Mini Project: Computer Music Generation and Playing

Introduction

The objectives of this lab is to use $Matlab$ to generate frequencies corresponding to the notes, sine waves corresponding to the frequencies above and then the whole music. Besides, another two important factors are needed to be considered, which are strength and harmonic waves. For the former one, envelope attenuation should be cared. For the letter one, the distribution of energy at different frequencies should be cared.

Result and analysis

2.Digital chords

In digital chords, basic notes $1, 2, 3, 4, 5, 6, 7$ are used to represent the seven basic levels of the scale, which are named do、re、mi、fa、sol、la、ti(si in China). And 0 represents the pulse. Other symbols are also contained in digital chords, such as dot and horizontal line. Numbers and symbols determine the frequency and duration of each sound.

2.1 Tones

Different numbers represent different tones and each number is divided into three tones, high, middle and low, corresponding to different frequencies of sound waves. The following figure shows the corresponding table of C notes and frequencies.

	1（C）	2(D)	3(E)	4(F)	5(G)	6(A)	7(B)
低音	262	294	330	349	392	440	494
中音	523	587	659	698	784	880	988
高音	1046	1175	1318	1397	1568	1760	1976
相邻音符频率比 $n+1/n$	1. 12	1. 12	1. 057	1. 12	1. 12	1. 12	1. 058

From the figure, what is obvious is that the low frequency is half the frequency of the same note. Similarly, the middle frequency is half the high frequency of the same note. Halving the frequency means that the interval is one octave apart. The dot marked below the mark of the basic note, called the bass point, represents lowering the basic note by a pure octave. Two dots indicate a reduction of two pure octaves. Similarly, the dot marked above the mark of the basic note, called the bass point, represents raising the basic note by a pure octave. And the two dots represent the basic note raised by two pure octaves.

Another discovery is that the ratio of frequencies between adjacent notes is not exactly equal.(The ratio of frequencies about $2:1$, $3:2$, $5:4$, $6:5$, $7:6$ are approximately equal to $2^{\frac{1}{6}}$,while that of frequencies about $4:3$, $1i:7$ are approximately equal to $2^{\frac{1}{12}}$.) According to `Law of twelve equals`, a group of notes (which is also called octaves) divided into twelve semitone chromatic intervals($2^{\frac{1}{12}}$). So, the interval between $3-4$ and $7-1i$ is a semitone while the interval between The interval between two other adjacent notes is whole tone, which is equal to two times of semitone.

What's more, the symbols `#` and b represent raise and falling tones. Raising tone means raising the original level one semitone and falling tone means reducing the original level one semitone. Below is a detailed table of C notes and frequencies.

音符	频率/Hz	音符	频率/Hz	音符	频率/Hz
低音1	262	中音1	523	高音1	1046
低音1#	277	中音1#	554	高音1#	1109
低音2	294	中音2	587	高音2	1175
低音2#	311	中音2#	622	高音2#	1245
低音3	330	中音3	659	高音3	1318
低音4	349	中音4	698	高音4	1397
低音4#	370	中音4#	740	高音4#	1480
低音5	392	中音5	784	高音5	1568
低音5#	415	中音5#	831	高音5#	1661
低音6	440	中音6	880	高音6	1760
低音6#	466	中音6#	932	高音6#	1865
低音7	494	中音7	988	高音7	1976

Considering the above method as the encoding method, the function $tone2freq$ is written.

```
function freq = tone2freq(tone, noctave,rising)
freq=440;
    switch tone
        case 0
            freq=0;
        case 1
            freq = freq*power(2,0);
        case 2
```

```
            freq=freq*power(2,1/6);
        case 3
            freq=freq*power(2,1/3);
        case 4
            freq=freq*power(2,5/12);
        case 5
            freq=freq*power(2,7/12);
        case 6
            freq=freq*power(2,9/12);
        case 7
            freq=freq*power(2,11/12);
    end
    freq=freq*power(2,noctave);
    freq=freq*power(2,rising/12);
end
```

In this function, $1 = 440Hz$ is as the tonic, $tone$ means number tones, whose range is $[0\ 7]$ (0 means pulse.), $nocatve$ means the number of high or low octaves.(0 means alto voice, positive numbers represent high the value of $noctave$ octaves and positive numbers represent low the value of $noctave$ octaves, one octaves means 2^1), $rising$ means raising or falling tones(1 means raising tones, -1 means falling tones, 0 means no raising or falling tones), and $freq$ means the output of frequency.

2.2 key signature

In 2.1, the other frequencies of notes can be calculated given the frequency of one note. At the beginning of the Number Musical Notation, there are marks like $1 = C$ or $1 = D$, which means the numerical notation is written in the key of C or D and also specifies the frequency of note 1 in the notation. The frequencies of note 1 in different key signature are shown in the table below.

C 调频率	261.5	293.5	329.5	349	391.5	440	494
C 调音符 (英文名)	1(C)	2(D)	3(E)	4(F)	5(G)	6(A)	7(B)
C 调	1=C (261.5)						
D 调		1=D (293.5)					
E 调			1=E (329.5)				
F 调				1=F (349)			
G 调					1=G (391.5)		
A 调						1=A (440)	
B 调							1=B (494)

Combining the table with the encoding method in 2.1, the function *tone*2*freq*2 is written.

```
function freq = tone2freq2(tone,scale,noctave,rising)
    freq=261.5; %initial the value of freq to C
    switch scale
        case 'C'
            freq=freq*power(2,0);
        case 'D'
            freq=freq*power(2,1/6);
        case 'E'
            freq=freq*power(2,1/3);
        case 'F'
            freq=freq*power(2,5/12);
        case 'G'
            freq=freq*power(2,7/12);
        case 'A'
            freq=freq*power(2,9/12);
        case 'B'
            freq=freq*power(2,11/12);
    end

     switch tone
        case 0
            freq=0;
        case 1
```

```
                freq = freq*power(2,0);
            case 2
                freq=freq*power(2,1/6);
            case 3
                freq=freq*power(2,1/3);
            case 4
                freq=freq*power(2,5/12);
            case 5
                freq=freq*power(2,7/12);
            case 6
                freq=freq*power(2,9/12);
            case 7
                freq=freq*power(2,11/12);
        end
        freq=freq*power(2,noctave);
        freq=freq*power(2,rising/12);
    end
```

In this function, the initial value of frequency is 261.5. $Scale$ means key signature and the meaning of other parameters is the same as those in 2.1.

2.3 Length of note

In the numerical notation, there are some representations indicating the length of note.

