AMATH 482
Homework 2 Report
Fangzheng Sun
1239942

Abstract

This report has two parts. The first part is to apply different filters to the 9-second piece of Handel's classic music and the second part is to apply filters to two pieces of *Mary had a little lamb* player by piano and recorder.

Introduction

Given a piece of music and its frequency information, I am going to apply different filtering methods, especially Gabor filtering, to the frequency and explore the differences of the spectrogram using different coefficients or different Gabor windows.

Theoretical background

The following theories are included in this project:

1. Fast Fourier Transform and Inverse Fast Fourier Transform. FFT converts a signal from its original domain (time/space) to a representation in the frequency domain. And iFFT does the inverse job.
2. Gaussian Filter. Given the center frequency, Gaussian filter is used in the frequency domain to help denoise the given frequencies by multiplying with the data in the frequency domain and results in a denoised frequency domain. The Gaussian filter equation is:

$$F(k) = e^{-\tau(k-k_0)^2}$$

Where τ measures the bandwidth of the filter and k = k0 is desired signal.

3. Gabor Filter. The simplest Gabor window to implement is a Gaussian time-filter centered at some time τ with width a. The equation is:

$$g(t) = e^{-a(t-b)^2}$$

In this time-frequency analysis, I create a "t-slide" to make sure the filter go over the whole piece of music.

4. Spectrogram. A spectrogram is a visual representation of the spectrum of frequencies in a sound or other signal as they vary with time or some other variable. In the analysis, spectrogram is produced after the filtering.

5. Mexican hat wavelet and step-function. These are two different Gabor windows.

   Mexican hat wavelet:

$$g(t) = (1-(t-t_0)^2)e^{-(t-t_0)^2/2}$$

   Step-function:

$$g(t) = \begin{cases} 1, & 0 \leq (t-t_0) < 1 \\ 0, & otherwise \end{cases}$$

The following sections are dividing by part 1 and part 2:

Part 1:

Algorithm Implementation and Development

1. Through use of the Gabor filtering we used in class, produce spectrograms of the piece of work. For this question, I firstly define time-frequency domain for the given data. Since the length is an odd number, I throw the last frequency away. Through the t-slide process, I multiply filter to the frequency and fft the filtered data. Meanwhile, the spectrogram is produced. The window width is a=1, t-slide is 1:0.1:9.

2. Explore the window width of the Gabor transform and how it effects the spectrogram. I take

a=0.01 and a=100.

3. Explore the spectrogram and the idea of oversampling versus potential undersampling. Here I take t-slide 1:0.02:9 and 1:3:9, two pretty extreme cases.

4. Use different Gabor windows. Perhaps you can start with the Gaussian window, and look to see how the results are effected with the Mexican hat wavelet and a step-function (Shannon) window. I change the filter equation to Mexican hat wavelet and step-function and then produce their spectrograms.

