Jeremy Aguirre

AMATH 482

February 7, 2020

Assignment #2:

The Gábor Transform Method for Time-Frequency Analysis

Abstract

In this assignment, an excerpt from Handel's Messiah was evaluated using time-frequency analysis. The method of evaluation was Gábor filtering with a Gaussian window, Mexican hat wavelet, and a Shannon window. Various window widths were tested to determine their effects on a resulting spectrogram. The effects of oversampling and undersampling on the spectrogram were also investigated. Excerpts of Mary Had a Little Lamb on piano and recorder were also evaluated using Gábor filtering. The information from the resulting spectrogram was used to reproduce the score for this song.

I. Introduction and Overview

In this assignment, an excerpt from Handel's Messiah was evaluated using time-frequency analysis. The method of evaluation was Gábor filtering with a Gaussian window, Mexican hat wavelet, and a Shannon window. Various window widths were tested to determine their effects on a resulting spectrogram. The effects of oversampling and undersampling on the spectrogram were also investigated. Excerpts of Mary Had a Little Lamb on piano and recorder were also evaluated using Gábor filtering. The information from the resulting spectrogram was used to reproduce the score for this song.

II. Theoretical Background

Gábor Transform

The Gábor transform is defined by the following equation:

$$\mathcal{G}[f](t,\omega) = \tilde{f}_g(t,\omega) = \int_{-\infty}^{\infty} f(\tau)\bar{g}(\tau - t)e^{-i\omega\tau}d\tau = (f, \bar{g}_{t,\omega}), \qquad (1)$$

It is also identified as the short-time Fourier transform (STFT). The Gábor transform extracts frequency information from a signal by sliding a time-filtering window across the signal. Each point in time across the signal is localized within this window. In this way, the Gábor transform is able to capture both time and frequency content across a signal. The Gábor transform is often used to analyze vocalization patterns and express their time-frequency properties in a spectrogram. Gábor filtering can utilize windows of varying shape in order to extract time-frequency information, such as the Gaussian, Mexican hat, and Shannon windows.

Gaussian Window

The Gaussian window is characterized by the parameters a and b which represent width and translation of the window across a signal, respectively.

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

Mexican Hat Wavelet

The Mexican hat wavelet is the second moment of the Gaussian in the frequency domain. It is characterized by the following equation:

$$\psi(t) = \frac{2}{\sqrt{3\sigma}\pi^{1/4}}\left(1-\left(\frac{t}{\sigma}\right)^2\right)e^{-\frac{t^2}{2\sigma^2}} \tag{3}$$

Shannon Window

The Shannon window is simply a step function.

III. Algorithm Implementation and Development

A portion of Handel's Messiah is stored in MATLAB and can be called by the command 'load handel.' This command can be used to produce the sampling frequency and a vector of the amplitude values over time. A vector representing time is produced by dividing L by the sampling frequency, where L is the length of the amplitude vector. In order to use the fft to extract the frequency information the wavenumber k is defined over [-L,L-1] and is rescaled by 2pi/L. This is done because the Fourier transform works over a period of 2pi and the -1 is added to the domain to account for aliasing. This requires that one of the amplitude values be discarded so that the amplitude vector has the same length as k. fftshift(k) is used to define ks to plot in the frequency domain. A time domain plot is made using the amplitude vector and the time vector. A frequency domain plot is made using ks and the normalized fft of the amplitude vector.

Equation two is used to create a Gaussian window with a width of a and a sliding translation parameter b. This window is applied to the amplitude vector using a for loop to apply the Gábor filtering method. Similar for loops are used to apply a Mexican hat wavelet and a Shannon window to the signal. From here, multiple values for the window width are tested to learn their effects on the resulting spectrograms. The spectrograms were created using the pcolor plotting command in MATLAB. After this, different scaling factors for the sliding translation parameter b are tested to determine the effects of oversampling and undersampling on the spectrograms.

For the analysis of Mary Had a Little Lamb, the audioread function in MATLAB was used to extract the amplitude vector and sampling frequency from two .wav audio

files. The Gábor filtering method was used with a Gaussian window to extract the time frequency information from these audio files. The for loop used is identical to the previous one. After the spectrograms were produced for each audio file, the frequency information was converted from angular frequency to frequency in hertz. With this frequency information and the figure provided, the music score for Mary Had a Little Lamb was reproduced for both piano and recorder.

IV. Computational Results

Part 1

Exploring window width

Widths of 0.2, 2, and 20 were used for the Gaussian window, the Mexican hat wavelet, and the Shannon window to produce the following spectrograms. It was determined by comparing the spectrograms that increasing the width of the filtering window lead to an increase in frequency resolution while decreasing time resolution. This result was expected due to the Heisenberg relationship that exists between time and frequency. Longer time-filtering windows are able to collect more frequency information, but there is a loss in time resolution due to their size. A shorter time-filtering window yields better time resolution but is less capable of collecting frequency information.

