Homework 2

Ofek Inbar

Abstract

1 Introduction and Overview

We are given the task of analyzing a short piece of Handel's work by generating different spectrograms and seeing the effect of changing the different parameter options. Then, we are given 2 `.wav` files, recordings of someone playing "Mary had a little lamb" on the piano and recorder, with the task of putting together the score for the pieces.

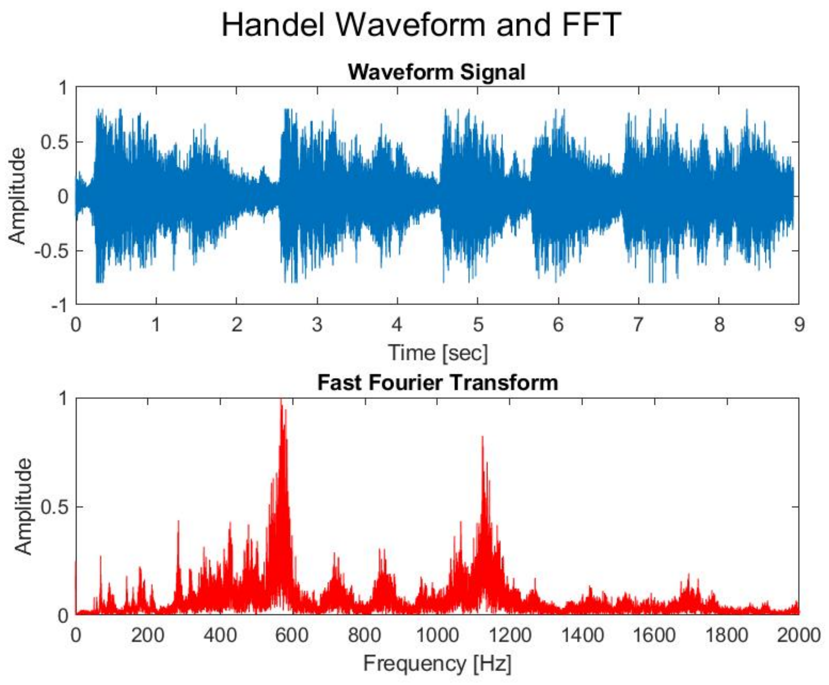

Figure 1: The Waveform and Fast Fourier Transform of the piece by Handel

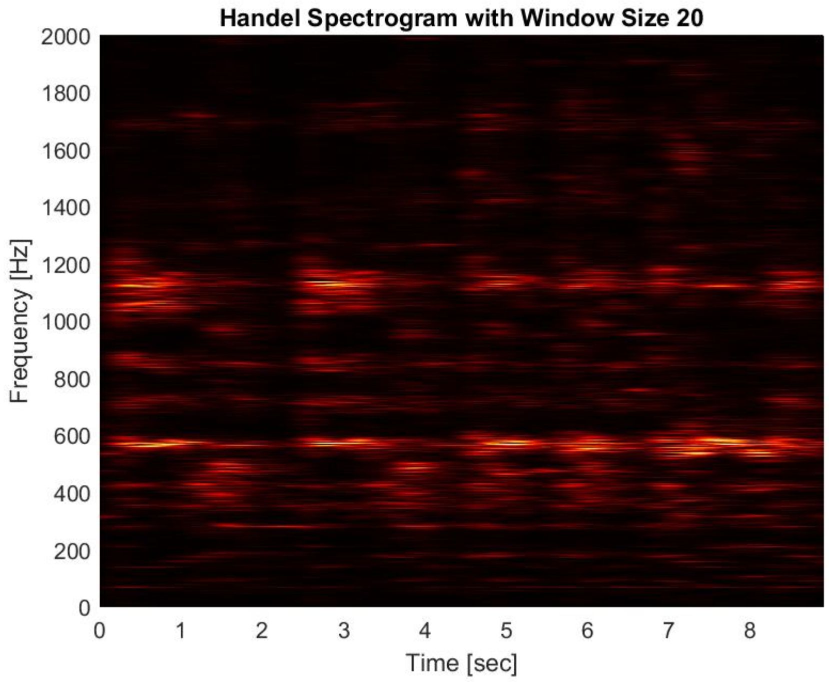

Figure 2: Handel spectrogram with $d\tau = 0.1$ and $\alpha = 20$

2 Theoretical Background

A Fast Fourier Transform is incredibly useful for discovering the component frequencies present in a given signal, but the frequency data comes at the cost of our time data. As is, the FFT can tell us the magnitudes of all of the frequencies present throughout the entirety of the signal, but cannot tell us anything about when they are strong and when they are weak.

One solution to this trade-off is Gabor filtering. The idea is to filter the signal in such a way that values close to a particular time τ remain largely unaffected, but values increasingly distant from τ are close to 0, so as to have little influence on the frequencies present in the FFT. This allows us to include an aspect of time into the Fourier Transform, by strengthening the signal at a particular point in time.

