



2.2 System Construction

The system is configured by change the sliders to control the gain. And load digital signal by choose a MP3 file to get input. First you should make a desired gain by each frequency range, and the coefficient will be auto calculated, then you can choose a music by the time you select the music the filtered data will be automatic generate by the filter. Then you can play the filter signal and stop it, what's more you can easily analysis it.

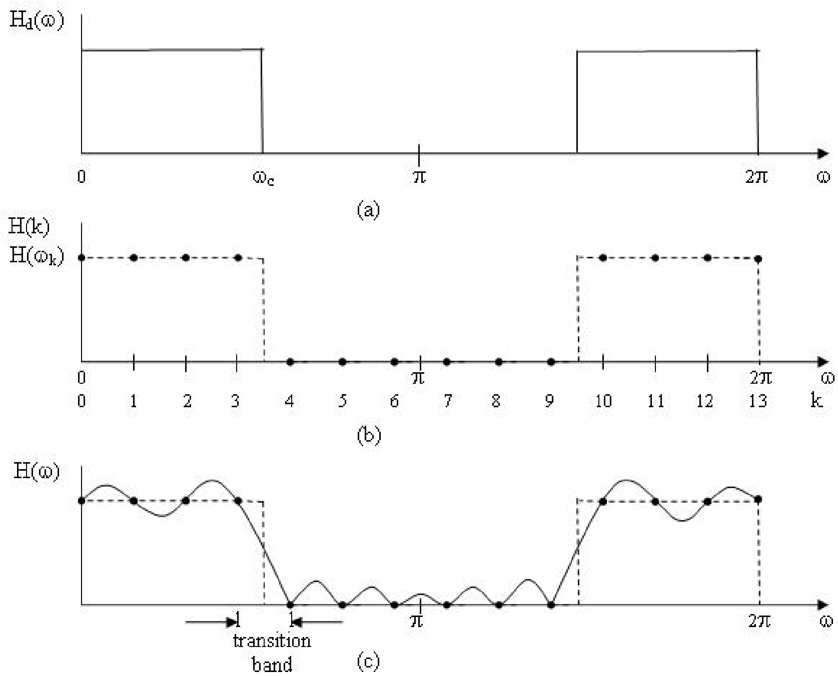
3 Algorithm and simulation

The frequency response of ideal FIR filter is the block wave, but it is unrealistic to compute in the computer because the reverse fourier transform of a squire block is *sinc* which is infinite in time domain. One of the most famous way is to use window function in this infinite impulse response trim the transfer CCDE to finite then do fourier transform to the CCDE trimmed by window function. The window function is strait forward and can easy design simple high/low/band pass/stop filters. But this method is complicated to design an arbitrary filter here.

Luckily, matlab offers a great FIR function fir2 which use the frequency sampling design method. Below shows the algorithms of frequency sampling method. See Figure 9.

$$H_d(e^j\omega) \Rightarrow H_d(K) \Rightarrow h(n)$$

Figure 9: Frequency Sampling Method



$$\omega = \frac{2\pi k}{N}$$

Then we can use IDFT to get the transfer function h(n).

$$h(n) = \frac{1}{N}\sum_{k=0}^{N-1} H_d(k) W_N^{(-nk)}$$

Since the tap is 512 so the order of the filter is 512 -1 = 511. So the value 511 is odd To ensure the coefficient of h(n) is always real. We can use the equation below[2] to get the real h(n).

$$h(n) = \frac{1}{N}[H(0) + 2\sum_{k=1}^{\frac{N-1}{2}} Re(H(k)e^{\frac{j2\pi an}{N}})]$$

So we can get the real h(x) coefficient by frequency sampling method.

The simulation can be seen in Figure 5.