The short horizontal line to the right of the base note indicates doubling the length of the note.

The short horizontal line below the base note indicates that the length of the note is reduced by half .

The small dot to the right of the base note indicates lengthening the note by half its original length.

3.Generating waveforms of different frequencies

Synthesizing all the introductions in $Section$ 2, the function of generating waveforms, gen_wave is written.

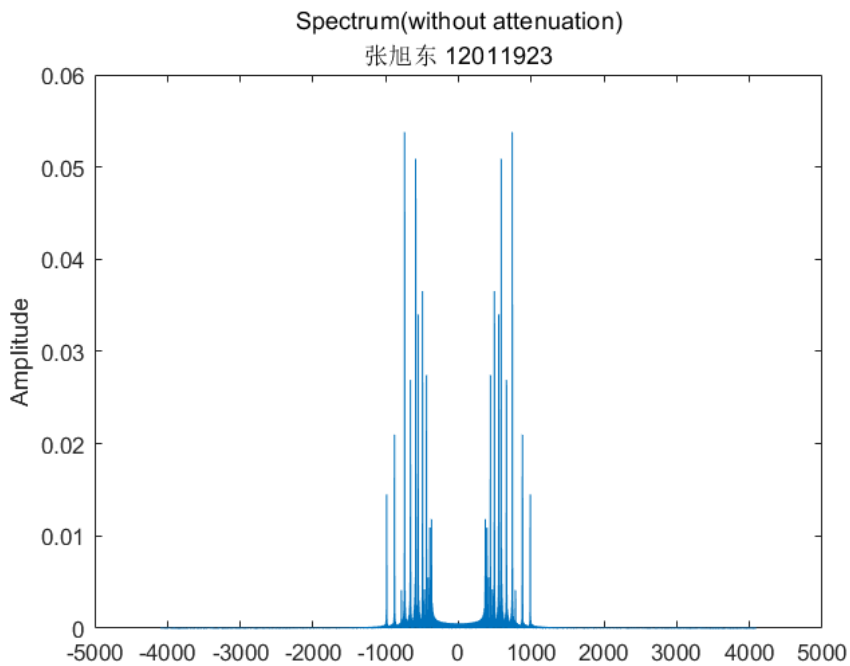
```
function [waves,
wavess]=gen_wave(tone,scale,noctave,rising,rythm,fs)
    freq=261.5; %initial the value of freq to C
    switch scale
        case 'C'
            freq=freq*power(2,0);
        case 'D'
            freq=freq*power(2,1/6);
        case 'E'
            freq=freq*power(2,1/3);
        case 'F'
            freq=freq*power(2,5/12);
        case 'G'
            freq=freq*power(2,7/12);
        case 'A'
            freq=freq*power(2,9/12);
        case 'B'
            freq=freq*power(2,11/12);
    end

     switch tone
        case 0
             freq=0;
        case 1
            freq = freq*power(2,0);
        case 2
            freq=freq*power(2,1/6);
        case 3
            freq=freq*power(2,1/3);
        case 4
            freq=freq*power(2,5/12);
        case 5
            freq=freq*power(2,7/12);
        case 6
            freq=freq*power(2,9/12);
        case 7
            freq=freq*power(2,11/12);
    end
    freq=freq*power(2,noctave);
    freq=freq*power(2,rising/12);
    t=linspace(0,rythm,fs*rythm);
    waves=sin(2*pi*freq*t);
```

```
    %attenuation
    wavess=waves.*exp(-2*t/rythm);
end
```

The parameter $rhythm$ means the duration of each note and the parameter f_s means sample frequencies, while the meaning of other parameters is the same as those in 2.2. The output $waves$ means the generated waveforms without envelope attenuation while the output $waves$ means the generated waveforms with envelope attenuation, which is easy to use in $Section$ 4.

What's more, the song, $Castle\ in\ the\ Sky$ is generated according to function gen_wave. According to the Number Musical Notation, each tone is generated and finally all of them are joint together to get the song.

The spectrum of $Castle\ in\ the\ Sky$ without envelope attenuation is below.



However, although the song sounds like $Castle\ in\ the\ Sky$, it is really strange because of the difference with the actual sound of the real instrument. So changes must be done.

There are details needed to be considered. When using $audiowrite$ to write it to the music file, f_s should be 44100 otherwise the music player would play noise. Because some music players cannot recognize the sample rate.

4. Exercise 4 , envelope attenuation

In real life, the vibration will experience attenuation and will not continue to vibrate at a fixed amplitude when playing instruments. For this reason, envelope attenuation functions are used to simulate music generation in real life.

This part is divided into two parts. The first part is to compare performances of different parameters in exponential attenuation and the second part is to compare performance of three different envelope attenuation functions, which are exponential attenuation, linear attenuation and square attenuation.

The input signal is shown below:

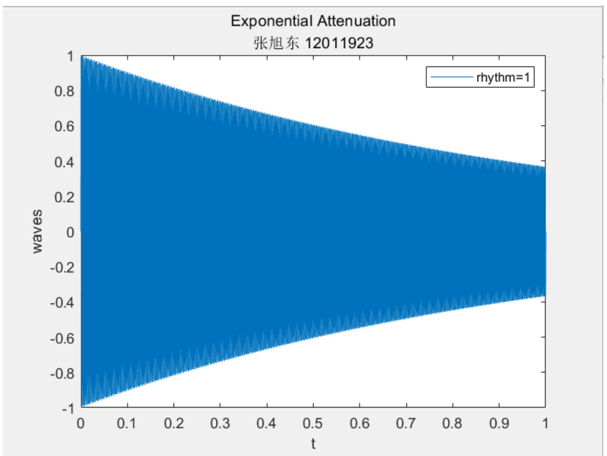
```
fs=8192;
f=440;
T=1/f;
rhythm=1;
t=linspace(0,rhythm,fs*rhythm);
y=sin(2*pi*f*t);
```