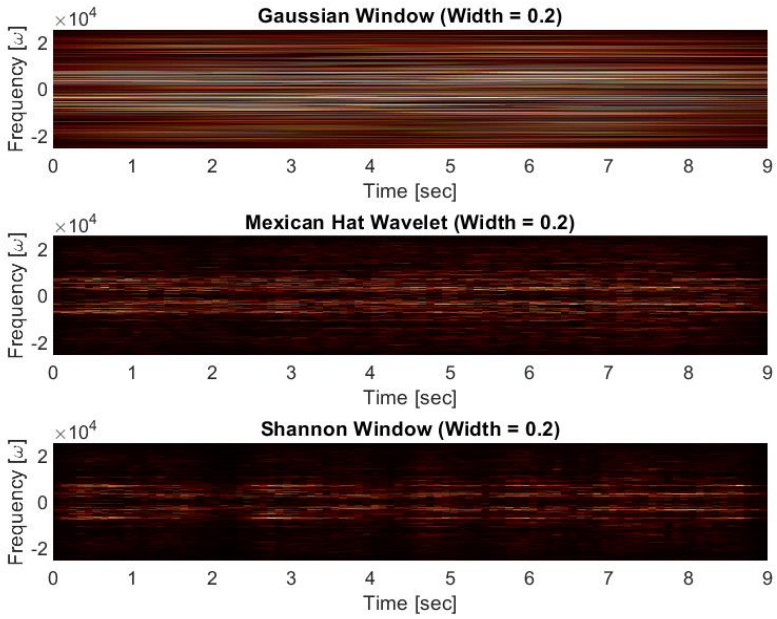

Figure 1. Gaussian window, Mexican hat wavelet, and Shannon window spectrograms for width of 0.2

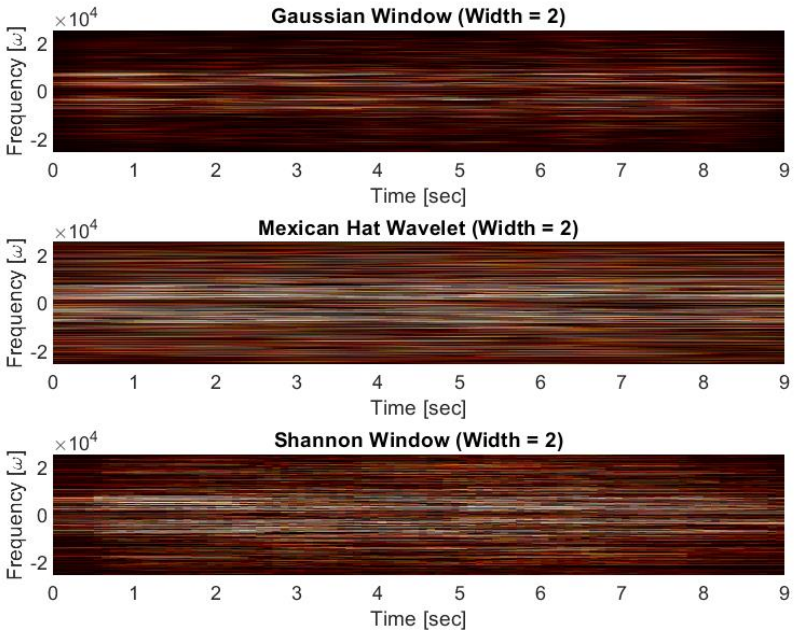

Figure 2. Gaussian window, Mexican hat wavelet, and Shannon window spectrograms for width of 2

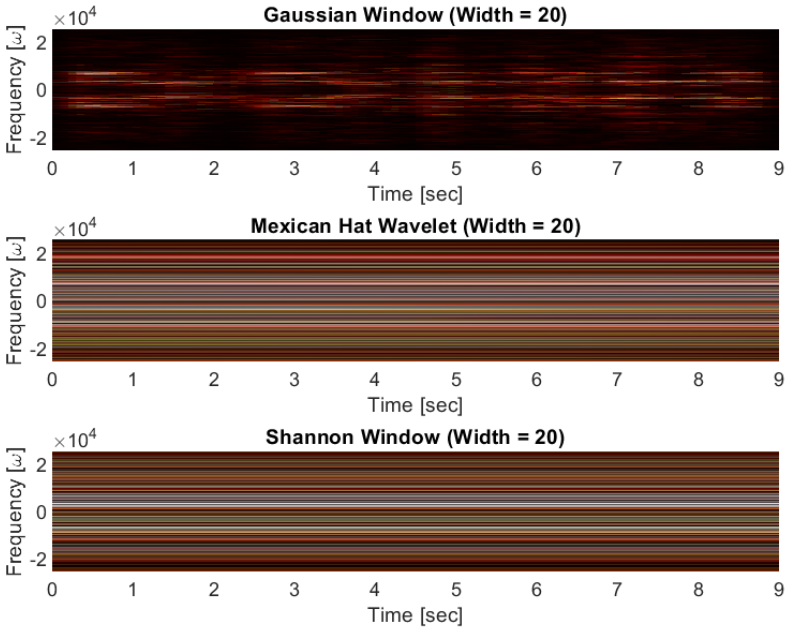

Figure 3. Gaussian window, Mexican hat wavelet, and Shannon window spectrograms for width of 20