Computational Results

Part 1

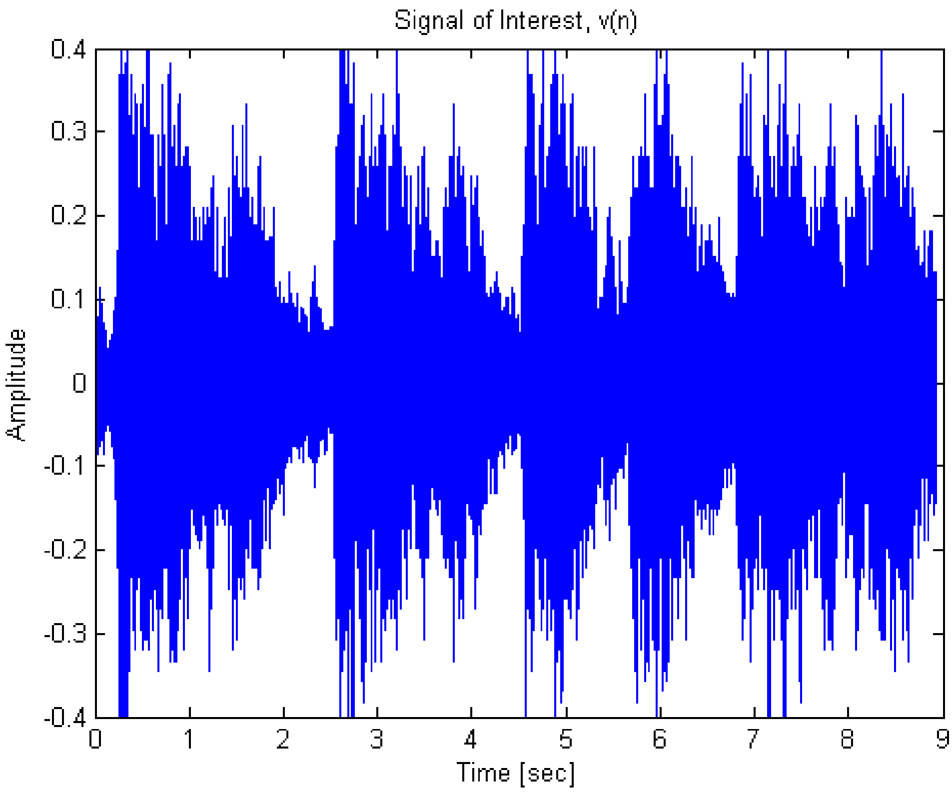


Figure 1 Original Frequency

1. From the graph of filtered data and the spectrogram, we can clear see the filtering effect on the data.

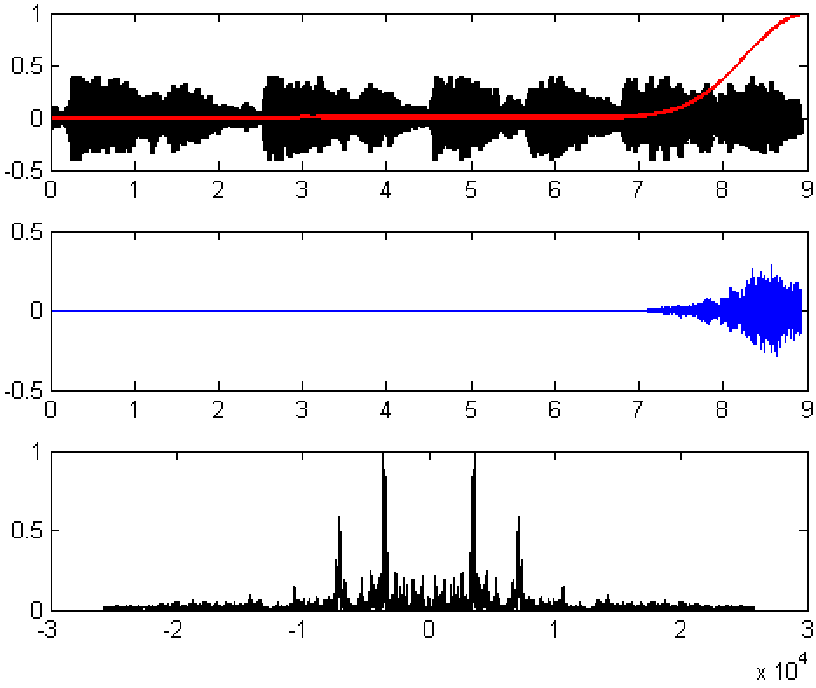


Figure 2 1. Frequency and Filter. 2. Filtered data. 3. Frequency range

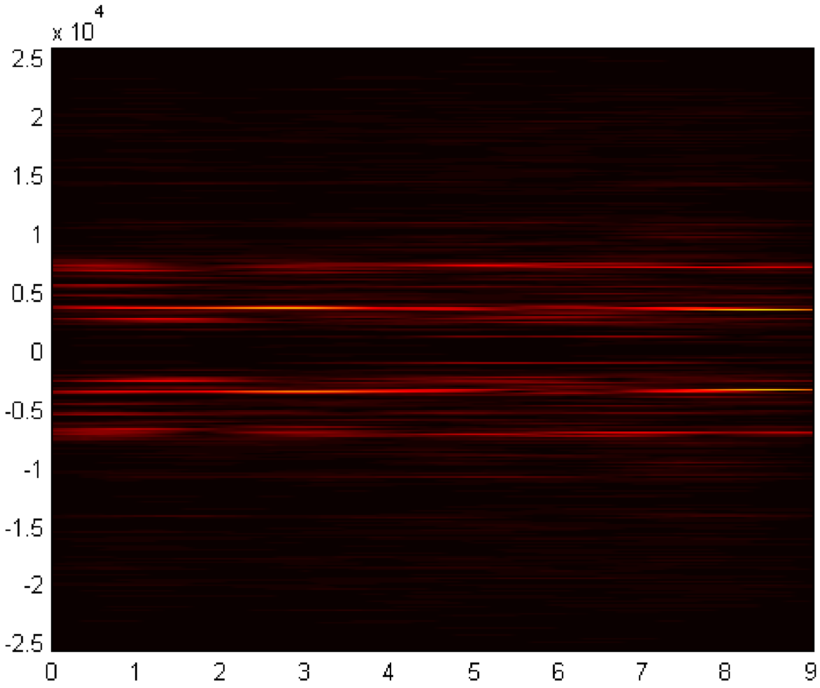
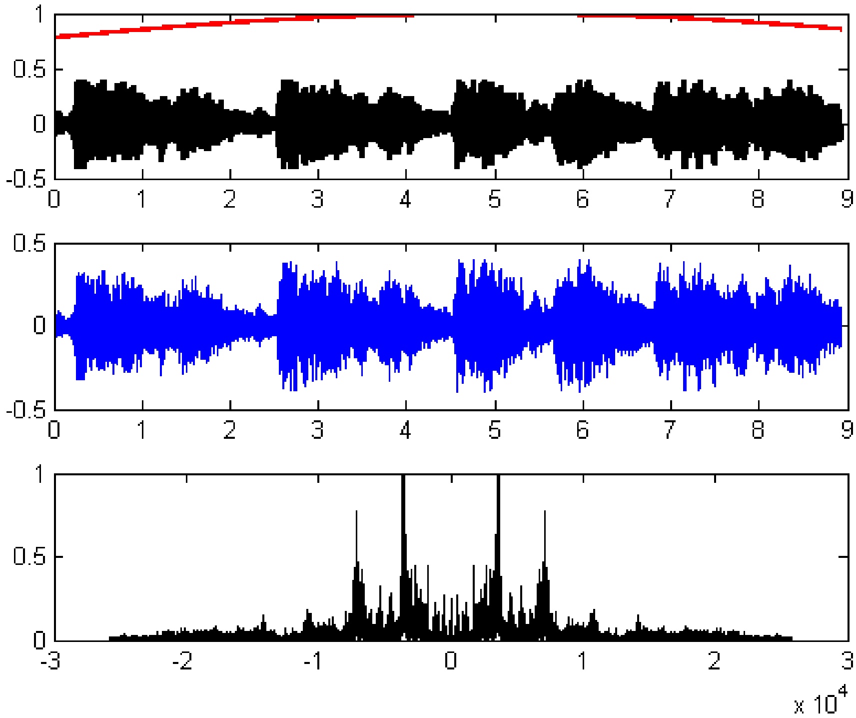
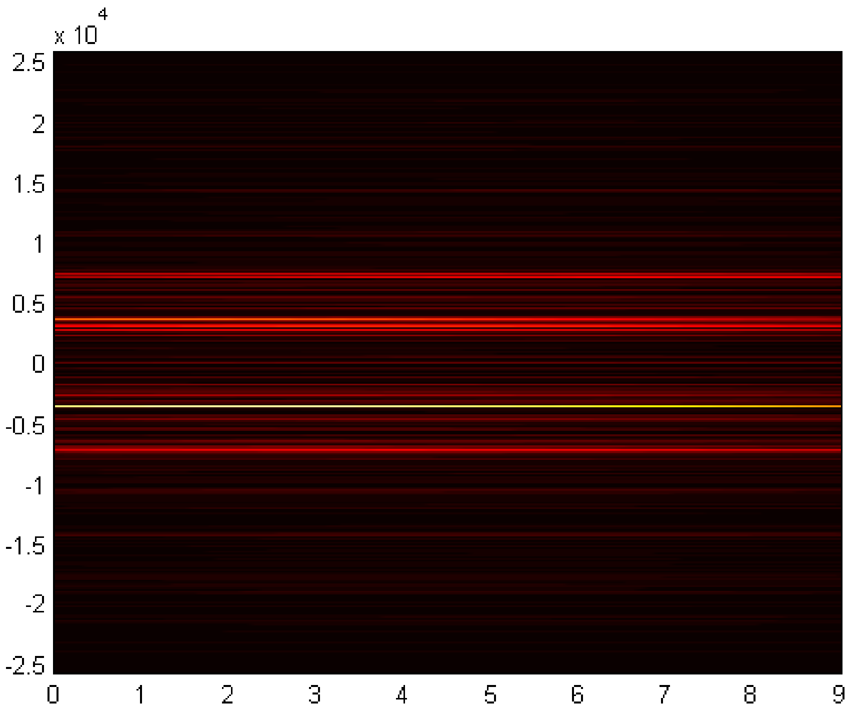