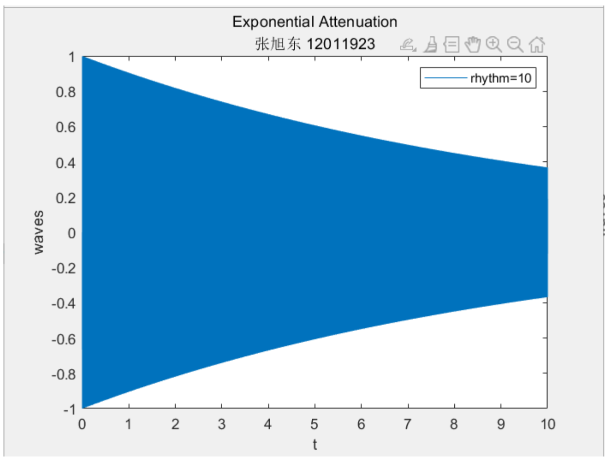
Exponential attenuation

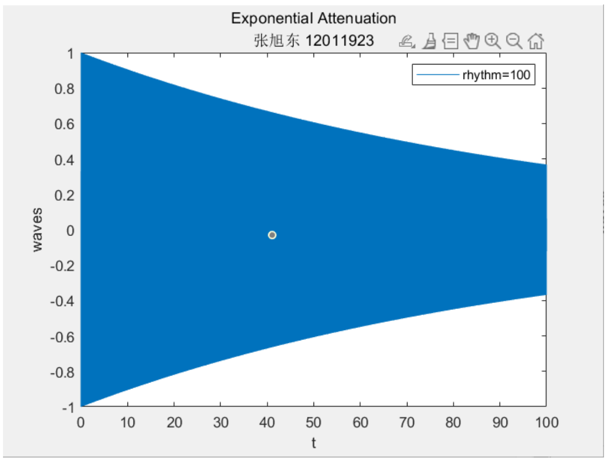
The envelope attenuation function is an exponential function which is :

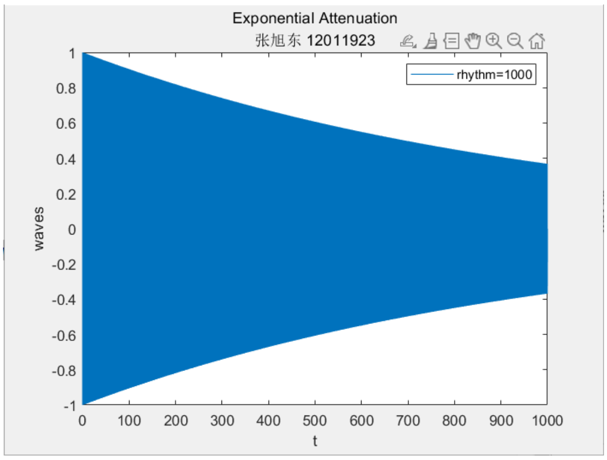
$$waves = y \times e^{-\frac{t}{rhythm}} \tag{1}$$

This function has been introduced in lab manual. Besides, what is found is that the shapes of final waves are very similar no matter rhythm increases. One possible reason is that attenuation is related to rhythm.

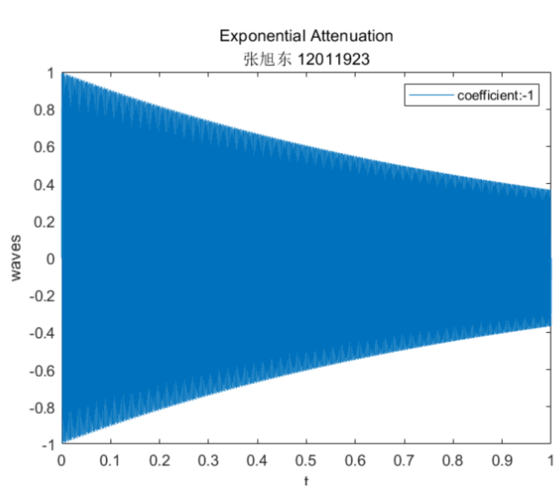


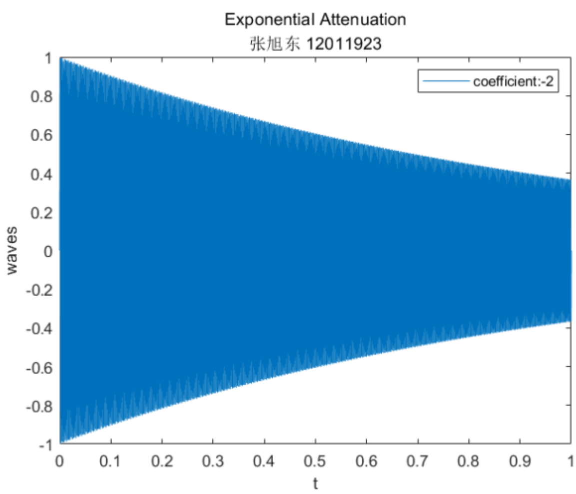


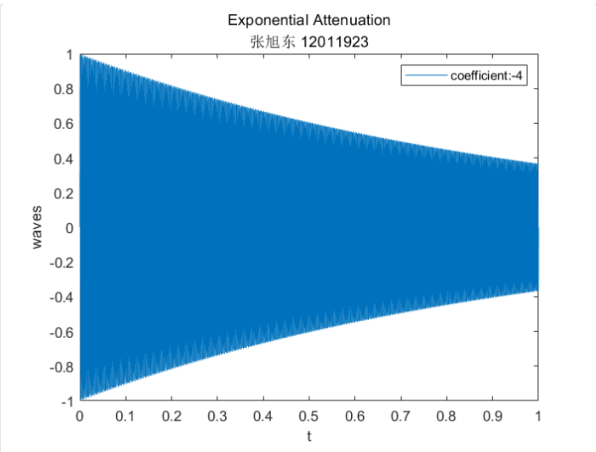


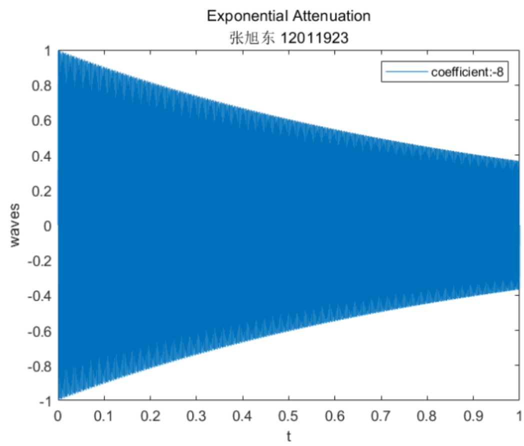


So, the value of rhythm can be set to the value you want. (In the following, the value of rhythm is 1.) And the coefficient of t is changed to $-1, -2, -4, -8, -16$ to compare the result. The waveforms are shown below:

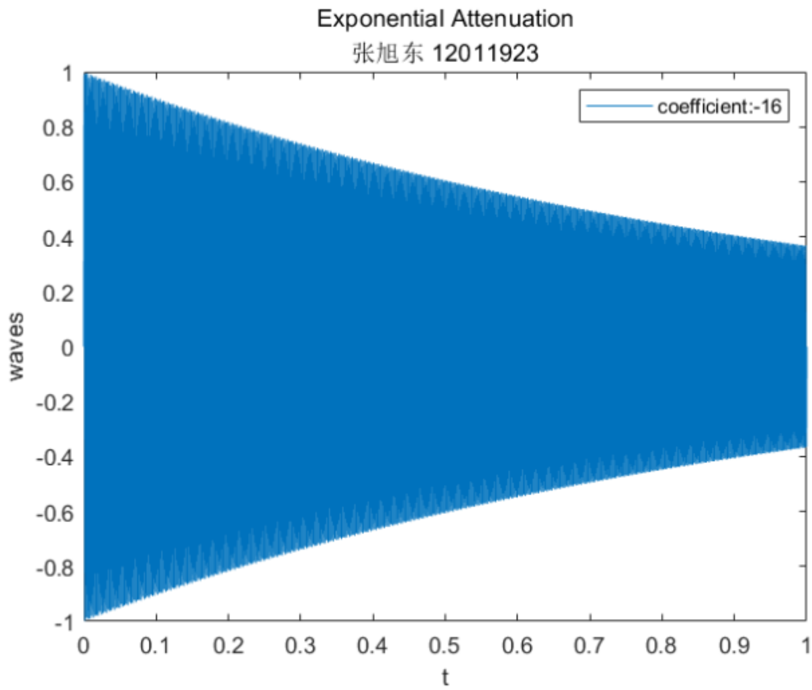








Exponential Attenuation



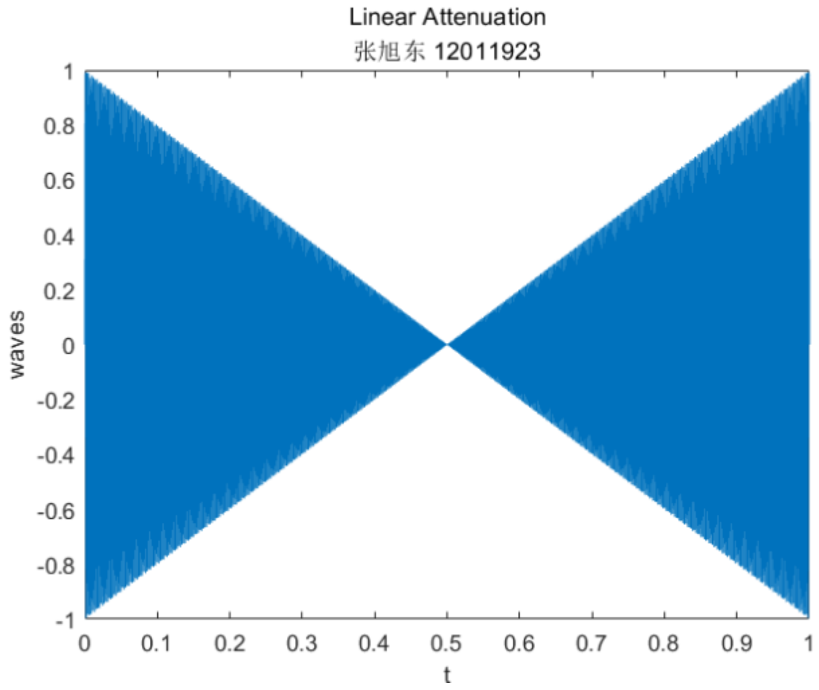
According to the sound, it is obvious that the speed of attenuation gets more and more faster with the coefficient of t increasing. What's more, the sound is like a "de" without almost the trails when the coefficient of t is 16. In my opinion, the second is the most similar to the note played in the real life of the above five. What has to be acknowledged is that the most appropriate coefficient of t may be not 2. There are more space to explore.

Linear attenuation

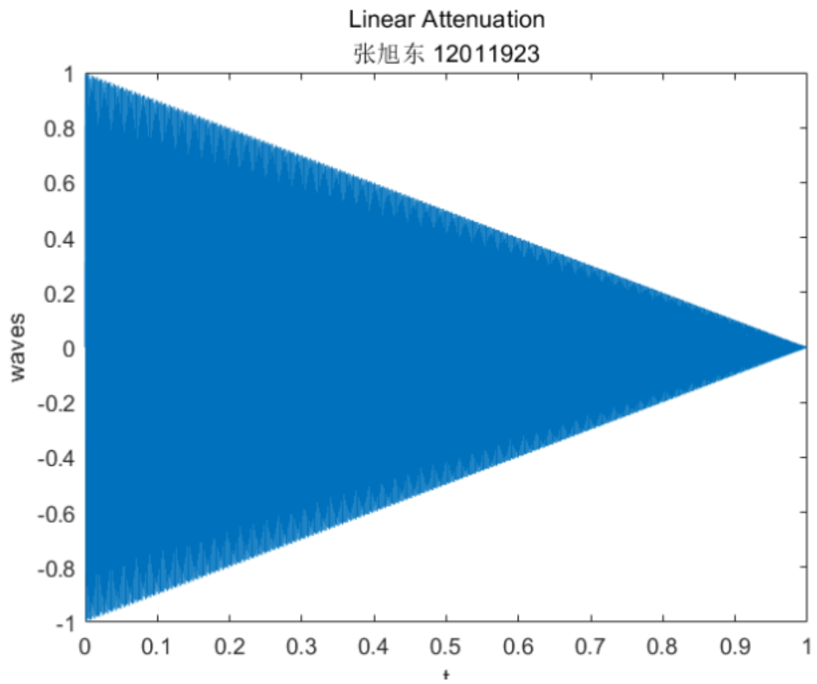
The envelope attenuation function is a linear function which is :

$$waves = y\times(1-\frac{t}{rhythm}) \tag{2}$$

There is a detail needed to be considered. The coefficient of t can't be greater than1.Otherwise, the waveform is totally different from real life. The result is shown below when the coefficient of t is 2:



The waveform is shown below when the coefficient of t is 1 :



