AMATH 482 homework 2

Author: Kuan-Wei Lee

Abstract: In this project, I am analyzing three different audio recordings. The first one is "Messiah" by Handel and the other two are the song "Mary had a little lamb" played on piano and recorder. The technique I am using to analyze the audio files is the time-frequency analysis. I will perform Gabor transformation and fast Fourier transformation on the three sets of data and produce spectrograms to see the frequency profiles of each file while maintaining some time resolution.

Sec. I

Introduction

Fourier analysis is a powerful tool when we are analyzing audio signals. Performing Fourier transformation on the data allows us to transform our signal from time of position space to the frequency space, which allows us to examine the frequency signature of the data. However, Fourier transformation collapses all the time and position resolution to frequency resolution and does not help us to understand how the frequency signature changes with time. To gain some time dependent information in the frequency space, we can rely on the time-frequency analysis. By performing both Gabor transformation to our data, we can capture the frequency signature of the data and see how the frequency profile changes with time.

Sec II.

Theoretical Background

Gabor Transformation:

Gabor transformation is a crucial part in the time-frequency analysis. What Gabor transform does is basically acting as a time filter that localized the signal over a specific window in time. The mathematical form of the Gabor transform is as followed:

$$\widetilde{f_g}(t,\omega) = \int_{-\infty}^{\infty} f(\tau)g(\tau-t)e^{-i\omega t}d\tau$$

$$g(\tau - t)\ is\ a\ function\ that\ acts\ as\ the\ time\ filter\ and\ it\ induce\ the\ localized\ Fourier\ integral$$

By choosing different shape and width of time filter $g(\tau - t)$, we can achieve obtain different time and frequency resolution of the original data. A wider filter will give us more frequency information but less time information while a narrower filter will give us more time information but less frequency information.

A modified version of Gabor transform is Wavelet analysis which allows the adjustments on the window size. Wavelet analysis starts with the consideration of a function known as the mother wavelet:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right), a\ is\ the\ scaling\ factor\ and\ b\ is\ the\ translation\ factor$$

Mother wavelets are designed to have certain property that is beneficial for a given problem. There are many choices for the mother wavelet. The common ones are:

Haar wavelet: $\psi(t) = f(x) = \begin{cases} 1,\ 0 \leq x < \frac{1}{2} \\ -1, \frac{1}{2} \leq x \geq 0 \\ 0\ \ other\ wise \end{cases}$

Mexican Hat: $\psi(t) = (1 - t^2)e^{-t^2/2}$

Gaussian : $\psi(t) = e^{-at^2}$

Sec III.

Algorithm Implementation and Development

According to the previous section, Gabor transform is a continuous integral from negative infinity to positive infinity. Since we are doing our analysis on MATLAB, we have to perform discrete Gabor transform with finite boundary. The way I implement Gabor Transform to MATLAB code is creating a function that acts as the time filter window of Gabor transform. I apply the time filter to the data and perform FFT (Fast Fourier Transform) on the time-filtered data. I also slide the window across the entire data with some finite step size. In the end, I get a n*m matrix where n is the number of time windows and m is the sample size of the data. By using the commend “pcolor”, I was able to generate the time-frequency plot of the data. In the code I wrote, I analyze the data with different width of the window, step sizes, and shapes of the window. I will discuss the findings in the later section.

Sec. IV

Computational results

In the first part of my code, I analyze the song “Messiah” by Handel. I perform Gabor transform on the audio signal and adjust the width of the time window. The graph below is the spectrogram produced by different width of Gaussian time window:

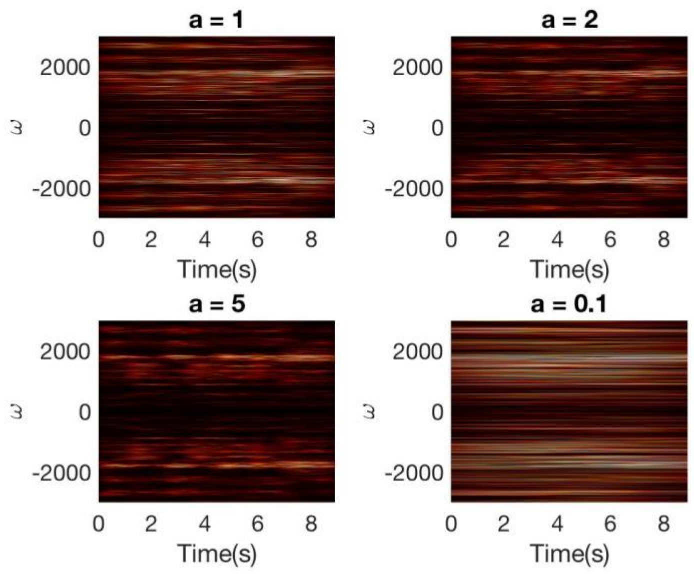


Figure 1. Spectrogram produced by different width of Gaussian windows (a = 1, 2,5,0.1)

As shown in the figure 1, the spectrogram produced by different with of Gaussian window looks very different. The parameter a indicates the width of the Gaussian window and greater a is associate with the narrower window. The spectrogram produced by greater parameter a has more time resolution because the window is narrow. Narrower windows means that the spectrogram contains more time information but less frequency information. The spectrogram with a = 0.1 has almost no time resolution but high frequency resolution because the window is very wide.

After experimenting how the width of windows affect the spectrogram, I tested how the step sizes of the adjacent window affects the spectrogram. The graph below is the spectrogram produced by fixing the window width and varying the step size.

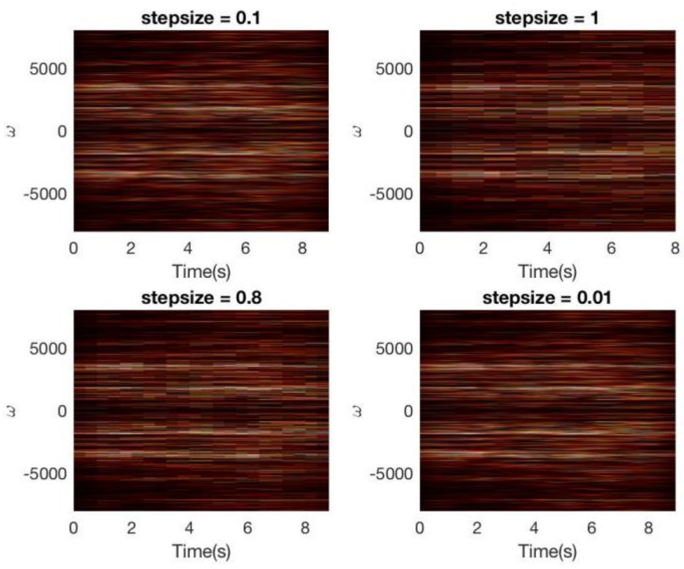


Figure 2. Spectrogram produced by fixing window width and varying step size

In figure 2 we can see that when step size is relatively large, the spectrogram looks “chunky” on the time axis. This may be caused by the under sampling of the data. When we shift the windows by too big of a step size, some information is lost. When step size is relatively small, we start to loose frequency resolution again because of the over sampling.

After exploring what does different window size and step size do to the spectrogram, it will be interesting to see how the shapes of windows affect the spectrogram. In my code, I chose to perform Gabor transformation with step function wavelet and Mexican hat wavelet and plot the spectrogram after each transformation (Figure 3).