One common filter (and the one we use in this paper) is a Gaussian curve with a mean of τ. The standard deviation varies depending on the decided-upon width of the time-window. Applying this filter means multiplying the original signal's values by the function $e^{-\alpha(t-\tau)^2}$ (t being the time in the signal, τ being the time we want to strengthen, and α being a constant used to determine the width of the window).

This approach, of course, comes with a trade-off in that the narrower the

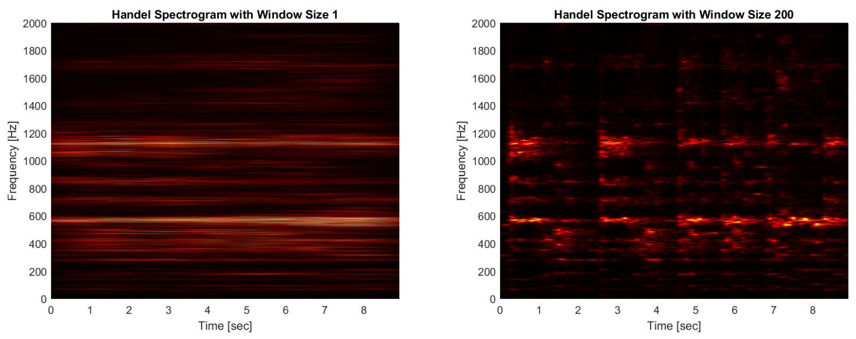

Figure 3: Handel spectrogram with $d\tau = 0.1$ and $\alpha = 1, 200$ (left to right)

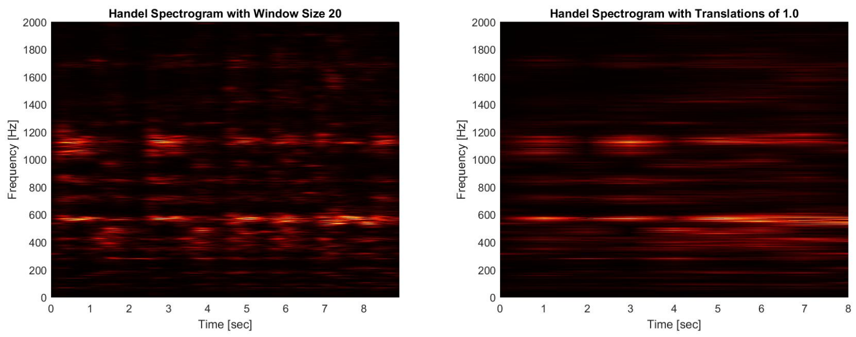

Figure 4: Handel spectrogram with $\alpha = 20$ and $d\tau = 0.1, 1.0$ (left to right)

window of time we choose to strengthen, the less frequency data we are able to get, since the FFT can only give us frequency data on whole number multiples of the fundamental frequency of the signal (one cycle for the entire duration), so the minimum frequency the FFT can pick up is the width of the window. Likewise, if we make the time window wide, we have less information about when in particular the resulting frequencies occur. For this reason, different band widths are used for different purposes.

3 Algorithm Implementation and Development

The algorithm for generating specrograms of the Handel piece is

1. For each value of τ between 0 and the total length of time (translating by $d\tau$) each time:
    (a) Filter the signal by multiplying it by $e^{-\alpha(t-\tau)^2}$.
    (b) Take the Fast Fourier Transform of the signal.

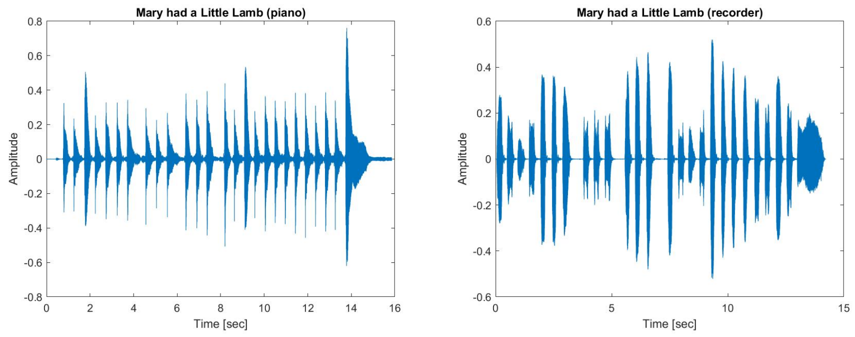

Figure 5: Waveform of music1.wav and music.wav (left to right)

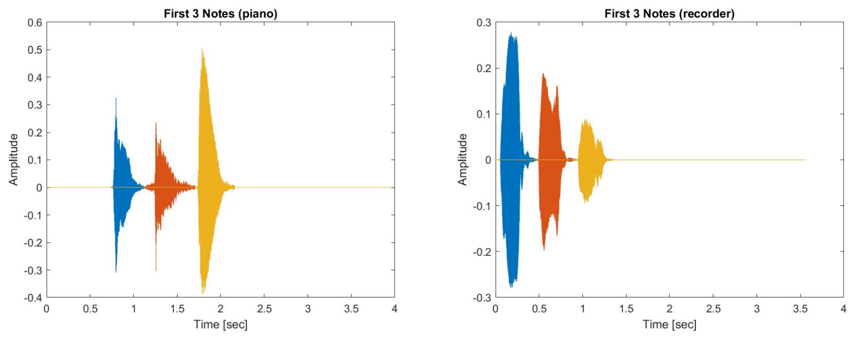

Figure 6: Waveform of the first 3 notes of music1.wav and music.wav (left to right)

(c) Shift the FFT, take the absolute value (to get rid of complex components), and store it in a new row of the spectrogram matrix.