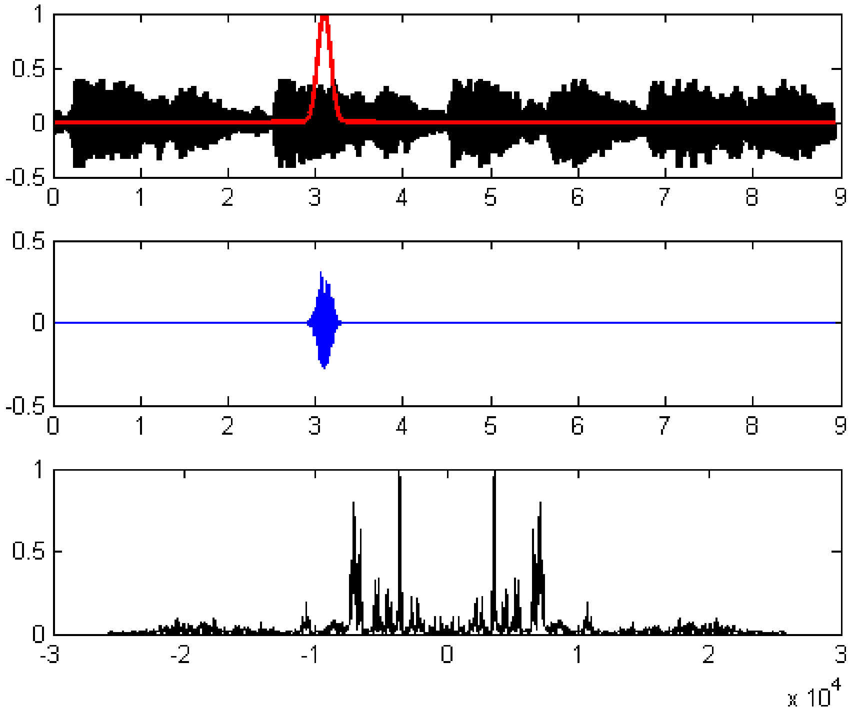
Figure 3 Spectrogram

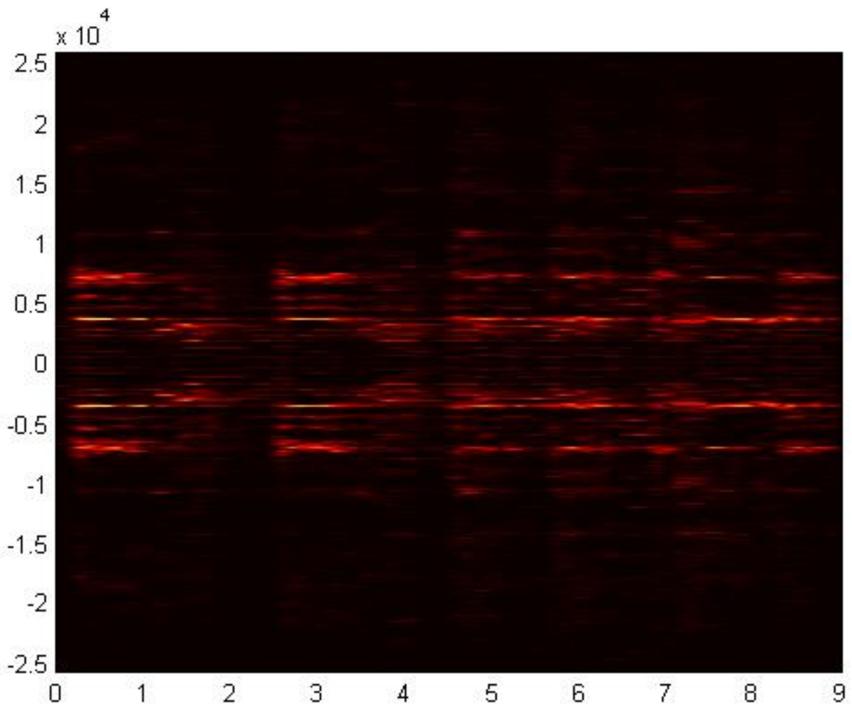
2. Different window width: a=0.01. It is clear that when “a” is too small, the filtering will have little effect on the given data, and in the spectrogram the frequencies are continuous. The window width trades off time and frequency resolution at the expense of each other.



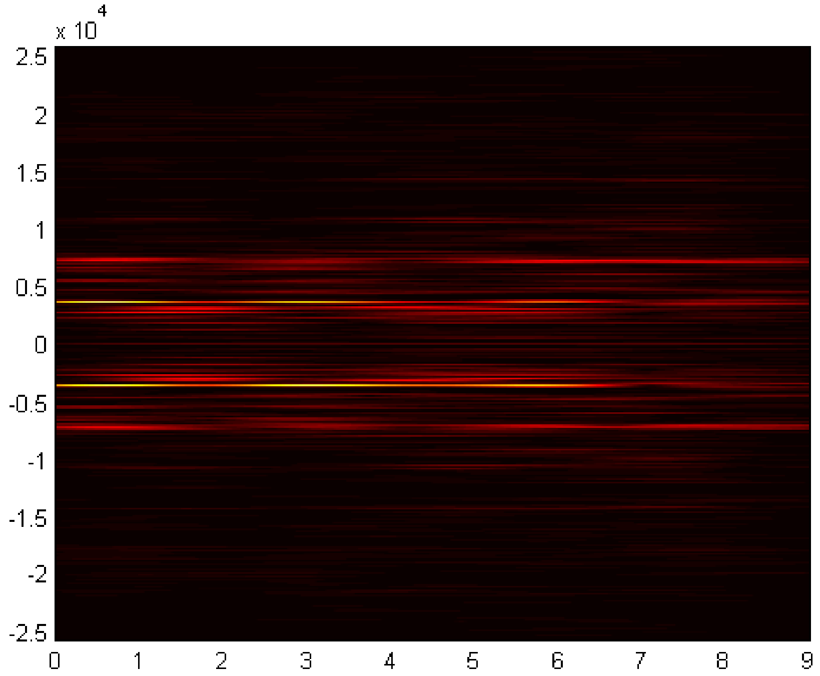


a=100. It is clear that when “a” is large, too much information is missing, causing the spectrogram not complete. The window width trades off time and frequency resolution at the expense of each other.