Effects of Oversampling and Undersampling

A sliding translation parameter b value of 1 was used to simulate undersampling and a value of .05 to simulate oversampling. Based on a comparison of the spectrograms produced below, it is clear that there are problems with undersampling. With undersampling, there is less overlap between time-filtering windows which leads to a lower resolution in both time and frequency. However, oversampling seems to be effective at reducing noise and producing a high-resolution spectrogram. Another added benefit of oversampling is the elimination of aliases.

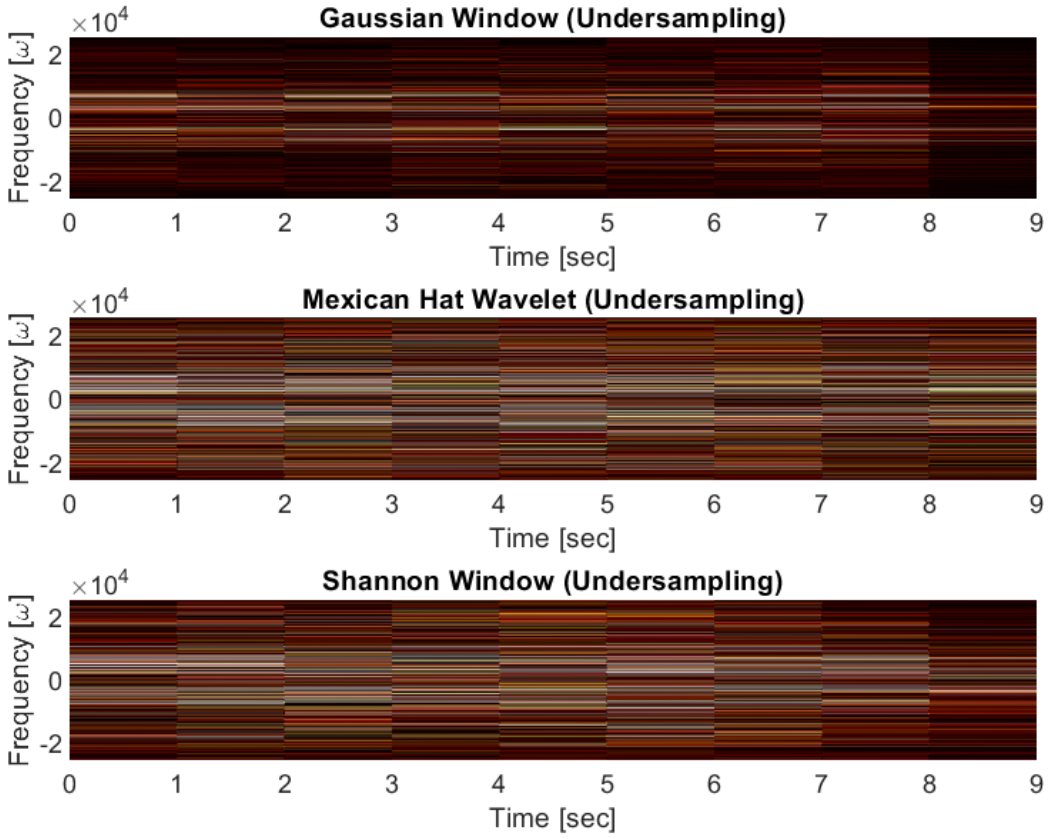

Figure 4. Gaussian window, Mexican hat wavelet, and Shannon window spectrograms with undersampling

