Gabór Transform

Author: Han Song
University of Washington
Department of Applied Mathematics
Seattle, WA

Abstract: This paper contains time-frequency analysis of multiple Gabor transform wavelet filters on one piece of music and 1 filter on 2 music files of a same song. We can see the difference in spectrograms for both parts.

GitHub Repo: [GaborTransform]

I. Introduction and Overview:

In this paper, we will analyze a few pieces of music with time-frequency analysis. First part will be Handel's Messiah, and second part is two different versions (piano and record) of Mary had a little lamb. Through use of the Gabor filtering we used in class, produce spectrograms of the pieces of work; explore the window width of the Gabor transform and how it effects the spectrogram; explore the spectrogram and the idea of oversampling (i.e. using very small translations of the Gabor window versus potential undersampling (i.e. using very course/large translations of the Gabor window); use different Gabor windows. 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.

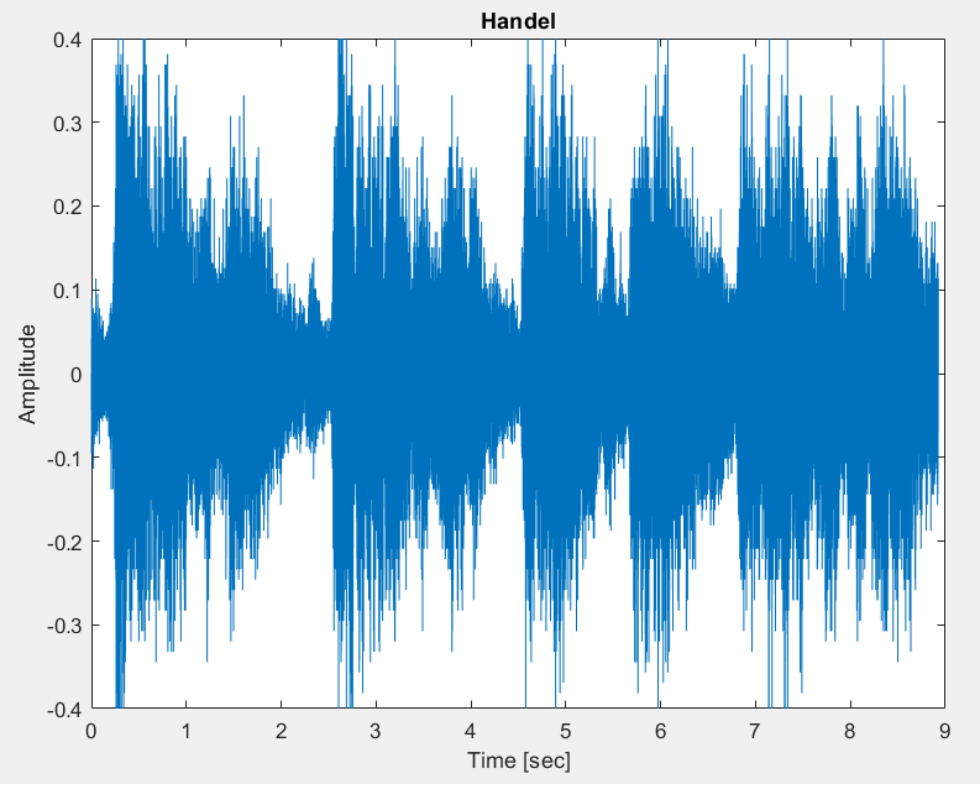

Figure 1: Frequency of Handel's Messiah in time domain

II. Theoretical Background:

In order to analyze time frequency of music, we need to understand Fourier transform. Fourier transform is an eigenfunction expansion:

$$F(k) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(x)e^{-ikx}dx \text{ and its inverse: } f(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} F(k)e^{ikx}dk$$

Where k is the wave number. However, Fourier transform only tells how intense a specific frequency overall, but not when frequency presents. Gabor transform is a hybrid tool where we apply a filter over time and extract the dominant frequency in that time window period.

$Gabor\ transform$: $g_{t,\omega}(\tau) = e^{-i\omega\tau}g(t-\tau)$ where g is the new Fourier kernel in Fourier transform.