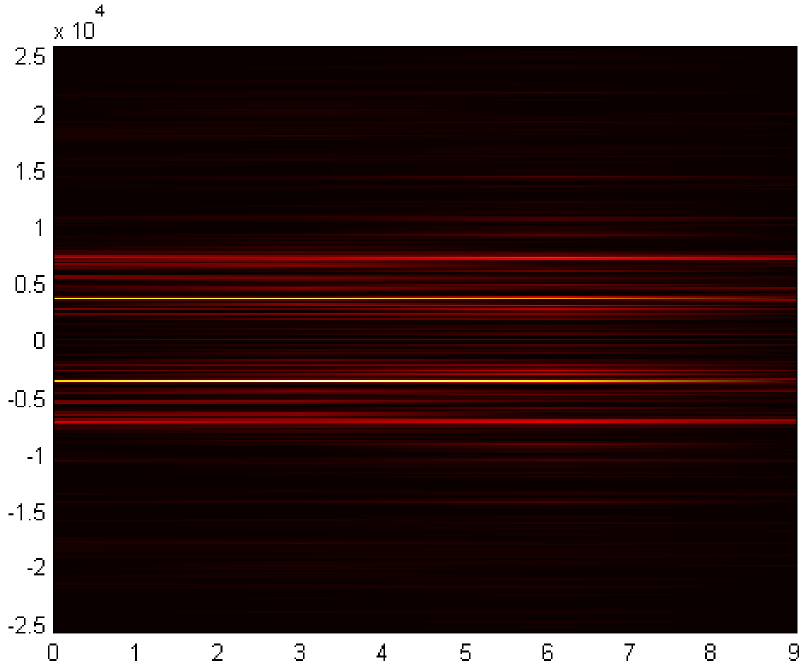




3. oversampling: t-slide 1:0.02:9. Compared with the original spectrogram, oversampling spectrogram is more intense since too much information is overlapping.

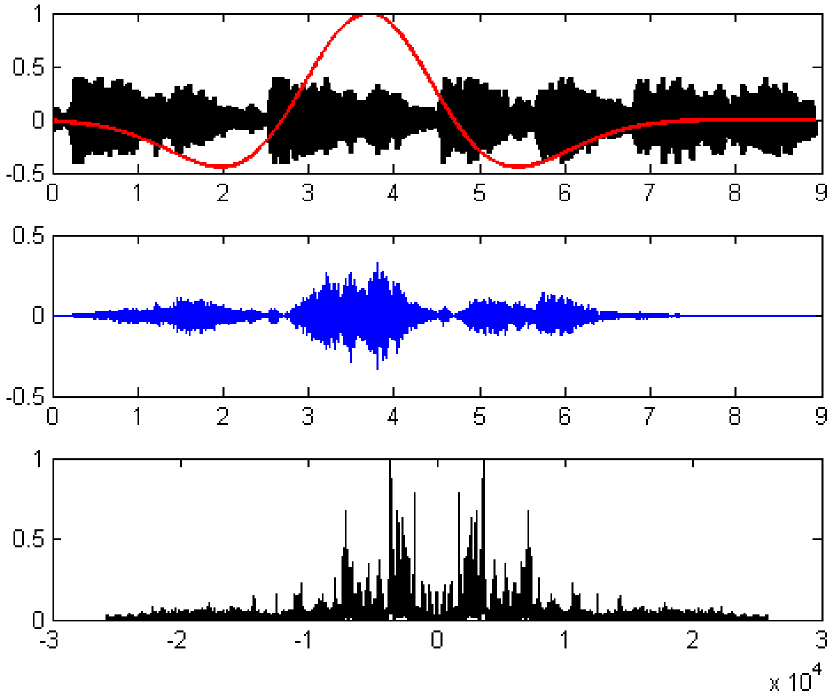


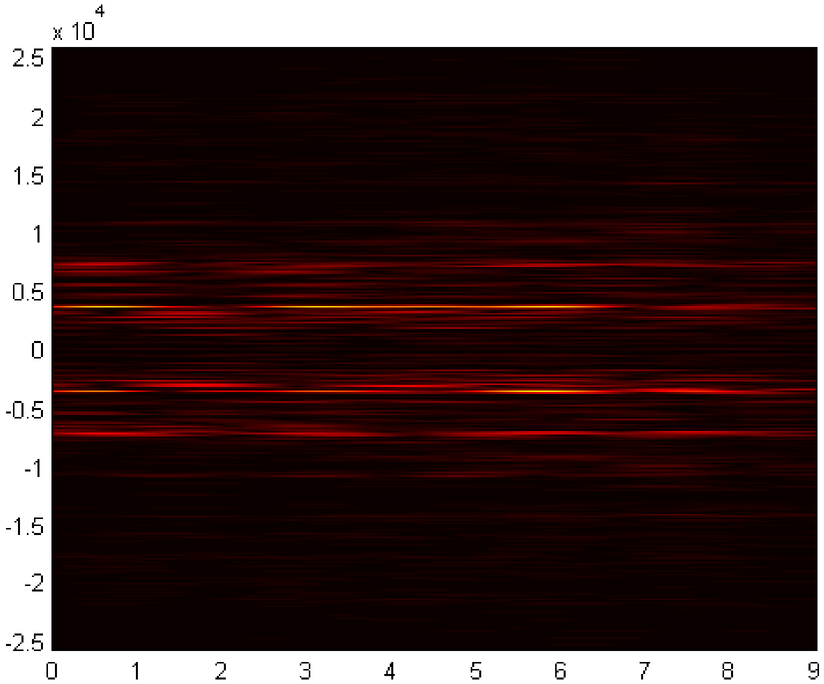
Undersampling: t-slide 1:3:9. This one is continuous. Since the filter moves fast, too many frequencies are uncovered.