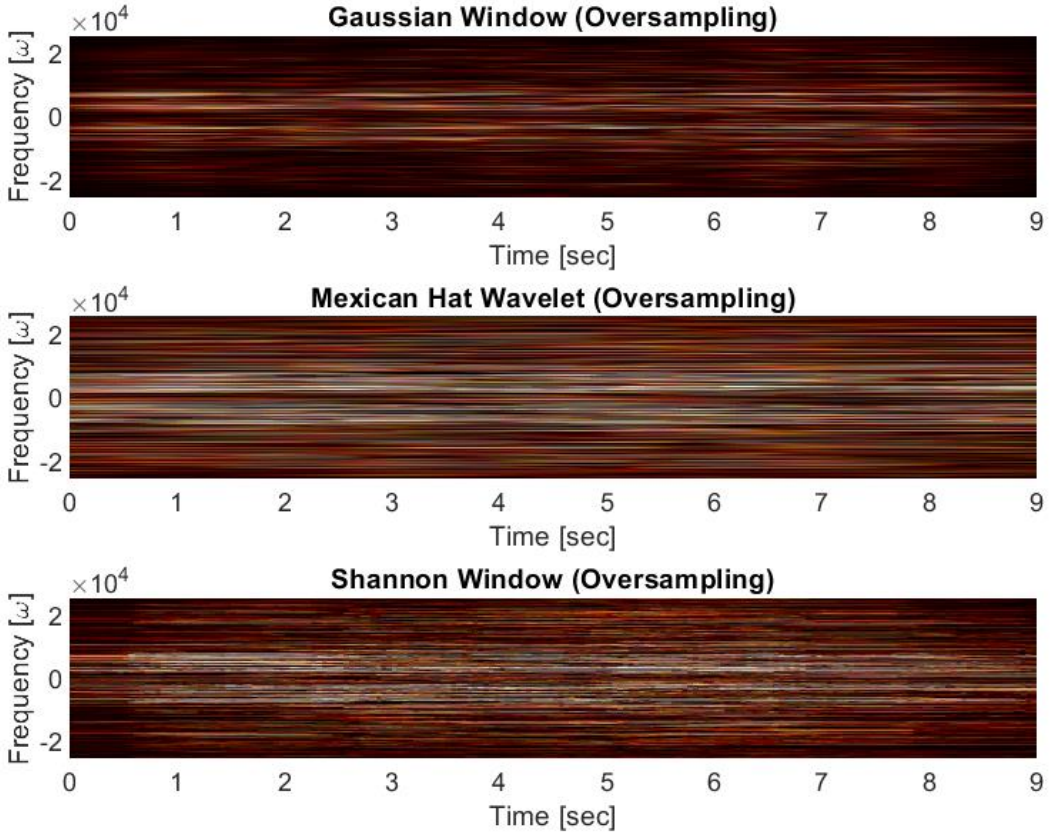

Figure 5. Gaussian window, Mexican hat wavelet, and Shannon window spectrograms with oversampling

Comparison of Gaussian window, Mexican hat wavelet, and Shannon window

The Gaussian window appears to be the most effective all around. It appears to maintain a balance between time resolution and frequency resolution in response to changes in window width. The shape of the Gaussian window is simply better suited due to the Heisenberg principle. The Shannon window appears to be the least effective due to its low resolution in time and frequency.