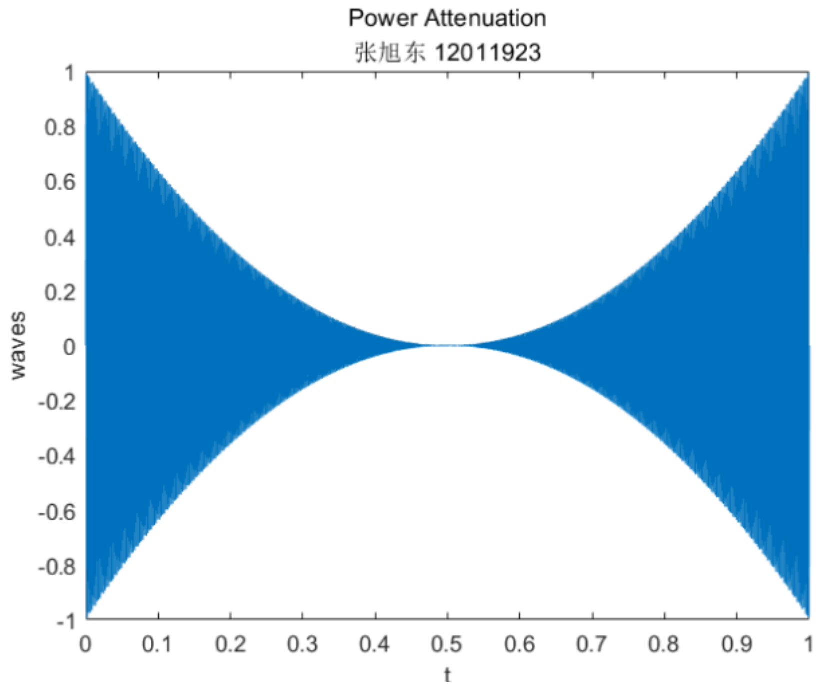
What can be seen is that the amplitude is decaying at a fixed rate, which is contrary to our common sense. Also, the sound is inanimate and lifeless.

Square attenuation

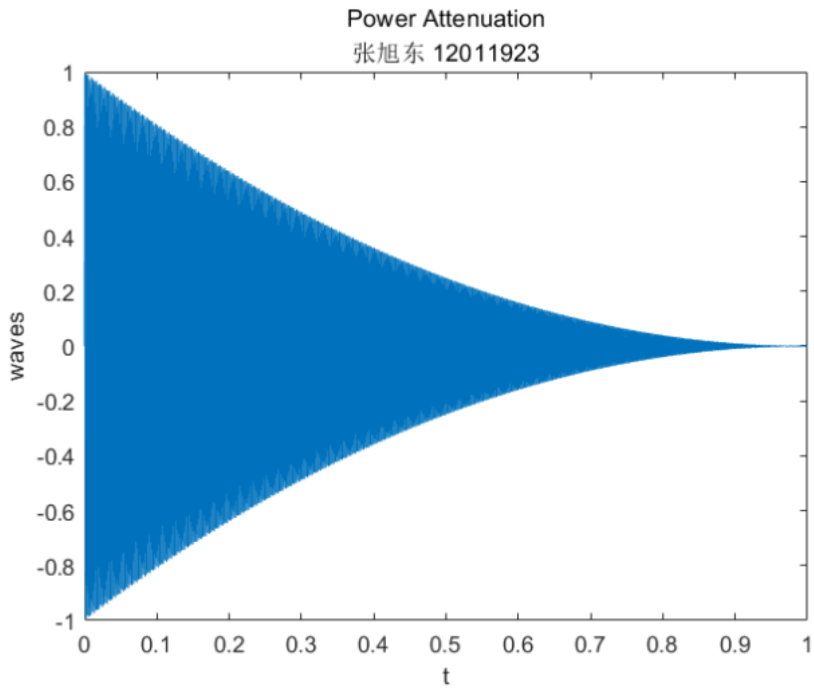
The envelope attenuation function is a linear function which is :

$$waves = y \times (1 - \frac{t}{rhythm})^2 \tag{3}$$

Same as linear attenuation, The coefficient of t can't be greater than1.Otherwise, the waveform is totally different from real life. The result is shown below when the coefficient of t is 2:



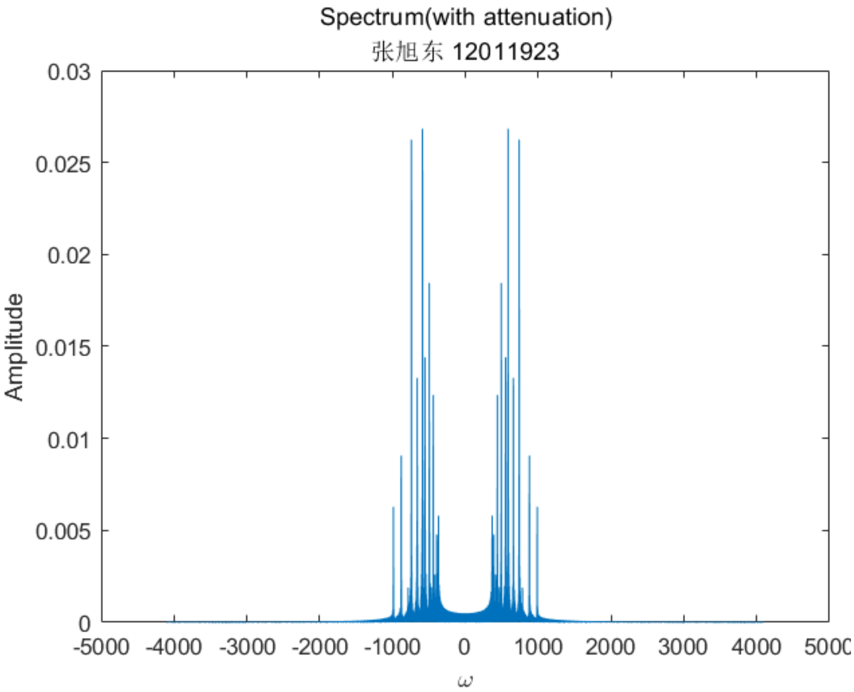
The waveform is shown below when the coefficient of t is 1 :



It is pleasing that it is a little similar to the exponential attenuation. So, guess: the similarity of the two attenuation would increase with appropriate coefficient of t. One major different of the two attenuation is that square attenuation rate is less than the exponential attenuation for square grows slower than the exponential.

Then exponential attenuation function is used to simulate the generation of $Castle\ in\ the\ sky$. Compared to the sound which doesn't contain envelope attenuation, what is obvious is that the sound is smoother, exactly at the higher frequency. That means it sounds more supple, which indicates exponential attenuation function has a good effect in simulating music generation in the real life.

The spectrum of $Castle\ in\ the\ Sky$ with envelope attenuation is below.



From what has been discussed above, in my opinion, the best attenuation function of the above three is exponential attenuation. What's more, as discussed in exponential attenuation, the coefficient of t has an effect on authenticity of simulation. There is a guess that the most appropriate coefficient is between in the interval "[2 4]" according to the performance of different coefficient of t.