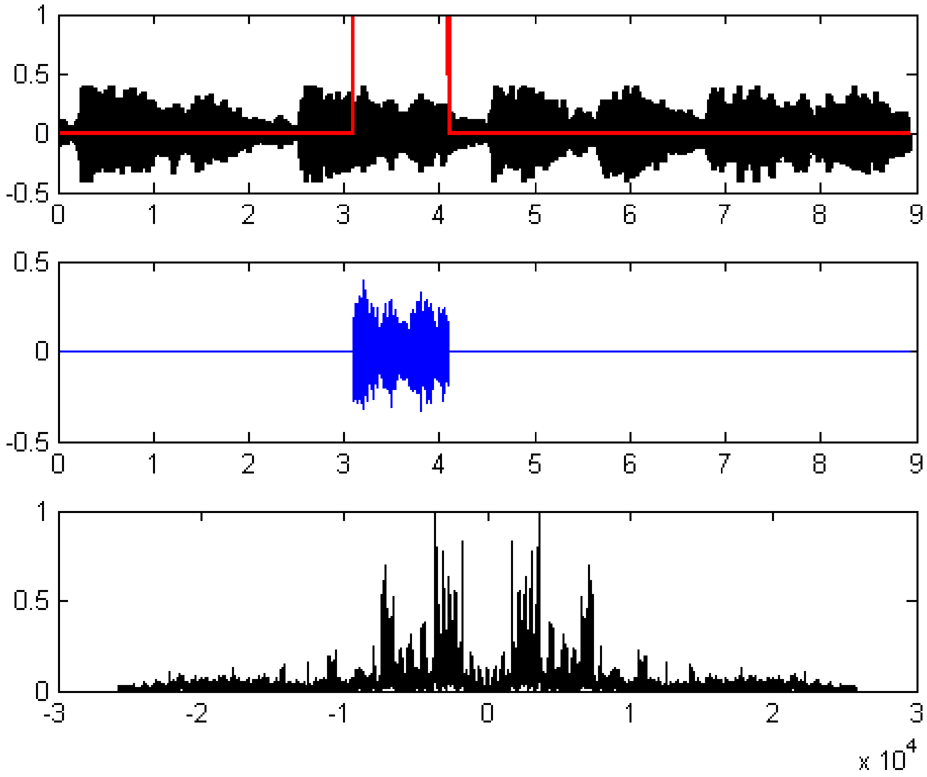
4. Different Gabor window.

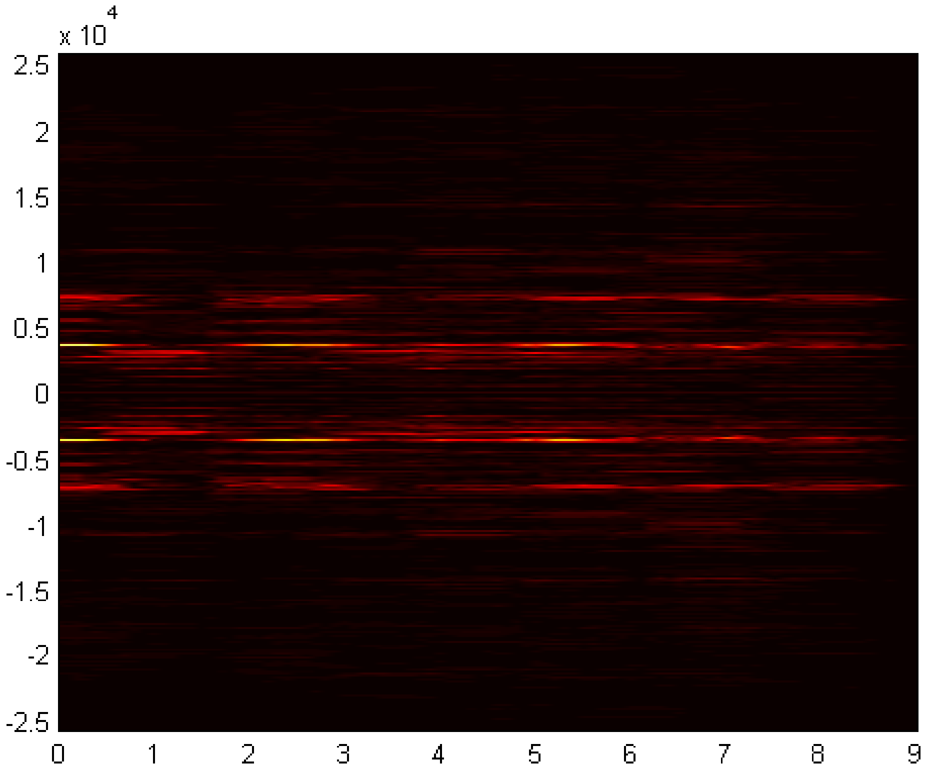
Mexican $g(t) = (1-(t-t_0)^2)e^{-(t-t_0)^2/2}$ The spectrogram is a little different from the original one. The frequencies are more distributed more sparsely since there are two smaller wave peaks on the other side of the filter curve.





Step function $g(t) = \begin{cases} 1, & 0 \le (t - t_0) < 1 \\ 0, & otherwise \end{cases}$ In the spectrogram, the lines are less continuous.





Summary and conclusions

The results are represented mostly by different spectrograms in the last section. In general, by applying different coefficients or using different shapes of filters, we will get different spectrograms.

Appendix A: MATLAB functions

linspace – Generate linearly spaced vector

fft – Fast Fourier Transform

fftshift – Shift Zero frequency component to center of spectrum

Appendix B: MATLAB code

Original code:

```
clear all; close all; clc

load handel;
v = y'/2;
L=length(v)/Fs;
n=length(v)-1; %n becomes and even number
t2=linspace(0,L,n+1); t=t2(1:n);
v=v(1:n);%throwing away the last film
k=(2*pi/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
figure(1),plot(t,v);%original frequency picture
xlabel('Time [sec]'); ylabel('Amplitude');
title('Signal of Interest, v(n)');
%% filter
figure(2)
spec=[];%start creating spectrogram
tslide=0:0.1:9;%filter different area by time
for j=1:length(tslide)
    g=exp(-1*(t-tslide(j)).^2);%gabor filtering
    subplot(3,1,1)
    plot(t,v,'k',t,g,'r','Linewidth',[2]);%frequency and filter
```

```
    vf=g.*v;
    vft=fft(vf);
    spec=[spec; abs(fftshift(vft))];%adding filter to spectrogram
    subplot(3,1,2)
    plot(t,vf);%filtered frequency
    subplot(3,1,3)
    plot(ks,abs(fftshift(vft))/max(abs(vft)),'k');%frequency range
    drawnow
end
%% spectrogram
figure(3)
pcolor(tslide,ks,abs(spec).'), shading interp
colormap(hot);
```

Different Gabor windows:

Mexican:

```
g=(1-(t-tslide(j)).^2).*(exp((-(t-tslide(j)).^2)/2));
```

Step function:

```
g=heaviside(t-tslide(j)) - heaviside(t-tslide(j)-1);
```

Part 2

Algorithm Implementation and Development

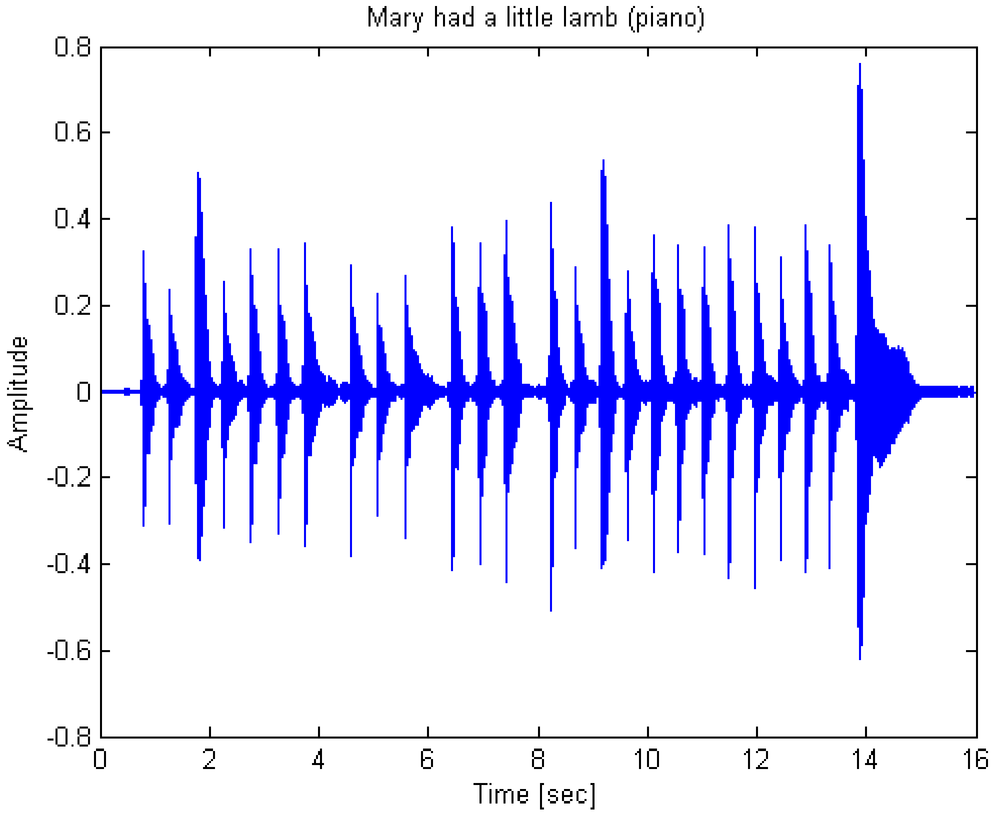
1. Through use of the Gabor filtering we used in class, reproduce the music score for this simple piece. For this part, I do similar job as part 1, applying the filter to the frequencies and producing the spectrogram. By testing different numbers of “a”, I got the clear view of the spectrogram, which allows me to count the frequencies of each one nodes. I try different window width and I choose a=80

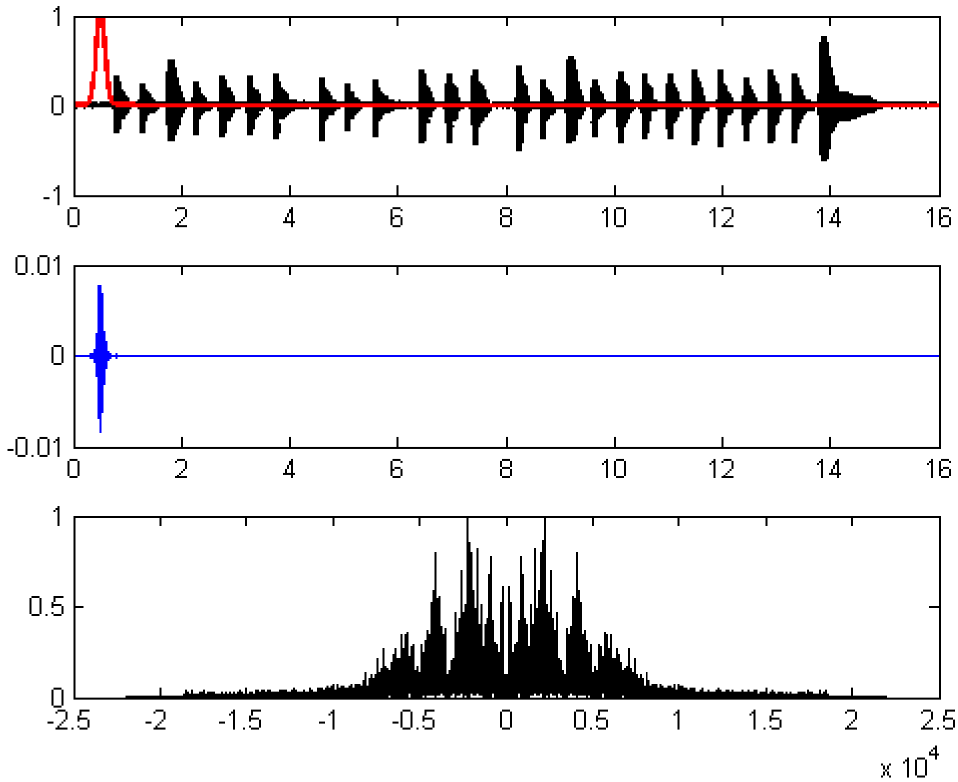
2. What is the difference between a recorder and piano? Can you see the difference in the time-

frequency analysis? The difference between a recorder and piano is there frequencies. They produce different frequencies.