III. Algorithm Implementation and Development:

1. Handel's Messiah

   Handel is a built-in music audio in MATLAB environment, when we load it, we are going to set it up so FFT process is optimized (i.e. periodic boundaries). I FFT the data and apply multiple wavelets window (Gaussian, Mexican hat, Shannon…) for comparison. I saved all data from filtered wavelets into spectrogram matrix for a spectrogram representation.

2. Mary had a little lamb

   For the second part of the paper, we are going to repeat the algorithm of Handel's Messiah with only one filter which I chose Gaussian and compared spectrogram of the two music files of the same song.

IV. Computational Results:

1. Handel's Messiah

a. Regular Mexican hat:

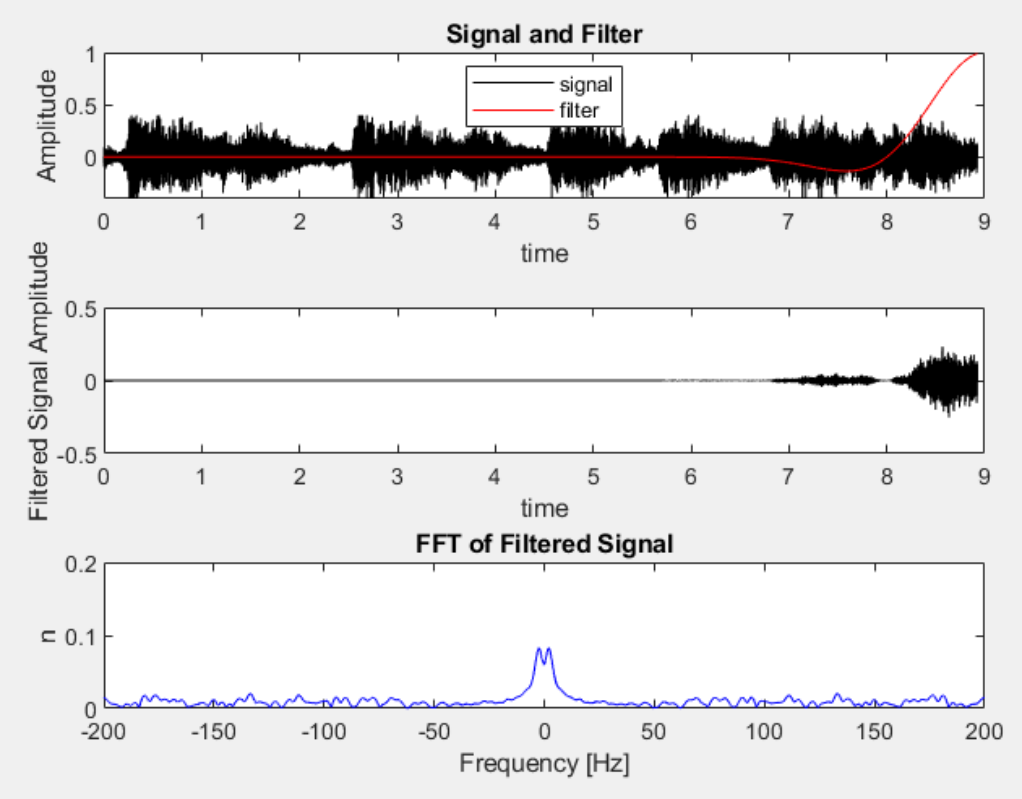

Figure 2: Gabor transform process with Mexican Hat wavelets filter on Handel's Messiah