Part 2

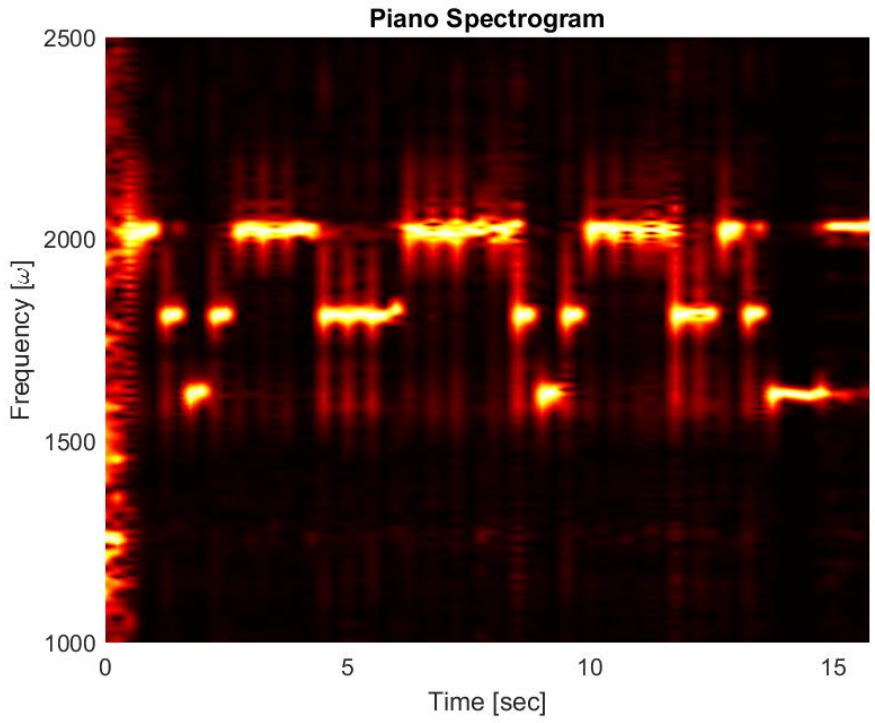

Figure 6. Spectrogram of the piano recording of Mary had a little lamb

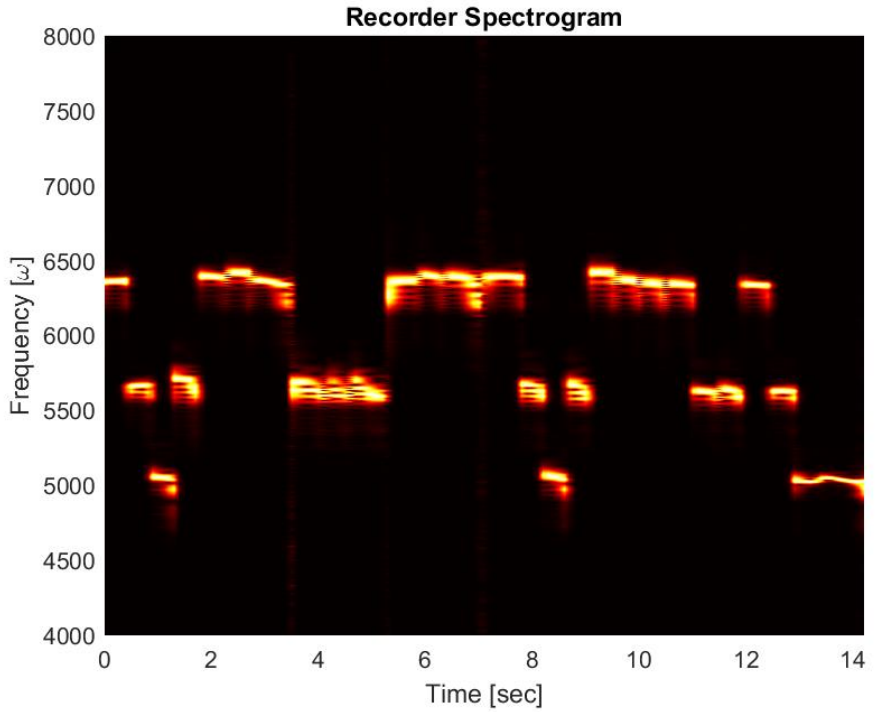

Figure 7. Spectrogram of the recorder recording of Mary had a little lamb

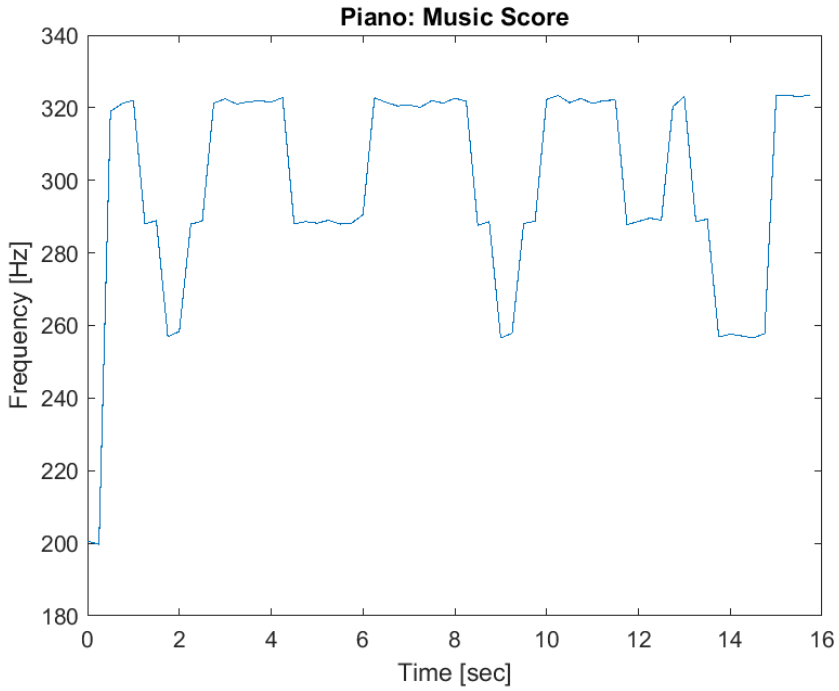

Figure 8. Music Score for Piano recording in frequencies

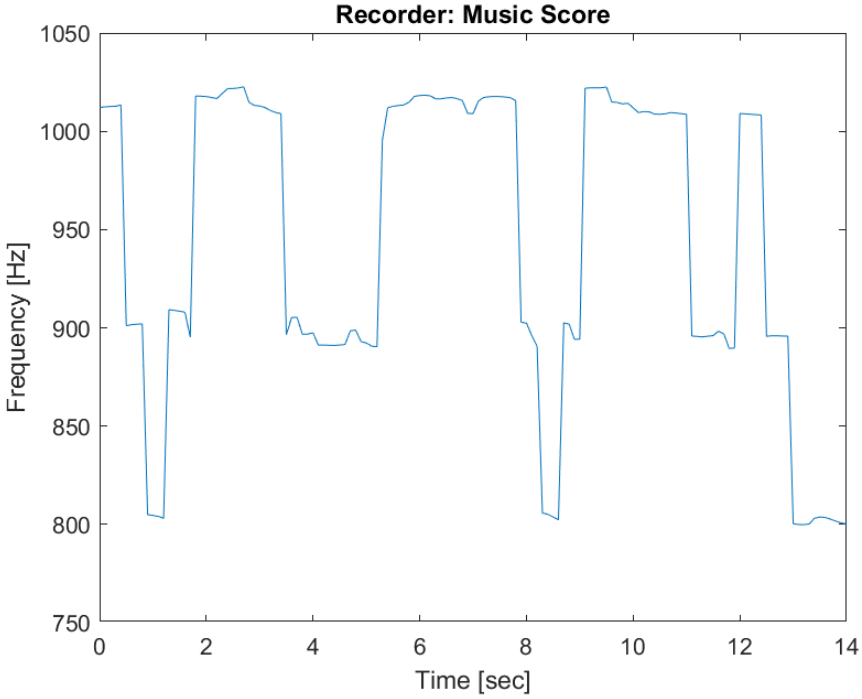

Figure 9. Music Score for recorder recording in frequencies

Music Score

According to the figure provided, the piano score is:
EDCDEEEDDDEEEEDCDEEEEDDEDC.

According to the figure provided, the recorder score is:
BAGABBBAAABBBBAGABBBBAABAG.

Differences between recorder and piano

Based on the spectrograms produced, it is clear that the recorder is playing much higher notes than the piano, almost two octaves higher. The piano appears to have many more overtones than the recorder.

V. Summary and Conclusions

In this assignment, an excerpt from Handel's Messiah was evaluated using time-frequency analysis. The method of evaluation was Gábor filtering with a Gaussian window, Mexican hat wavelet, and a Shannon window. Various window widths were tested to determine their effects on a resulting spectrogram. The effects of oversampling and undersampling on the spectrogram were also investigated. Excerpts of Mary Had a Little Lamb on piano and recorder were also evaluated using Gábor filtering. The information from the resulting spectrogram was used to reproduce the score for this song. The Gábor transform is an important signal processing tool for analyzing vocalization patterns and speech. The methods used in this assignment have excellent potential future applications in these areas.

Appendix A

fft - applies the fast Fourier transform algorithm within MATLAB to a vector

fftshift – used to shift the center frequency to zero and back again as needed for easier analysis

pcolor(X,Y,C) - draws a pseudocolor plot of the elements of C at the location specified by X and Y