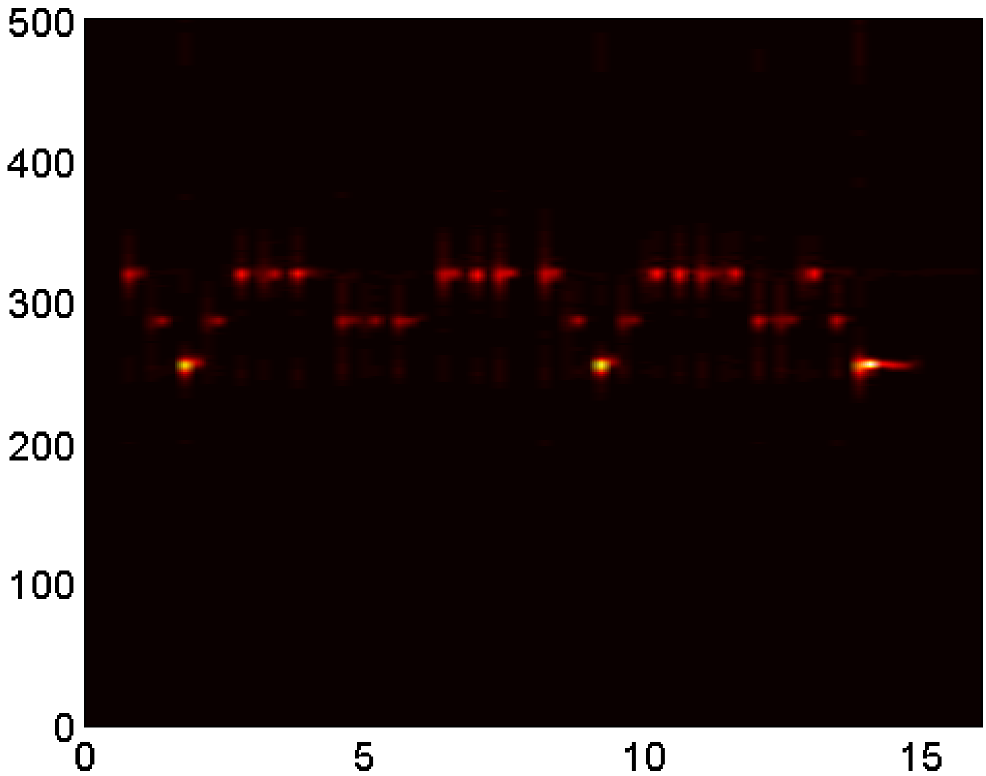
Computational Results

Piano



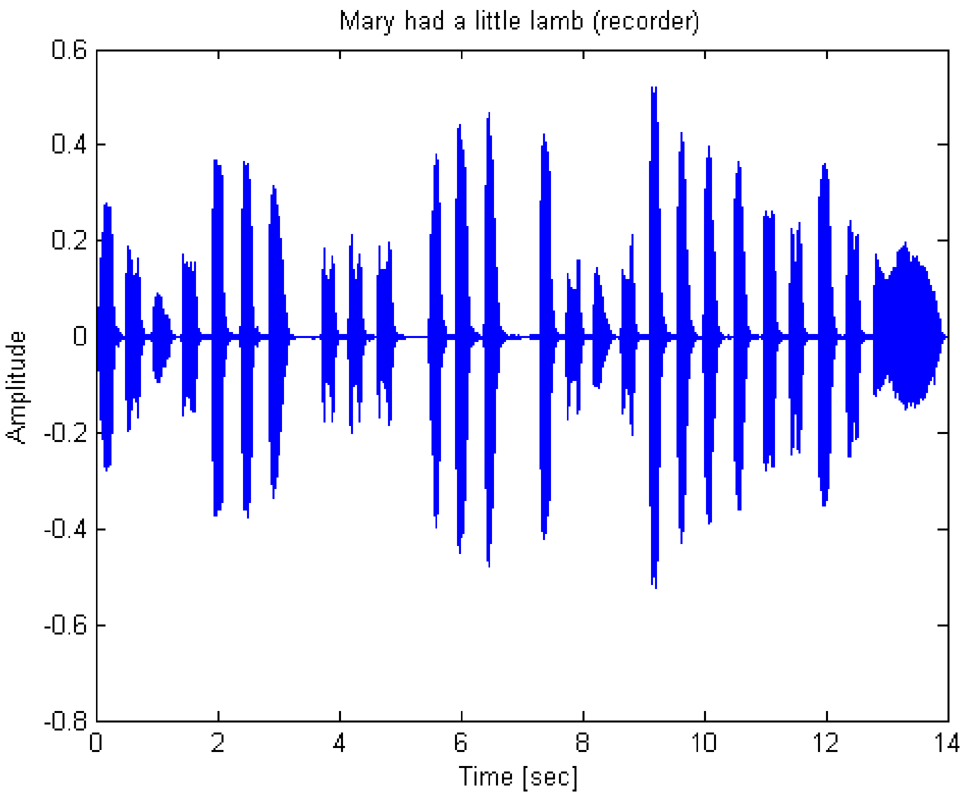


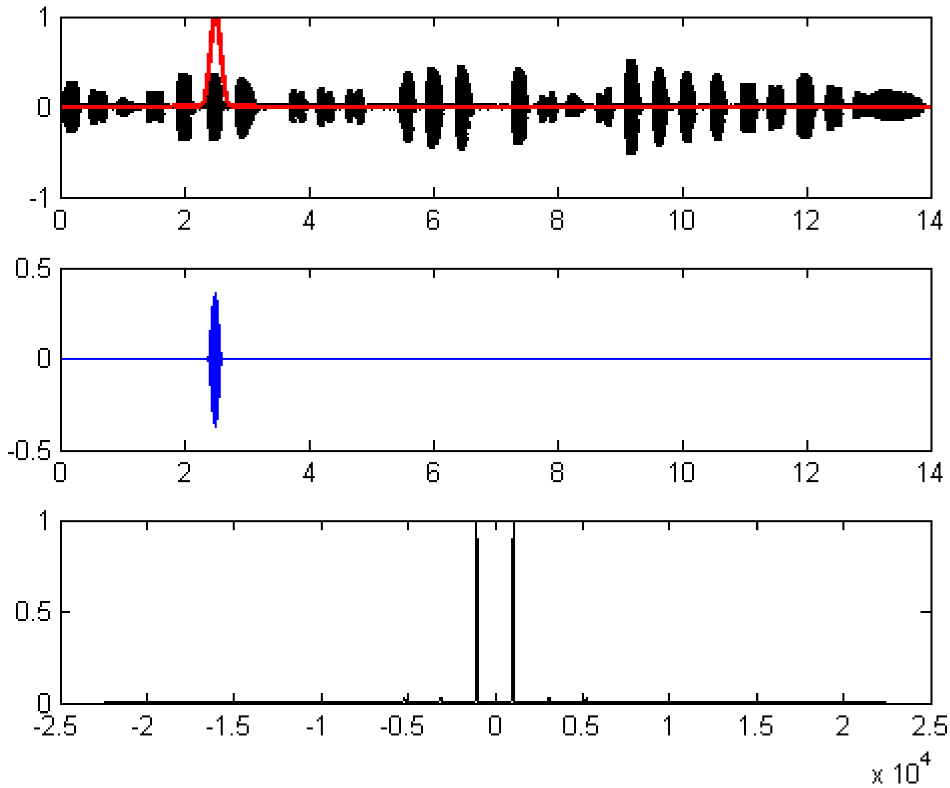
1
0
-1
0
2
4
6
8
10
12
14
16
0.01
0
-0.01
0
2
4
6
8
10
12
14
16
1
0.5
0
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
x 10^4