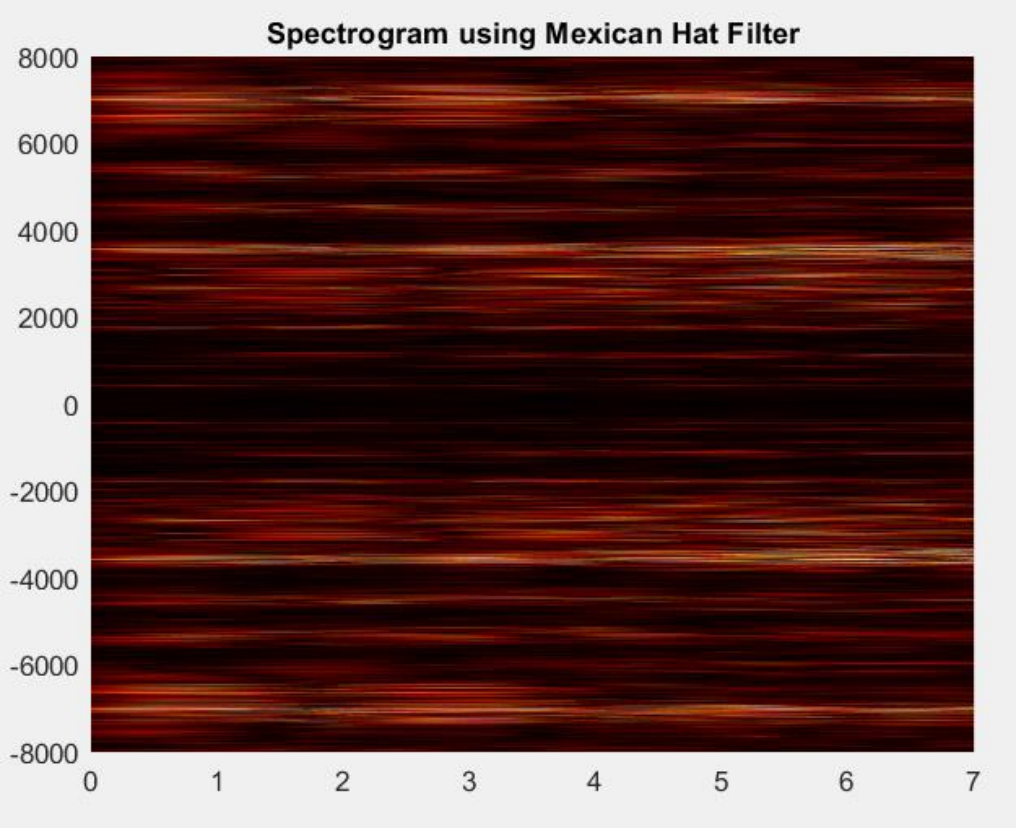

Figure 3 Spectrogram using Mexican Hat wavelets filter on Handel's Messiah

b. Big Mexican hat:

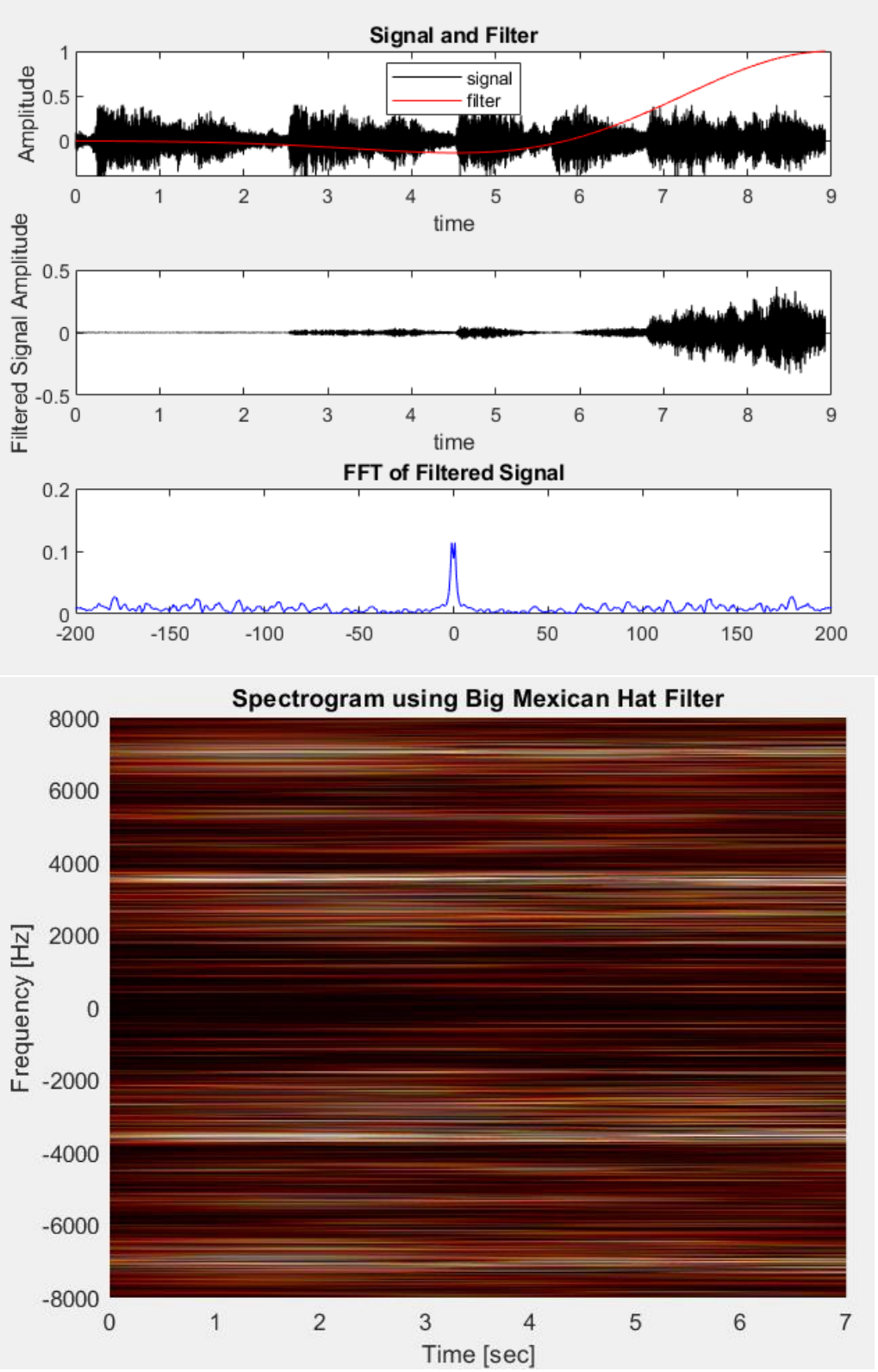

Figure 4: Gabor filter (top) and Spectrogram (bottom) using big Mexican Hat wavelets filter on Handel's Messiah

c. Small Mexican hat:

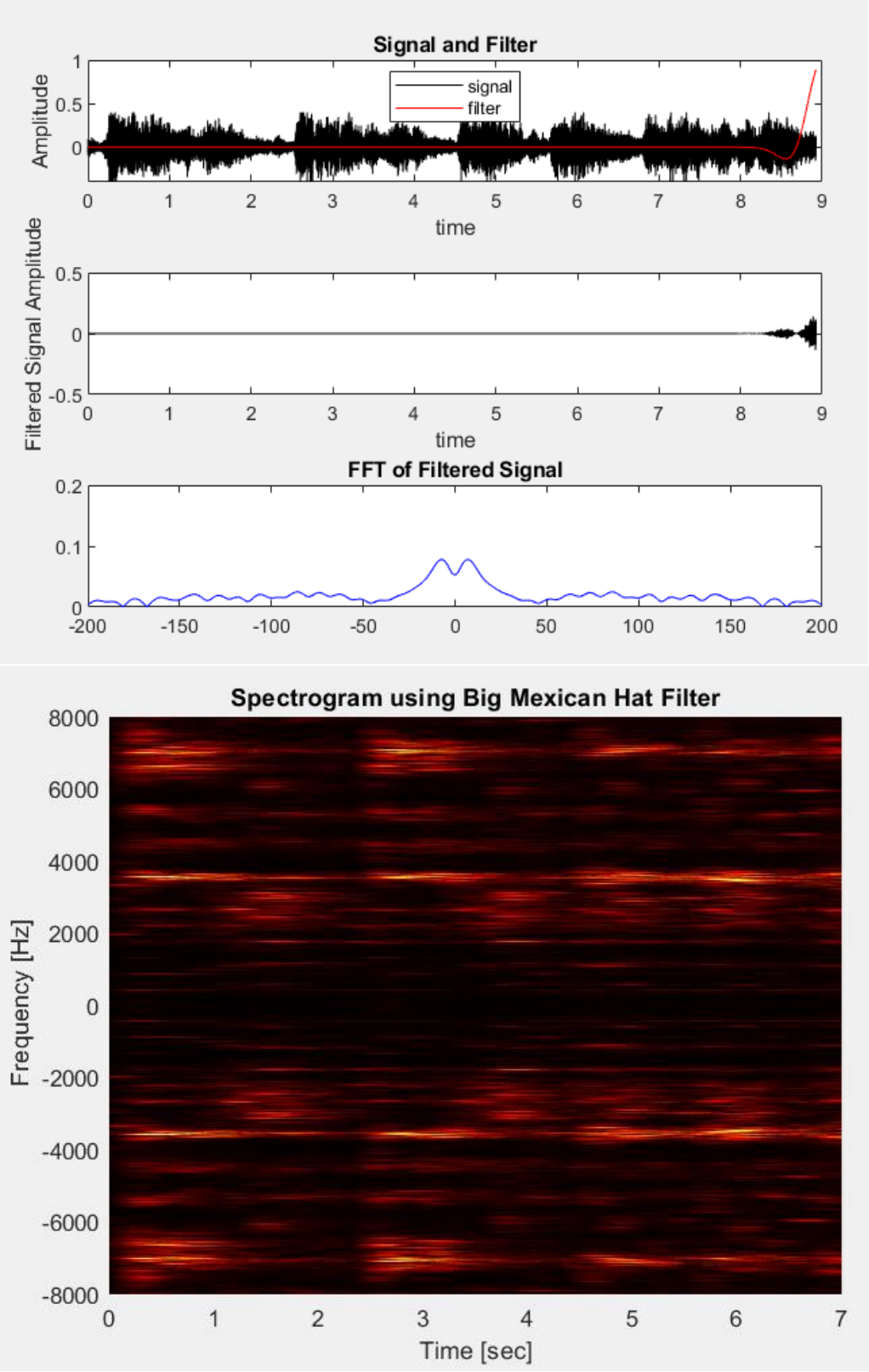

Figure 5: Gabor filter (top) and Spectrogram (bottom) using small Mexican Hat wavelets filter on Handel's Messiah

From figure 3,4, and 5, we can see that the wavelet window size matters in extracting frequency over time. There is a risk of over or under-sampling.

2. Mary had a little lamb

From plotting the songs themselves, I can see the discrepancy between the two versions of the same song. I then expected there would be different between the 2 spectrograms as well.