audioread - reads data from a .wav file and returns the amplitude and sampling frequency

Appendix B

```
%% AMATH 482: Assignment #2
clear all; close all; clc;
load handel
v = y(1:length(y)-1)';
```

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

figure(1)
subplot(2,1,1)
plot(t,v);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Signal of Interest, v(n)');

subplot(2,1,2)
plot(ks,abs(fftshift(vt))/max(abs(vt)))
xlabel('Frequency [\omega]');
ylabel('FFT(v)');

%p8 = audioplayer(S,Fs);
%playblocking(p8);
print('-f1','Signal_of_Interest','-dpng')
%% Gaussian Window with Width of 0.2
a = 0.2;
tstep1 = 0.1;
vgt_spec = [];
b1 = 0:tstep1:L;
for i=1:length(b1)
    g = exp(-a*(t-b1(i)).^2);
    vg = g.*v;
    vgt = fft(vg);
    vgt_spec = [vgt_spec; abs(fftshift(vgt))];
end

%% Mexican Hat Wavelet with Width of 0.2
a = 0.2;
tstep2 = 0.1;
vmt_spec = [];
b2 = 0:tstep2:L;
for j = 1:length(b2)
    m = 2/(sqrt(3*a)*(pi)^(1/4))*(1-((t - b2(j))/a).^2)...
        .* exp(-(t - b2(j)).^2 / (2*a^2));
    vm = m.*v; vgt = fft(vm);
    vmt_spec = [vmt_spec; abs(fftshift(vgt))];
end

%% Shannon Window with Width of 0.2
```

```
window_width = 0.2;
tstep3 = 0.1;
vst_spec = [];
b3 = 0:tstep3:L;

for k=1:length(b3)
    s = (abs(t - b3(k)) < window_width);
    vs = s.*v;
    vst = fft(vs);
    vst_spec = [vst_spec; abs(fftshift(vst))];
end

%% Spectrograms for Width of 0.2
figure (2)
subplot(3,1,1);
pcolor(b1,ks,vgt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Gaussian Window (Width = 0.2)')
colormap(hot)

subplot(3,1,2)
pcolor(b2,ks,vmt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Mexican Hat Wavelet (Width = 0.2)');
colormap(hot)

subplot(3,1,3);
pcolor(b3,ks,vst_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Shannon Window (Width = 0.2)');
colormap(hot)

print('-f2','Width_02','-dpng')
%% Gaussian Window with Width of 2
a = 2;
tstep1 = 0.1;
vgt_spec = [];
b1 = 0:tstep1:L;
for i=1:length(b1)
    g = exp(-a*(t-b1(i)).^2);
    vg = g.*v;
    vgt = fft(vg);
    vgt_spec = [vgt_spec; abs(fftshift(vgt))];
end
```