5. Exercise 5 , harmonic wave

In real life, when playing instruments, in addition to the fundamental frequency, there are also varying numbers of standing waves due to the sound principle of music instruments. According to the principle of standing wave, the length of the string vibration must be an integer multiple of half wavelength, that means the frequency of the sound consists of the fundamental frequency and the integer multiple harmonic frequency of the fundamental frequency. The main energy is concentrated in the fundamental frequency. Different instruments have different proportions of harmonic energy, which generates waves with entirely different timbre. According to survey, what is found is that most of the existing research is about the digital imitation of piano sounds. What's more, the amplitude of harmonic waves of different notes is different. The `figure1` is the amplitude of harmonic waves at a simplified version while the `figure2` is the amplitude of harmonic waves at a detailed version. The simplified version and *do* notes of detailed version is chosen to do the following simulation.(which to choose is up to you.) Besides, assume the amplitude of the 7th harmonic wave is 0.

a^1	基波	二次谐波	三次谐波	四次谐波	五次谐波	六次谐波	…
频率/Hz	440.1	880.8	1320.6	1760.9	2201.2	2640.7	…
幅值	0.1262	0.0152	0.0062	0.0044	0.0059	0.0043	…

figure1

	基频		二次		三次		四次		五次		六次		七次	
	频率	幅值	频率	幅值	频率	幅值	频率	幅值	频率	幅值	频率	幅值	频率	幅值
do	261.9	0.044	523.6	0.081	786.1	0.017	1048	0.009	1313	0.006	1570	0.007	1832	0.013
re	293.8	0.077	587.2	0.043	882.3	0.011	1177	0.022	1467	0.019	1761	0.004	2055	0.009
mi	329.8	0.173	659.9	0.051	989.9	0.016	1318	0.005	1647	0.008	1977	0.007	2307	0.003
fa	349.3	0.134	699.3	0.035	1050	0.007	1396	0.007	1745	0.009	2095	0.005	2444	0.006
so	392.4	0.083	784.4	0.035	1179	0.006	1569	0.011	1646	0.009	1975	0.002	2746	0.004
la	440.2	0.127	880.9	0.013	1320	0.007	1760	0.005	2201	0.006	2640	0.005	3080	0.002
si	494.0	0.113	988.6	0.014	1482	0.006	1976	0.005	2469	0.006	2964	0.004	3458	0.002

figure2

This part is divided into two parts: the first part is the comparison of different amplitudes of harmonic waves about single tone and the second part is the comparison of different amplitude of harmonic waves about $Castle\ in\ the\ sky$. What's more, exponential attenuation function is added to both of them in order to simulate music more really, whose expression is:

$$waves = y \times e^{-\frac{2t}{rhythm}} \tag{4}$$

Single Tone

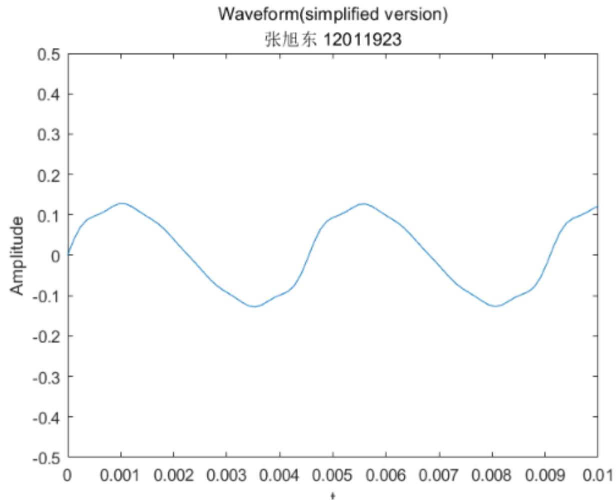
Two harmonic energy ratios is shown below:

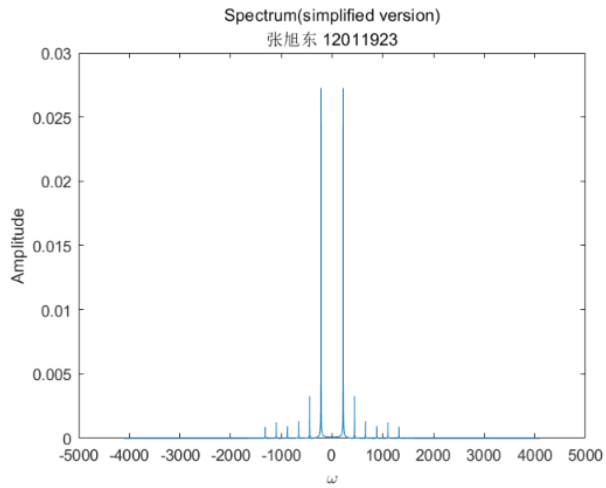
```
k1=[0.1262 0.0152 0.0062 0.0044 0.0059 0.0043 0];
k2=[0.044 0.081 0.017 0.009 0.006 0.007 0.013];
```

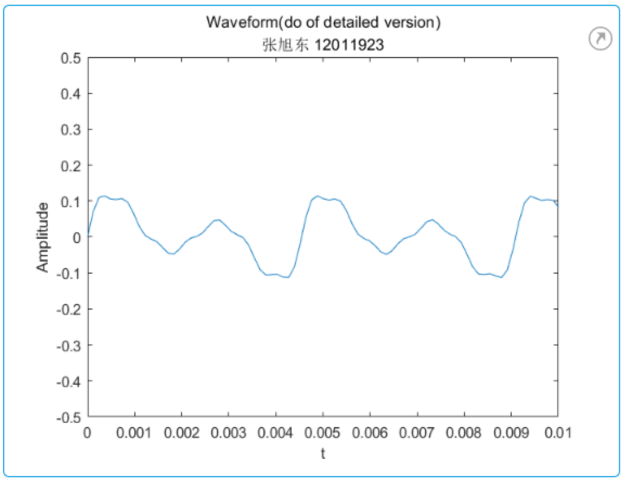
The MATLAB code is shown below:

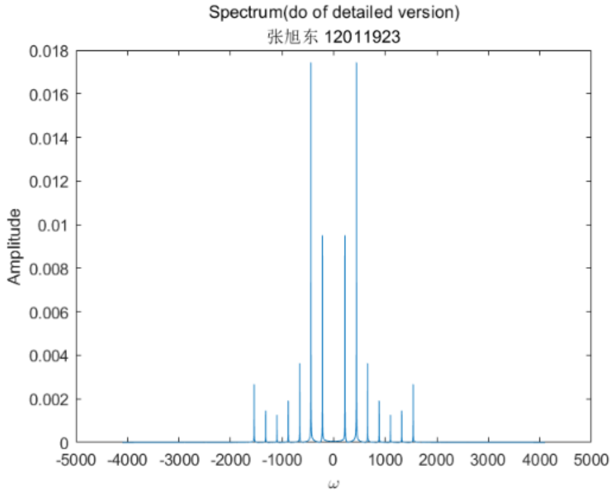
```
fs=8192;%采样率
fc=220;%基频
rhythm=1;%时宽
n=fs*rhythm;%采样点个数
t=linspace(0,rhythm,fs*rhythm);
f=linspace(-fs/2,fs/2-1,n);
k1=[0.1262 0.0152 0.0062 0.0044 0.0059 0.0043 0];
k2=[0.044 0.081 0.017 0.009 0.006 0.007 0.013];
y1=0;
y2=0;
for i =1:length(k1)
    y1=y1+k1(i)*sin(2*pi*i*fc*t).*exp(-2*t/rhythm);
    y2=y2+k2(i)*sin(2*pi*i*fc*t).*exp(-2*t/rhythm);
end
plot(t,y1);
xlabel('t');
ylabel('Amplitude')
title(['Waveform(simplified version)' newline '张旭东 12011923'])
axis([0 0.01 -0.5 0.5]);
plot(f,abs(fftshift(fft(y1./n))));
xlabel('\omega');
ylabel('Amplitude')
title(['Spectrum(simplified version)' newline '张旭东 12011923'])
plot(t,y2);
xlabel('t');
ylabel('Amplitude')
title(['Waveform(do of detailed version)' newline '张旭东
12011923'])
axis([0 0.01 -1 1]);
plot(f,abs(fftshift(fft(y2./n))));
xlabel('\omega');
ylabel('Amplitude')
title(['Spectrum(do of detailed version)' newline '张旭东
12011923'])

sound(y1,fs);
pause(1.5);
sound(y2,fs);
```

There are some details needed to be considered. Firstly, when do `DFT`, the target function needs to be divided by the total number of samples. Secondly, according to the `Nyquist sampling theorem`, sample frequency f_s need to be greater than two times of signal frequency f_c.





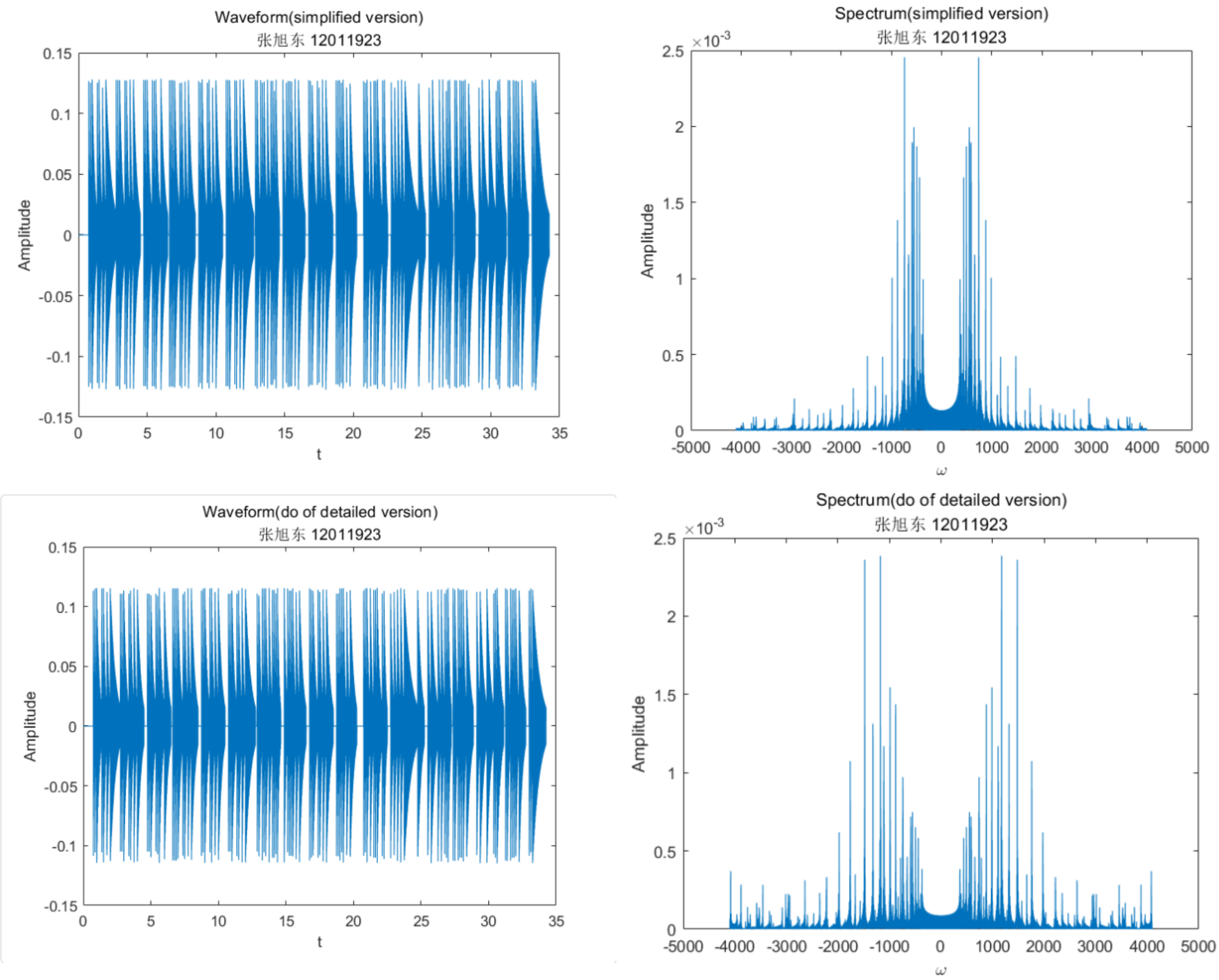




The first thing to do is to compare their waveform and it is obvious that the waveform of simplified version is smoother than the waveform of detailed version. The second thing to do is to compare their spectrum in frequency domain. It is concluded that the larger the amplitude of the i_{th} harmonic wave in time domain, the larger energy of the i_{th} harmonic wave in frequency domain, which is the same as the theory. At last, timbre difference between is analyzed. For simplified version, the started frequency is low and gives a soft feeling. For *do* of detailed version, the started frequency is high and gives a vigilant feeling. In my opinion, I prefer the first one.