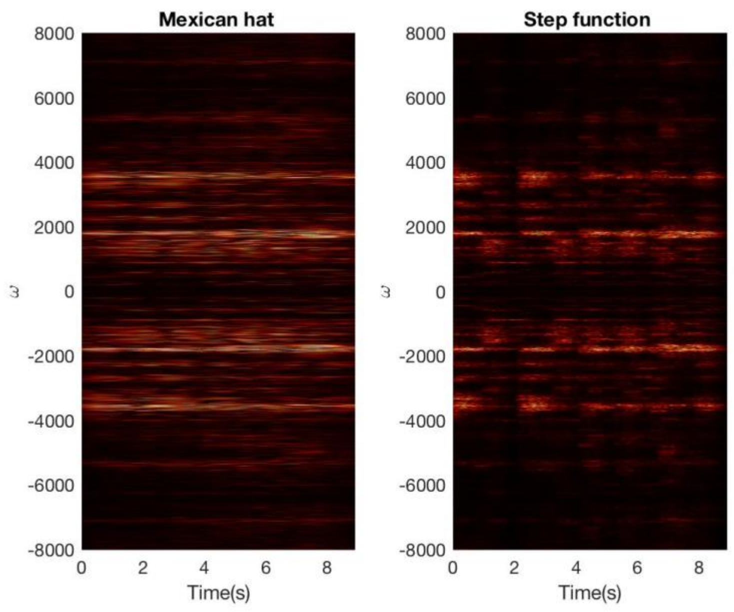
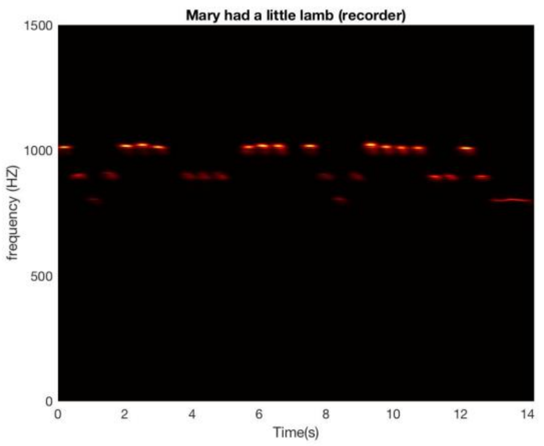


Figure 3. Spectro gram produced by different shape of window.

In figure 3 we can see that the spectrogram produced by Mexican hat has more frequency resolution compare to the spectrogram produced by step function window and the spectrogram produced by the step function window has more time resolution.

In the second part of the project, I analyze the song “Mary had a little lamb” played by piano and recorder. I perform Gabor transform on both audio signals with Gaussian window. I tried different values for the width of the window and the step size and found out that window width a = 40 and step size dt = 0.1 produced the best results. The graphs below are the spectrogram of the song played by piano and recorder.



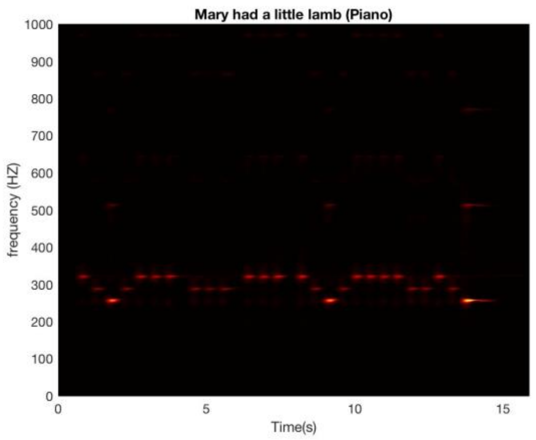


Figure 4. Spectrograms of "Mary had a little lamb" played by the recorder (left) and piano(right)

From the spectrograms above, we can see that the notes played by the recorder are higher while the notes played by piano are lower. Another interesting thing in the graph is that both instruments have some higher frequency over tones. The notes played by the piano seems "cleaner" while the notes played by the recorder is a little bit unstable as it has some frequency that are slightly lower or higher than the main frequency.

According to the frequency in the graph above, the score played by the recorder is :

C-B♭-A♭-B♭-C-C-C—B♭-B♭-B♭--C-C-C—C-B♭-A♭-C-C-C-C-B♭-B♭-C-B♭-A♭

According to the frequency in the graph above, the score played by the piano is :

E-D-C-D-E-E-E—D-D-D--E-E-E—E-D-C-D-E-E-E-E-D-D-E-D-C

Sec V.

Conclusion and summary

With Gabor transform, I was able to produce the spectrogram of the three different audio files. By manipulating the window width, step size, and the shape of the window, I was able to examine how the parameters of Gabor transform affect the spectrogram. By choosing the suitable parameters, I was able to plot the spectrogram of "Mary had a little lamb" music files and reproduce the scores by looking at the frequency in the spectrogram. With Gabor transform, we can capture the time evolution of the frequency profile that cannot be given by the Fourier transform alone.