```
%% Mexican Hat Wavelet with Width of 2
a = 2;
tstep2 = 0.1;
vmt_spec = [];
b2 = 0:tstep2:L;
for j = 1:length(b2)
    m = 2/(sqrt(3*a)*(pi)^(1/4))*(1-((t - b2(j))/a).^2)...
        .* exp(-(t - b2(j)).^2 / (2*a^2));
    vm = m.*v; vgt = fft(vm);
    vmt_spec = [vmt_spec; abs(fftshift(vgt))];
end

%% Shannon Window with Width of 2
window_width = 2;
tstep3 = 0.1;
vst_spec = [];
b3 = 0:tstep3:L;

for k=1:length(b3)
    s = (abs(t - b3(k)) < window_width);
    vs = s.*v;
    vst = fft(vs);
    vst_spec = [vst_spec; abs(fftshift(vst))];
end

%% Spectrograms for Width of 2
figure (3)
subplot(3,1,1);
pcolor(b1,ks,vgt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Gaussian Window (Width = 2)')
colormap(hot)

subplot(3,1,2)
pcolor(b2,ks,vmt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Mexican Hat Wavelet (Width = 2)');
colormap(hot)

subplot(3,1,3);
pcolor(b3,ks,vst_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Shannon Window (Width = 2)');
```

```
colormap(hot)

print('-f3','Width_2','-dpng')
%% Gaussian Window with Width of 20
a = 20;
tstep1 = 0.1;
vgt_spec = [];
b1 = 0:tstep1:L;
for i=1:length(b1)
    g = exp(-a*(t-b1(i)).^2);
    vg = g.*v;
    vgt = fft(vg);
    vgt_spec = [vgt_spec; abs(fftshift(vgt))];
end

%% Mexican Hat Wavelet with Width of 20
a = 20;
tstep2 = 0.1;
vmt_spec = [];
b2 = 0:tstep2:L;
for j = 1:length(b2)
    m = 2/(sqrt(3*a)*(pi)^(1/4))*(1-((t - b2(j))/a).^2)...
        .* exp(-(t - b2(j)).^2 / (2*a^2));
    vm = m.*v; vgt = fft(vm);
    vmt_spec = [vmt_spec; abs(fftshift(vgt))];
end

%% Shannon Window with Width of 20
window_width = 20;
tstep3 = 0.1;
vst_spec = [];
b3 = 0:tstep3:L;

for k=1:length(b3)
    s = (abs(t - b3(k)) < window_width);
    vs = s.*v;
    vst = fft(vs);
    vst_spec = [vst_spec; abs(fftshift(vst))];
end

%% Spectrograms for Width of 20
figure (4)
subplot(3,1,1);
pcolor(b1,ks,vgt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Gaussian Window (Width = 20)')
```

```
colormap(hot)

subplot(3,1,2)
pcolor(b2,ks,vmt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Mexican Hat Wavelet (Width = 20)');
colormap(hot)

subplot(3,1,3);
pcolor(b3,ks,vst_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Shannon Window (Width = 20)');
colormap(hot)

print('-f4','Width_20','-dpng')
%% Gaussian Window: Oversampling
a = 2;
tstep1 = 0.05;
vgt_spec = [];
b1 = 0:tstep1:L;
for i=1:length(b1)
    g = exp(-a*(t-b1(i)).^2);
    vg = g.*v;
    vgt = fft(vg);
    vgt_spec = [vgt_spec; abs(fftshift(vgt))];
end

%% Mexican Hat Wavelet: Oversampling
a = 2;
tstep2 = 0.05;
vmt_spec = [];
b2 = 0:tstep2:L;
for j = 1:length(b2)
    m = 2/(sqrt(3*a)*(pi)^(1/4))*(1-((t - b2(j))/a).^2)...
        .* exp(-(t - b2(j)).^2 / (2*a^2));
    vm = m.*v; vgt = fft(vm);
    vmt_spec = [vmt_spec; abs(fftshift(vgt))];
end

%% Shannon Window: Oversampling
window_width = 2;
tstep3 = 0.05;
vst_spec = [];
b3 = 0:tstep3:L;
```

```
for k=1:length(b3)
    s = (abs(t - b3(k)) < window_width);
    vs = s.*v;
    vst = fft(vs);
    vst_spec = [vst_spec; abs(fftshift(vst))];
end

%% Spectrograms (Oversampling)
figure (5)
subplot(3,1,1);
pcolor(b1,ks,vgt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Gaussian Window (Oversampling)')
colormap(hot)

subplot(3,1,2)
pcolor(b2,ks,vmt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Mexican Hat Wavelet (Oversampling)');
colormap(hot)

subplot(3,1,3);
pcolor(b3,ks,vst_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Shannon Window (Oversampling)');
colormap(hot)

print('-f5','Oversampling','-dpng')
%% Gaussian Window: Undersampling
a = 2;
tstep1 = 1;
vgt_spec = [];
b1 = 0:tstep1:L;
for i=1:length(b1)
    g = exp(-a*(t-b1(i)).^2);
    vg = g.*v;
    vgt = fft(vg);
    vgt_spec = [vgt_spec; abs(fftshift(vgt))];
end