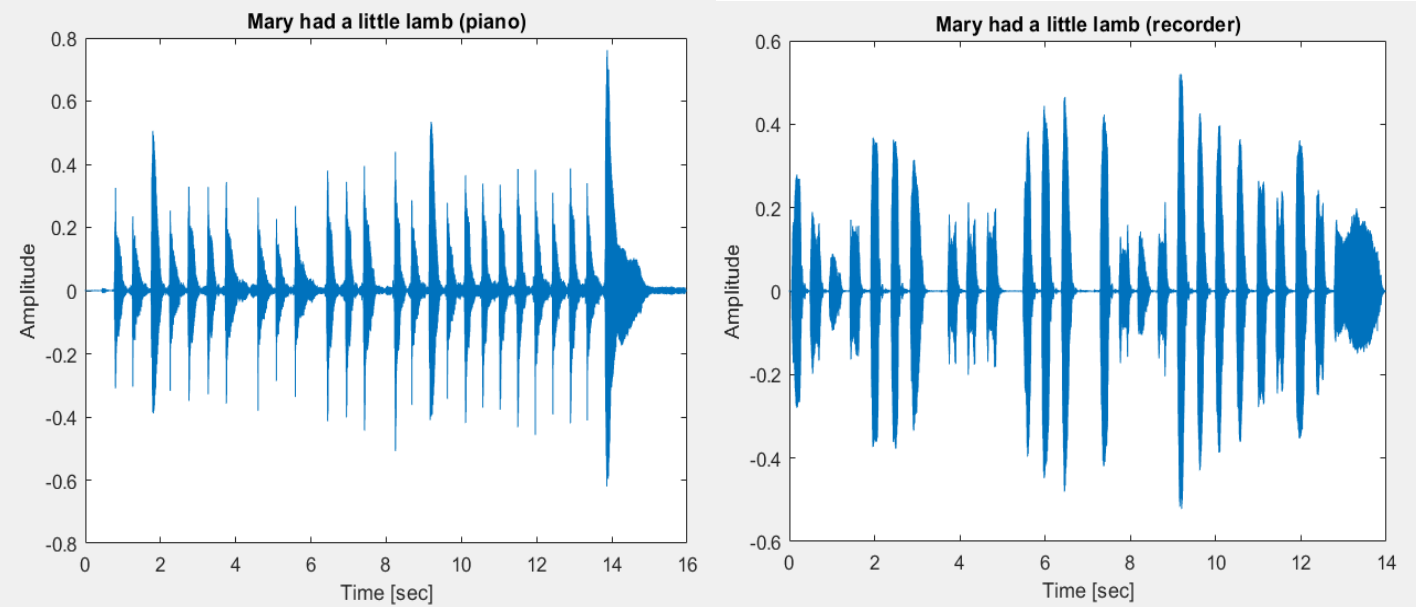

Figure 6: Mary had a little lamb

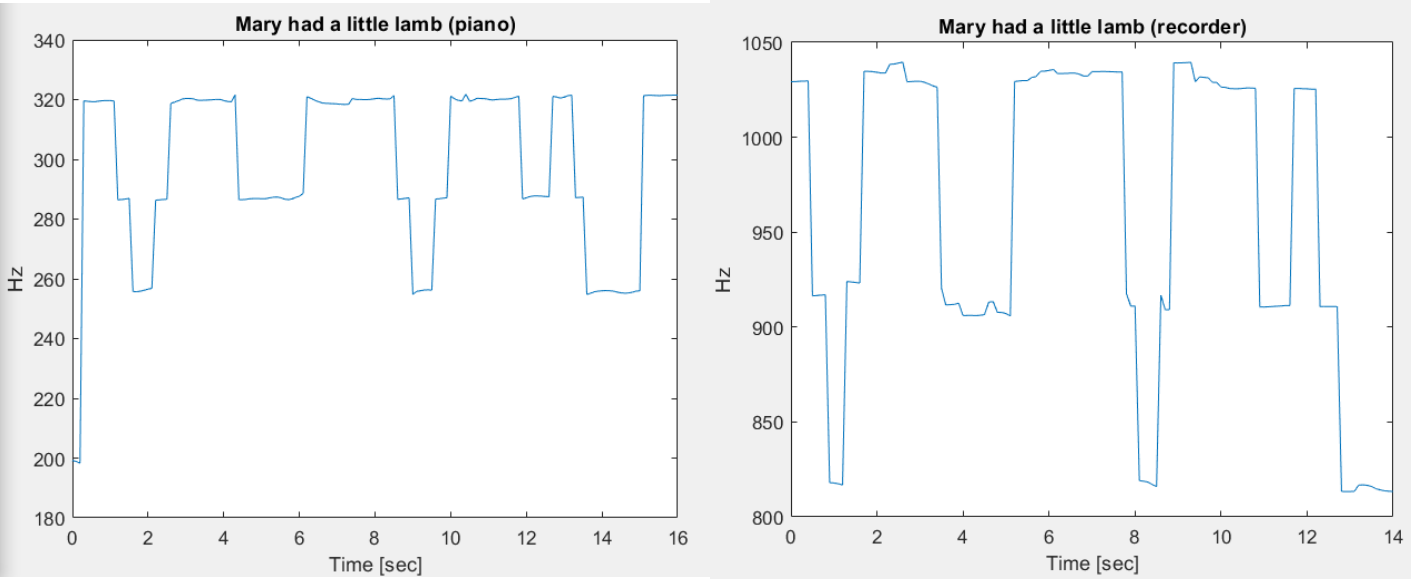

Figure 7: Time-Frequency plots of Mary had a little lamb

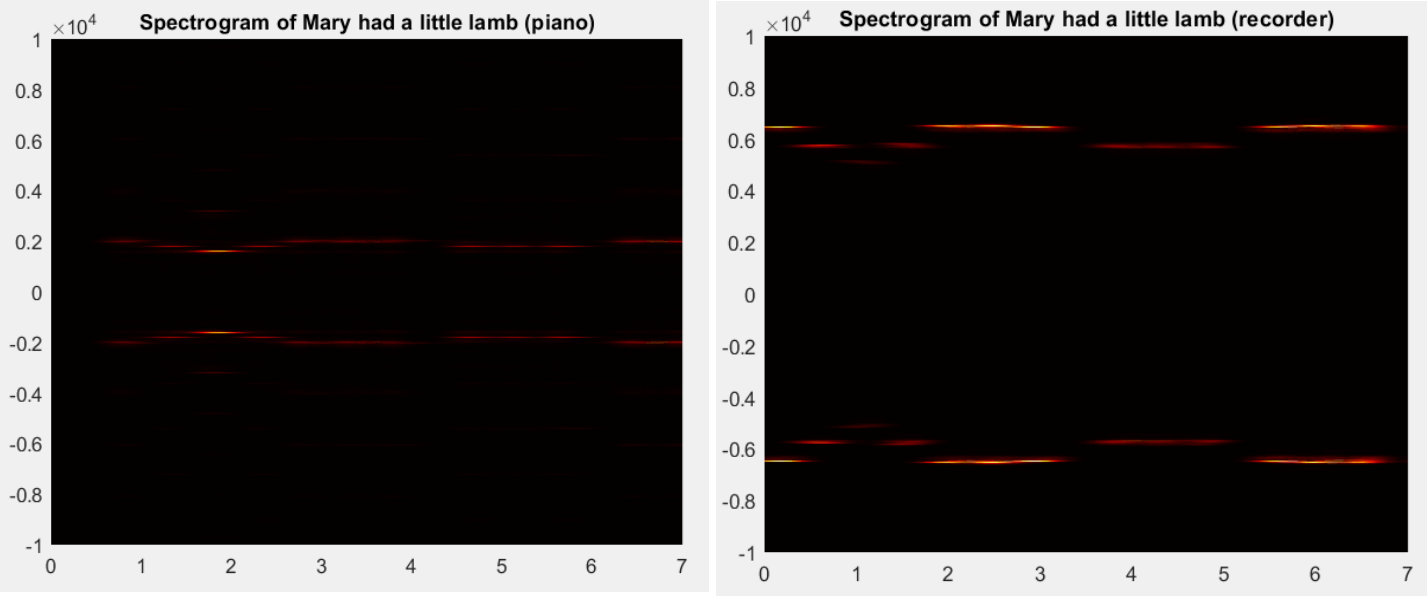

Figure 8: Spectrogram of Mary had a little lamb

As expected, the recorder version have higher frequency ranges, and showed better than the piano version.

V. Summary:

Gabor transform is useful in time-frequency analyzing. There is a risk of undersampling or oversampling, so it is crucial to find the perfect balance.

Appendix A: MATLAB functions

fft(X) is the discrete Fourier transform (DFT) of vector X. For matrices, the fft operation is applied to each column. For N-D arrays, the fft operation operates on the first non-singleton dimension.