Appendix A.

```
L = length(v)/Fs;
n = length(v);
t2 = linspace(0,L,n+1); t = t2(1:n);
k=(2*pi/(2*L))*[0:(n/2) (-n/2):-1]; ks=fftshift(k);
```

These codes are used to create vectors for the FFT.

```
tslide = 0:0.1:L;
a_vec = [1 2 5 0.1];
for jj = 1:length(a_vec)
    sgt_spec = zeros(length(tslide),n);
    a = a_vec(jj);
    for i = 1:length(tslide)
        g = exp(-a*(t-tslide(i)).^2);
        sg = v.*g;
        sg_ft = fft(sg);
        sgt_spec(i,:) = abs(fftshift(sg_ft));
    end
    subplot(2,2,jj)
    pcolor(tslide,ks,sgt_spec.')
    shading interp
    title(['a = ',num2str(a)],'Fontsize',16)
    set(gca,'Ylim',[-3000 3000],'Fontsize',16)
    ylabel('\omega')
    xlabel('Time(s)')
    colormap(hot)
end
```

This part of code is for performing Gabor transform with different window width.
a_vec contains four different values for window width.
In the for loop, `g = exp(-a*(t-tslide(i)).^2);` is the time window that is applied to the signal. After the filter is applied to the signal, I performed FFT on the signal. I plotted four spectrograms associated with different a values in a single subplot. I set the axis to zoom into the section that contains most of the frequency ($-3000 \leq \omega \leq 3000$).

```
g = (1-(t-tslide(i)).^2).*exp(-a*(t-tslide(i)).^2);
```

The code above creates a Mexican hat wavelet

```
    a = 0.5;
    width = t-tslide(i);
    g = zeros(1,length(t));
    g(0<=width&width<=a/2) = 1;
    g(a/2<=width&width<=a) = -1;
```