%% Mexican Hat Wavelet: Undersampling
a = 2;
tstep2 = 1;
vmt_spec = [];
```

```
b2 = 0:tstep2:L;
for j = 1:length(b2)
    m = 2/(sqrt(3*a)*(pi)^(1/4))*(1-((t - b2(j))/a).^2)...
        .* exp(-(t - b2(j)).^2 / (2*a^2));
    vm = m.*v; vgt = fft(vm);
    vmt_spec = [vmt_spec; abs(fftshift(vgt))];
end

%% Shannon Window: Undersampling
window_width = 2;
tstep3 = 1;
vst_spec = [];
b3 = 0:tstep3:L;

for k=1:length(b3)
    s = (abs(t - b3(k)) < window_width);
    vs = s.*v;
    vst = fft(vs);
    vst_spec = [vst_spec; abs(fftshift(vst))];
end

%% Spectrograms (Undersampling)
figure (6)
subplot(3,1,1);
pcolor(b1,ks,vgt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Gaussian Window (Undersampling)')
colormap(hot)

subplot(3,1,2)
pcolor(b2,ks,vmt_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Mexican Hat Wavelet (Undersampling)');
colormap(hot)

subplot(3,1,3);
pcolor(b3,ks,vst_spec.'), shading interp
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
title('Shannon Window (Undersampling)');
colormap(hot)

print('-f6','Undersampling','-dpng')

%% Part 2: Piano
```

```
clear all, close all, clc
[y,Fs] = audioread('music1.wav');
v = y';
L = length(v)/Fs;
n = length(v);
tr_piano = (1:length(v))/Fs;
k = (2*pi/(L)) * [0:n/2-1 -n/2:-1];
ks = fftshift(k);
vt = fft(v);

figure(7)
subplot(2,1,1)
plot(tr_piano,v);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Mary had a little lamb (piano)');

subplot(2,1,2)
plot(ks,abs(fftshift(vt))/max(abs(vt)))
xlabel('Frequency [\omega]');
ylabel('FFT(v)');
print('-f7','Piano','-dpng')

p8 = audioplayer(y,Fs); playblocking(p8);

a = 50;
tspan = 0.25;
Piano_Score = [];
vgt_spec = [];
b = 0:tspan:L;
for i=1:length(b)
    g=exp(-a*(tr_piano-b(i)).^2);
    vg=g.*v;
    vgt=fft(vg);
    [M, I] = max(abs(vgt));
    Piano_Score = [Piano_Score; abs(k(I))/(2*pi)];
    vgt_spec=[vgt_spec; abs(fftshift(vgt)) / max(abs(vgt))];
end

figure(8)
plot(b,Piano_Score)
title('Piano: Music Score');
xlabel('Time [sec]');
ylabel('Frequency [Hz]');
print('-f8','Piano_Score','-dpng')

figure(9)
```

```
pcolor(b, ks, abs(vgt_spec).'), shading interp
ylim([1000 2500])
title('Piano Spectrogram')
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
colormap(hot)
print('-f9','Piano_Spec','-dpng')

%% Part 2: Recorder
clear all, close all, clc
[y,Fs] = audioread('music2.wav');
v = y';
L = length(v)/Fs;
n = length(v);
tr_recorder = (1:length(v))/Fs;
k = (2*pi/(L)) * [0:n/2-1 -n/2:-1];
ks = fftshift(k);
vt = fft(v);

figure(10)
subplot(2,1,1)
plot(tr_recorder,v);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Mary had a little lamb (recorder)');

subplot(2,1,2)
plot(ks,abs(fftshift(vt))/max(abs(vt)))
xlabel('Frequency [\omega]');
ylabel('FFT(v)');
print('-f10','Recorder','-dpng')

p8 = audioplayer(y,Fs); playblocking(p8);

a = 35;
tspan = 0.1;
Recorder_Score = [];
vgt_spec=[];
b=0:tspan:L;
for j=1:length(b)
    g=exp(-a*(tr_recorder-b(j)).^2);
    vg=g.*v;
    vgt=fft(vg);
    [M, I] = max(abs(vgt));
    Recorder_Score = [Recorder_Score; abs(k(I)/(2*pi))];
    vgt_spec=[vgt_spec; abs(fftshift(vgt)) / max(abs(vgt))];
end
```

```
figure(11)
plot(b, Recorder_Score);
xlim([0 14])
title('Recorder: Music Score');
xlabel('Time [sec]');
ylabel('Frequency [Hz]');
print('-f11','Recorder_Score','-dpng')

figure(12)
pcolor(b, ks, abs(vgt_spec).'), shading interp
ylim([4000 8000])
title('Recorder Spectrogram')
xlabel('Time [sec]');
ylabel('Frequency [\omega]');
colormap(hot)
print('-f12','Recorder_Spec','-dpng')
```