Castle in the sky

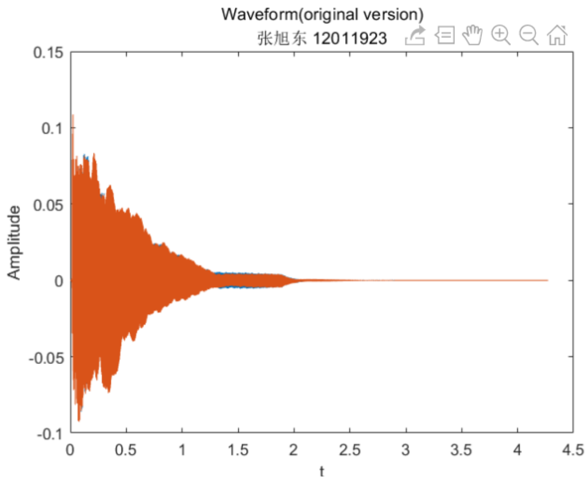
The two harmonic energy ratios is the same as that in single tone. The result is shown below.

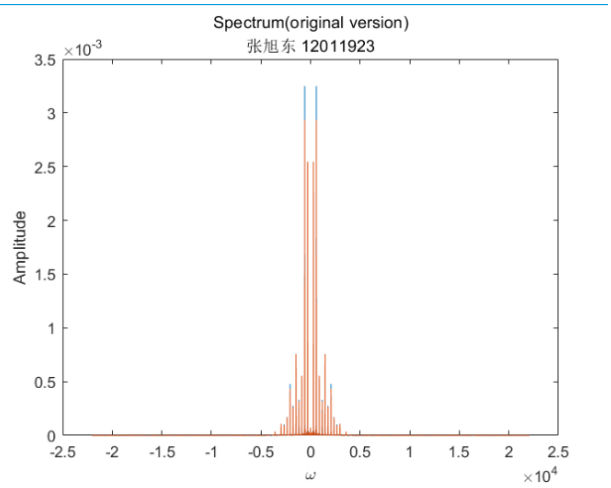


Analyzing from time domain, nothing can be gotten. However, there are lots of information in frequency domain. Firstly, most of energy is concentrated in low frequencies for simplified version and most of energy is concentrated in intermediate frequencies for *do* of detailed version. Secondly, energy distributed in high frequency for *do* of detailed version is more than that for simplified version. After that, timbre difference between is analyzed. For simplified version, the sound gives a light and ethereal feeling. For *do* of detailed version, the sound gives a stagnant and slightly hoarse feeling. Absolutely, I prefer the first one.

6.Simutation For Piano

Before simulating the piano, what is needed to do is download the standard $1 = D$ tone and analyze the waveform and spectrum of real piano. The figures are follows.

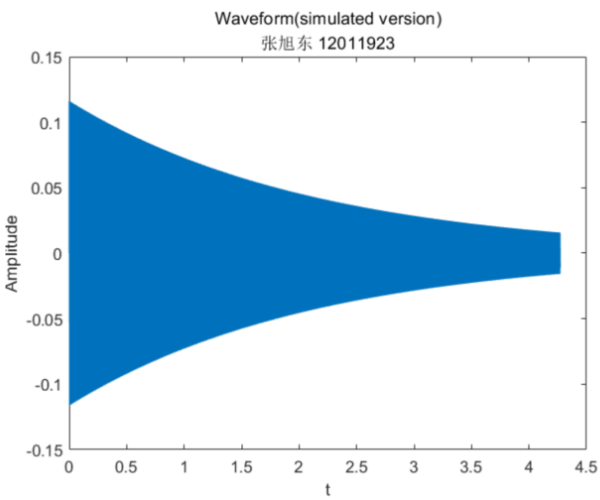


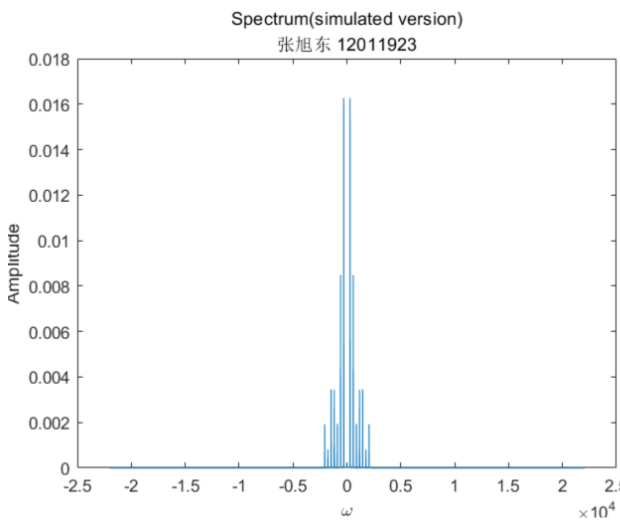


Then, the amplitude of harmonic waves for piano is found according to the reference. The amplitude of harmonic waves is:

```
k1=[0.077 0.043 0.011 0.022 0.019 0.004 0.009];
```

After that, exponential attenuation function, $waves = y \times e^{-\frac{2t}{rhythm}}$ is used to simulate the sound of piano. The waveform and spectrum of simulation are follows.





Starting analysis with waveform in time domain. It is obvious that the speed of attenuation of the waveform of real piano is faster than that of waveform of simulated sound. Also, the amplitude variation law of the former is relatively disordered relative to that of the latter. Then,analyze frequency domain. Most of the energy is concentrated in roughly the same range of frequencies. However, the bandwidth of the former is larger than that of the latter and the amplitude of the former is an order of magnitude smaller than that of the latter.

Conclusion

In this lab, how to use $Matlab$ simulation for the tone and digital music is learned. Firstly, principle of Number Musical Notation is understood. Secondly, envelope attenuation is needed to considered when simulating music generation because the vibration will experience attenuation and will not continue to vibrate at a fixed amplitude when playing instruments in real life. Last but not least, in addition to the fundamental frequency, there are also varying numbers of standing waves due to the sound principle of music instruments, which is also needed to considered.

Reference

[1] 曹莎莎. 一种钢琴乐音仿真模型的研究[D].合肥工业大学,2017.

[2] 刘超.基于频谱包络的钢琴乐音仿真模型构建[D].咸阳师范学院，2021.