Figure 10: Maximum of origin data and filtered data

```
ans =

    0.9990    1.0000


ans =

    2.9367    3.1462
```

```
app.filtered = filter(app.fir2_no, 1, app.origin);
max(app.origin)
max(app.filtered)
```

You can see from Figure 10. The maximum of the MP3 amplitude is 1, and the maximum of data filtered is 3.1. To prevent the possibility of signal clipping, normalize the filtered signal before it is output to the DAC. Rather than use normal **sound** function. I use **soundsc** function to normalize the date before play it. soundsc(y) scales the values of audio signal y to fit in the range from –1.0 to 1.0, and then sends the data to the speaker at the default sample rate of 8192 hertz. By first scaling the data, soundsc plays the audio as loudly as possible without clipping. The mean of the dynamic range of the data is set to zero. [3]

4 Results, analysis and discussion

To analysis the result, first we need to compare the input and output sample data when gain equal to 0 DB. And it is the same in Figure7.

Figure 11: Classic Frequency Response

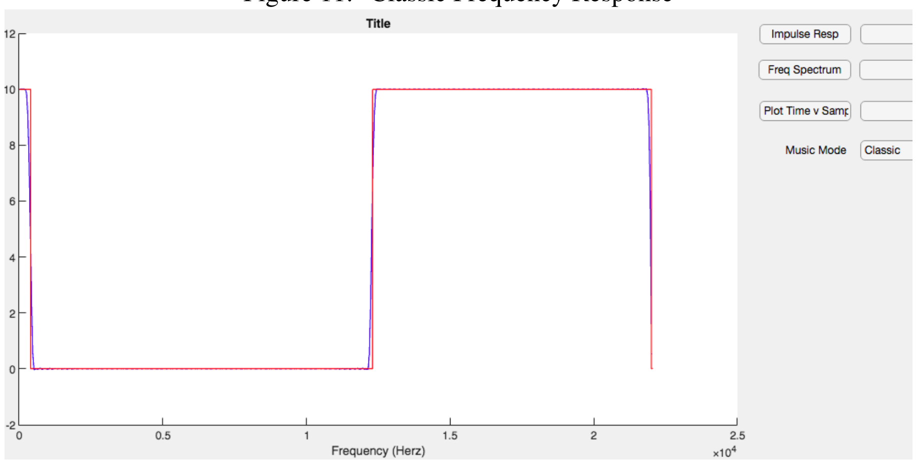
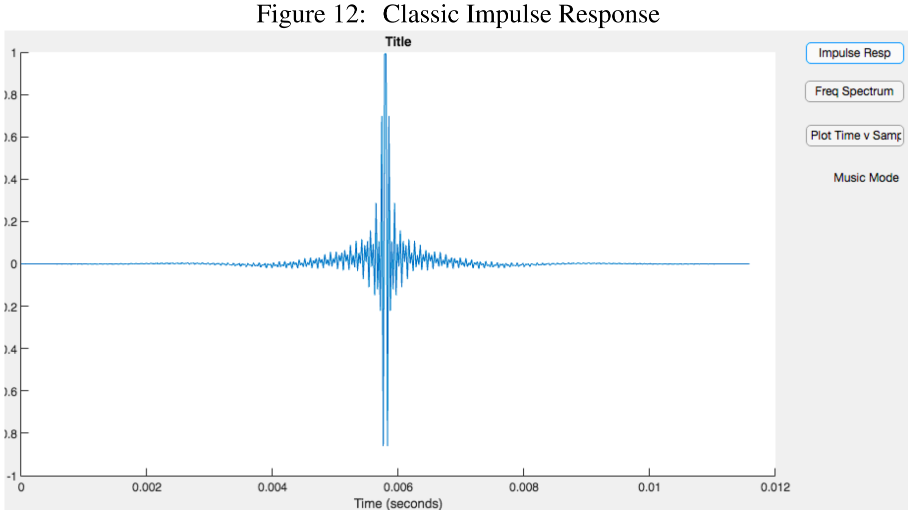


Figure 12: Classic Impulse Response



And let us analysis the 2 condition classic music filter and pop music filter. First let us see the frequency response at Figure 11 and Figure 13. The red line stand for the ideal FIR filter and the blue line is the actual filter. You can see the real FIR filtered transition edge is a vertical line which means it pass band equal to stop band. Transition band equal to 0. And for the real FIR the transition band at high frequency is larger than the transition band in low frequency.