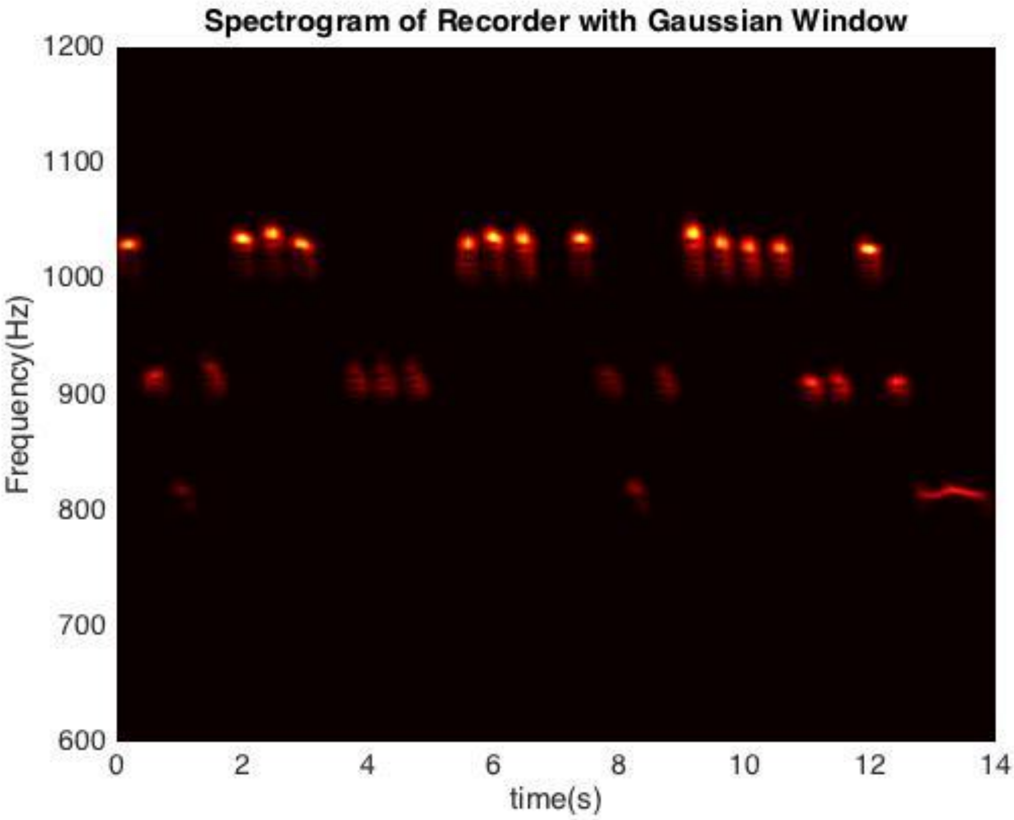


500
400
300
200
100
0
0
5
10
15

Recorder


Mary had a little lamb (recorder)
Amplitude
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
2
4
6
8
10
12
14
Time [sec]


1
0
-1
0.5
-0.5
0
2
4
6
8
10
12
14
-2.5
-2
-1.5
-1
-0.5
0.5
1.5
2.5
x 10^4



Summary and conclusions

The spectrograms of the piano and recorder version of *Mary had a little lamb* are clear enough to demonstrate the result. And the questions are answered in the previous sections.

Piano: E4 D4 C4 D4 E4 E4 E4   D4 D4 D4   E4 E4 E4   E4 D4 C4 D4 E4 E4 E4 C4 D4 D4 E4 D4 D4

Recorder: C6 B5♭ A5♭ B5♭ C6 C6 C6   B5♭ B5♭ B5♭   C6 C6 C6   C6 B5♭ A5♭ B5♭ C6 C6 C6 A5♭ B5♭ B5♭ C6 B5♭ A5♭

Appendix A: MATLAB functions

linspace – Generate linearly spaced vector

fft – Fast Fourier Transform

fftshift – Shift Zero frequency component to center of spectrum

Appendix B: MATLAB code

Piano

```
clear all; close all; clc

tr_piano=16; % record time in seconds
y=wavread('music1'); Fs=length(y)/tr_piano;
y=y';
L=length(y)/Fs;
n=length(y);
t2=(1:length(y))/Fs; t=t2(1:n);
k=(1/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
figure(1),plot(t,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (piano)'); drawnow
%% filter
figure(2)
spec=[];
tslide=0:0.2:t(end);
for j=1:length(tslide)
    g=exp(-80*(t-tslide(j)).^2);
    subplot(3,1,1)
    plot(t,y,'k',t,g,'r','Linewidth',[2]);
    yf=g.*y;
    yft=fft(yf);
    spec=[spec; abs(fftshift(yft))];
    subplot(3,1,2)
    plot(t,yf);
    subplot(3,1,3)
    plot(ks,abs(fftshift(yft))/max(abs(yft)),'k');
    drawnow
end
%% spectrogram
figure(3)
pcolor(tslide,ks,abs(spec).'), shading interp
set (gca,'Ylim',[0,500],'fontsize',[14]);
colormap(hot);
```

Recorder

```
clear all; close all; clc

tr_rec=14; % record time in seconds
y=wavread('music2'); Fs=length(y)/tr_rec;
y=y';
L=length(y)/Fs;
n=length(y);
t2=(1:length(y))/Fs; t=t2(1:n);
k=(1/L)*[0:n/2-1 -n/2:-1]; ks=fftshift(k);
figure(1),plot(t,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (recorder)'); drawnow
%% filter
figure(2)
spec=[];
tslide=0:0.1:t(end);
for j=1:length(tslide)
    g=exp(-80*(t-tslide(j)).^2);
    subplot(3,1,1)
    plot(t,y,'k',t,g,'r','Linewidth',[2]);
    yf=g.*y;
    yft=fft(yf);
    spec=[spec; abs(fftshift(yft))];
    subplot(3,1,2)
    plot(t,yf);
    subplot(3,1,3)
    plot(ks,abs(fftshift(yft))/max(abs(yft)),'k');
    drawnow
end
%% spectrogram
figure(3)
pcolor(tslide,ks,abs(spec).'), shading interp
set (gca,'Ylim',[600,1200],'fontsize',[14]);
colormap(hot);
```