Appendix B: MATLAB Code

```
%%%%%%%%%%%%%%%%%%%%%% HW2: Gabor transform
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: Han Song
% AMATH 582

% Part 1: Handel
clear all; close all; clc
load handel
v = y'/2;
figure(1)
plot((1:length(v))/Fs,v);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Handel');
% This is in time domain

% p8 = audioplayer(v,Fs);
% playblocking(p8);

L = 9;
n = length(v);
t2 = (1:n+1)/Fs;
t = t2(1:n);
k = (2*pi/L)*[0:n/2-1 -n/2:-1];
ks = fftshift(k);

vt = fft(v(1:end-1));
figure(2)
subplot(2,1,1)
plot(t,v)
title('Time Domain')
xlabel('time[sec]')
ylabel('Amplitude')
subplot(2,1,2)
plot(ks,fftshift(vt))
title('Frequency Domain')
xlabel('frequency [Hz]')
ylabel('n')
% Gabor

tslide = 0:0.1:9;
s = zeros(length(tslide),length(t)-1);
for j = 1:length(tslide)
    %g = exp(-(t-tslide(j)).^10);
    %g = exp(-0.01*(t-tslide(j)).^10); %bigger window
```

```
    %g = exp(-100*(t-tslide(j)).^10); %smaller window:
localizing in time
    g = (1-(t-tslide(j)).^2).*exp(-(t-tslide(j)).^2);
%Mexican hat
    %g = (1-0.1*(t-tslide(j)).^2).*exp(-0.1*(t-
tslide(j)).^2); %Mexican hat big
    %g = (1-10*(t-tslide(j)).^2).*exp(-10*(t-
tslide(j)).^2); %Mexican hat small
    %g = exp(-(t-tslide(j)).^2); %Gaussian Wavelet
    %g = exp(-0.1*(t-tslide(j)).^2); %Gaussian Big
    %g = exp(-10*(t-tslide(j)).^2); %Gaussian Small
    %g = abs(t-tslide(j)) <= 0.5; %Shannon function
    %g = 0.1*abs(t-tslide(j)) <= 0.5; %Shannon function big
    %g = 10*abs(t-tslide(j)) <= 0.5; %Shannon function
small

    vf = g.*v; % signal filter
    vft = fft(vf(1:end-1));
    s(j,:) = abs(fftshift(vft)); %save fourier transform of
the signal

    figure(3)
    subplot(3,1,1)
    plot(t,v,'k-',t,g,'r-')
    title('Signal and Filter')
    xlabel('time')
    ylabel('Amplitude')
    legend('signal','filter','Location','North')
    subplot(3,1,2)
    plot(t,vf,'k-')
    xlabel('time')
    ylabel('Filtered Signal Amplitude')
    axis([0 9 -0.5 0.5])
    subplot(3,1,3)

plot(fftshift(k),abs(fftshift(vft))/max(abs(fftshift(vft)))
,'b')
    title('FFT of Filtered Signal')
    xlabel('Frequency [Hz]')
    ylabel('n')
    axis([-200 200 0 0.2])
    pause(0.2)
end

figure(4)
```

```
pcolor(tslide,fftshift(k),s.'),shading interp,
colormap(hot)
axis([0 7 -8000 8000])
title('Spectrogram using Mexican Hat Filter')


%% Part 2: Piano

L = 16;
n = 701440;
t2 = linspace(0,L,n+1);
t = t2(1:n); % make n+1 points and throw out the last point
k = (2*pi/L)*[0:n/2-1 -n/2:-1];

t_p=16; % record time in seconds
y=audioread('music1.wav');
Fs=length(y)/t_p;
y = y.';
figure(5)
plot((1:length(y))/Fs,y);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (piano)'); drawnow

figure(6)
tslide = 0:0.1:16;
Max = zeros(length(tslide),1);
s = zeros(length(tslide), length(t));
for j = 1:length(tslide)
    g = exp(-20*(t-tslide(j)).^2); %simple Gaussian
    Sf = g.*y; % signal filter
    Sft = fft(Sf);
    [f_max,ind] = max(Sft);
    Max(j) = k(ind);
    s(j,:) = abs(fftshift(Sft));

    subplot(3,1,1)
    plot(t,y,'k-',t,g,'r-')
    title('Signal and Filter')
    xlabel('time')
    ylabel('Amplitude')
    legend('signal','filter','Location','North')
    subplot(3,1,2)
    plot(t,Sf,'k-')
    xlabel('time')
    ylabel('Filtered Signal Amplitude')
    axis([0 16 -1 1])
```

```
    subplot(3,1,3)

plot(fftshift(k),abs(fftshift(Sft))/max(abs(fftshift(Sft)))
,'b')
    title('FFT of Filtered Signal')
    xlabel('Frequency [Hz]')
    ylabel('n')
    axis([-100 100 0 0.1])
    pause(0.2)
end
figure(7)
plot(tslide,abs(Max./(2*pi)))
xlabel('Time [sec]'); ylabel('Hz');
title('Mary had a little lamb (piano)')
figure(8)
pcolor(tslide,fftshift(k),s.'),shading interp,
colormap(hot)
axis([0 7 -10000 10000])
title('Spectrogram of Mary had a little lamb (piano)')

% Record
L = 14;
n = 627712;
t2 = linspace(0,L,n+1);
t = t2(1:n); % make n+1 points and throw out the last point
k = (2*pi/L)*[0:n/2-1 -n/2:-1];

figure(9)
tr_rec=14; % record time in seconds
y2=audioread('music2.wav'); Fs2=length(y2)/tr_rec;
y2 = y2.';
plot((1:length(y2))/Fs2,y2);
xlabel('Time [sec]'); ylabel('Amplitude');
title('Mary had a little lamb (recorder)');

tslide = 0:0.1:14;
Max = zeros(length(tslide),1);
s = zeros(length(tslide), length(t));
for j = 1:length(tslide)
    g = exp(-20*(t-tslide(j)).^2); % simple Gaussian
    Sf2 = g.*y2; % signal filter
    Sft2 = fft(Sf2);
    [f_max2,ind] = max(Sft2);
    Max(j) = k(ind);
    s(j,:) = abs(fftshift(Sft2));
end
```

```
figure(10)
plot(tslide,abs(Max./(2*pi)))
xlabel('Time [sec]'); ylabel('Hz');
title('Mary had a little lamb (recorder)')
figure(11)
pcolor(tslide,fftshift(k),s.'),shading interp,
colormap(hot)
axis([0 7 -10000 10000])
title('Spectrogram of Mary had a little lamb (recorder)')
```