Figure 13: Pop Frequency Response

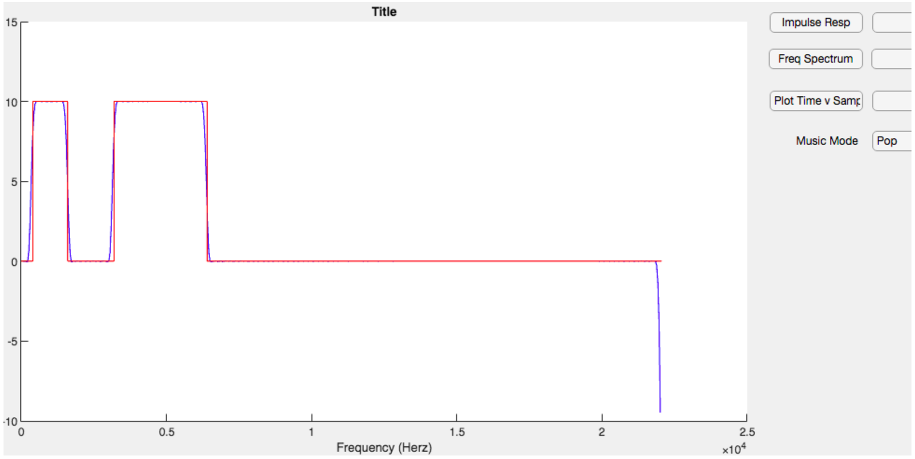
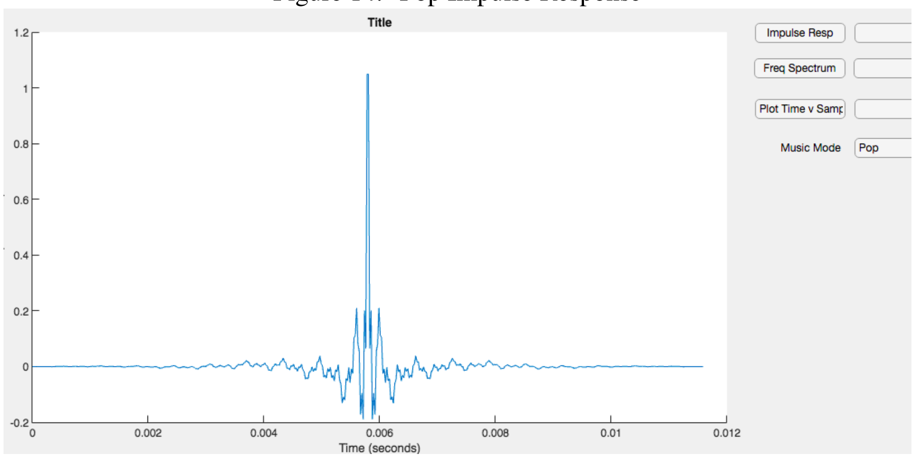


Figure 14: Pop Impulse Response



As for the impulse response Figure 12 and Figure 14. For the classic filter which enhance the high frequency the frequency of the wave is high while the pop filter frequency is low. And since the classic filter enhanced the frequency below 500Hz and above 4000Hz. The pop filter enhance the frequency between 500Hz and 4000Hz. So the filter is orthogonal there wave can cancel each other out. So the impulse response of classic music filter and pop music filter add together is the impulse function.

5 Conclusions and recommendations

This Frequency sampling method designed FIR filter is good and works well but there are some other good design method can be implement.

I just try the frequency sampling method in this report and use hamming function, I the future can try some other window function to decrease the signal leakage.

I also recommend to use all kind of design method such as Window design method, Weighted least squares design, Parks-McClellan method etc. To compare the similarity and difference.

References

[1] EN.WIKIPEDIA.ORG. (2017). Finite impulse response. [online] Available at: https://en.wikipedia.org/wiki/Finite$_{i}mpulse_{r}esponse[Accessed 20 Oct.2017]$.

[2] YOUTUBE. (2017). How to Design FIR FIlters using frequency sampling method. [online] Available at: https://www.youtube.com/watch?v=OODbpZqhPSg [Accessed 20 Oct. 2017].

[3] MATHWORKS.COM [online] Available at: https://www.mathworks.com/help/matlab/ref/soundsc.html?searchHighlight=soundscs$_{t}id = doc_{s}rchtitle[Accessed 20 Oct.2017]$.