The code above creates a step function wavelet

Appendix B.

```
clear all
close all
clc

load handel
v = y';
plot((1:length(v))/Fs,v);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Signal of Interest, v(n)');

L = length(v)/Fs;
n = length(v);
t2 = linspace(0,L,n+1); t = t2(1:n);
k=(2*pi/(2*L))*[0:(n/2) (-n/2):-1]; ks=fftshift(k);

v_ft = fft(v);

%% Different width of Gabor window
tslide = 0:0.1:L;
a_vec = [1 2 5 0.1];
for jj = 1:length(a_vec)
    sgt_spec = zeros(length(tslide),n);
    a = a_vec(jj);
    for i = 1:length(tslide)
        g = exp(-a*(t-tslide(i)).^2);
        sg = v.*g;
        sg_ft = fft(sg);
        sgt_spec(i,:) = abs(fftshift(sg_ft));
    end
    subplot(2,2,jj)
    pcolor(tslide,ks,sgt_spec.')
    shading interp
    title(['a = ',num2str(a)],'Fontsize',16)
    set(gca,'Ylim',[-3000 3000],'Fontsize',16)
    ylabel('\omega')
    xlabel('Time(s)')
    colormap(hot)
end

%% Different step size of Gabor window

stpsize = [0.1 1 0.8 0.01];
a = 1;
for jj = 1:length(stpsize)
    tslide = 0:stpsize(jj):L;
    sgt_spec = zeros(length(tslide),n);

    for i = 1:length(tslide)
```

```
        g = exp(-a*(t-tslide(i)).^2);
        sg = v.*g;
        sg_ft = fft(sg);
        sgt_spec(i,:) = abs(fftshift(sg_ft));
    end
    subplot(2,2,jj)
    pcolor(tslide,ks,sgt_spec.')
    shading interp
    title(['stepsize = ',num2str(stpsize(jj))],'Fontsize',12)
    set(gca,'Ylim',[-8000 8000],'Fontsize',12)
    ylabel('\omega')
    xlabel('Time(s)')
    colormap(hot)
end

%% Different shape of Gabor window

tslide = 0:0.1:L;
sgt_spec = zeros(length(tslide),n);
a = 1;
for i = 1:length(tslide)
    g = (1-(t-tslide(i)).^2).*exp(-a*(t-tslide(i)).^2);
    sg = v.*g;
    sg_ft = fft(sg);
    sgt_spec(i,:) = abs(fftshift(sg_ft));
end
subplot(1,2,1)
pcolor(tslide,ks,sgt_spec.')
shading interp
title('Mexican hat ','Fontsize',12)
set(gca,'Ylim',[-8000 8000],'Fontsize',12)
ylabel('\omega')
xlabel('Time(s)')
colormap(hot)

for i = 1:length(tslide)
    a = 0.5;
    width = t-tslide(i);
    g = zeros(1,length(t));
    g(0<=width&width<=a/2) = 1;
    g(a/2<=width&width<=a) = -1;
    sg = v.*g;
    sg_ft = fft(sg);
    sgt_spec(i,:) = abs(fftshift(sg_ft));
end
subplot(1,2,2)
pcolor(tslide,ks,sgt_spec.')
shading interp
title('Step function ','Fontsize',12)
set(gca,'Ylim',[-8000 8000],'Fontsize',12)
ylabel('\omega')
```

```
xlabel('Time(s)')
colormap(hot)

%% Part 2 of HW2
clear all;close all; clc
[y,Fs] = audioread('music1.wav');
tr_piano=length(y)/Fs; % record time in seconds
plot((1:length(y))/Fs,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (piano)');

L = length(y')/Fs;
n = length(y');
t2 = linspace(0,L,n+1); t = t2(1:n);
k=(2*pi/(2*L))*[0:(n/2-1) -n/2:-1]; ks=fftshift(k);
% p8 = audioplayer(y,Fs); playblocking(p8);
%%
tslide = 0:0.1:L;
sgt_spec = zeros(length(tslide),n);
N = length(y');
dF = Fs/N;
f = -1*(Fs/2):dF:Fs/2-dF;
tau = 1;
filter = zeros(1,length(y'));
    a = 40;
    for i = 1:length(tslide)
        g = exp(-a*(t-tslide(i)).^2);
%         filter(f-500<=0) = 1;
        sg = y'.*g;
%         sg = sg.*filter;
        sg_ft = fft(sg);
%         sg_ft = fftshift(sg_ft).*filter;
        sgt_spec(i,:) = abs(fftshift(sg_ft));
%         sgt_spec(i,:) = abs(sg_ft);
    end
N = length(y');
dF = Fs/N;
f = -1*(Fs/2):dF:Fs/2-dF;

figure(2)

pcolor(tslide,f,sgt_spec.')
shading interp
set(gca,'Ylim',[0 1000],'Fontsize',12)
title('Mary had a little lamb (Piano)')
ylabel('frequency (HZ)')
xlabel('Time(s)')
colormap(hot)

%%
```

```matlab
clear all;close all; clc
[y,Fs] = audioread('music2.wav');
tr_piano=length(y)/Fs; % record time in seconds
plot((1:length(y))/Fs,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (piano)');
%%
L = length(y')/Fs;
n = length(y');
t2 = linspace(0,L,n+1); t = t2(1:n);
k=(2*pi/(2*L))*[0:(n/2-1) -n/2:-1]; ks=fftshift(k);
p8 = audioplayer(y,Fs); playblocking(p8);
tslide = 0:0.1:L;
sgt_spec = zeros(length(tslide),n);
N = length(y');
dF = Fs/N;
f = -1*(Fs/2):dF:Fs/2-dF;
tau = 1;
filter = zeros(1,length(y'));
    a = 40;
    for i = 1:length(tslide)
        g = exp(-a*(t-tslide(i)).^2);
        filter(f-500<=0) = 1;
        sg = y'.*g;
%         sg = sg.*filter;
        sg_ft = fft(sg);
%         sg_ft = fftshift(sg_ft).*filter;
        sgt_spec(i,:) = abs(fftshift(sg_ft));
%         sgt_spec(i,:) = abs(sg_ft);
    end
N = length(y');
dF = Fs/N;
f = -1*(Fs/2):dF:Fs/2-dF;

figure(2)

pcolor(tslide,f,sgt_spec.')
shading interp
set(gca,'Ylim',[0 2000],'Fontsize',12)
title('Mary had a little lamb (recorder)')
ylabel('frequency (HZ)')
xlabel('Time(s)')
colormap(hot)
```