2. Plot the spectrogram.

This was done a number of times with different values of α and $d\tau$.

To analyze the recordings in part 2, first we plotted the waveform and used MATLAB's figure viewer to measure approximate time intervals during which just one note was played. Then, we took the Fourier transform of the signal filtered through a step-function filter (as opposed to a Gaussian). This allowed us to extract the primary frequency associated with that section of the piece, and thereby get the frequency of the note being played. Once we had this it was just a matter of checking which notes were associated with frequencies that were closest to the ones we measured.

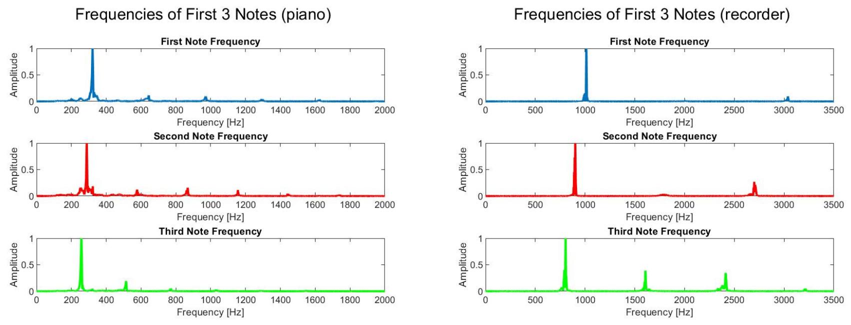

Figure 7: Frequencies of the first 3 notes of music1.wav and music.wav (left to right)

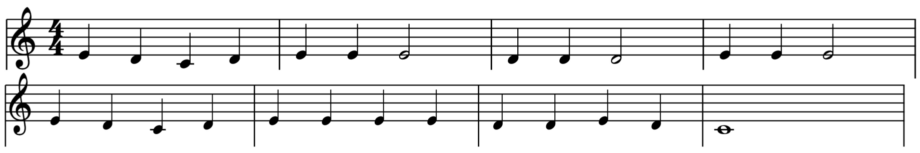

Figure 8: Score for music1.wav

4 Computational Results

The piece by Handel that we were given to analyze is a choral performance with a number of repeating parts (see the Waveform in Figure 1). With $d\tau = 0.1$, we made spectrograms of the signal with $\alpha = 1, 20, 200$ to compare the effects (see Figures 2, 3). With $\alpha = 20$, we made spectrograms with $d\tau = 0.1, 1.0$ (see Figure 4). One thing to notice is that as the α value increases (and thus the window size decreases), we see the lines on the spectrogram get shorter and clearer (more localized), but if we take it too far we start to see them behave erratically, kind of like in a photo that has been artificially sharpened to a large degree. When it comes to $d\tau$, using too large a value can lead to the same long, blurry lines as using too small a value for α (this is called undersampling).

When analyzing the `music1.wav` and `music2.wav` files, we first plotted their amplitude against the time axis (see Figure 5). We can already see from this plot that the piano recording has sharper beginnings on each note played, wheras

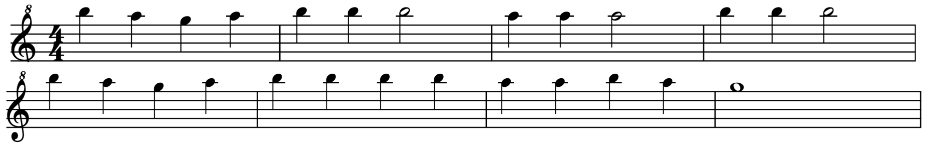

Figure 9: Score for music1.wav

the recorder comes in more gradually. Since the piece being played only has 3 notes, each of which is played once right after the other at the beginning, we filtered out the first 3 notes to analyze (see Figure 6). Taking the Fast Fourier Transforms of these gives us their frequencies.

For the piano piece, we found that the first note has a frequency of 321.6463 Hz, the second has 288.4506 Hz, and the third has 257.7697 Hz. The notes on the keyboard that are closest to these frequencies are E, D, and C in the octave of middle C. The frequencies we found for the recorder piece were 1012.5 Hz, 901.5141 Hz, and 804.2809 Hz, respectively, which most closely correspond to B, A, and G in the octave that is two octaves above middle C. Note that the frequencies recorder notes all have approximately the same ratio with the corresponding piano note (around 3.13). You can see the scores for these pieces in Figures 8 and 9.

	Piano	Recorder
Note 1	321.6463	1012.5
Note 2	288.4506	901.5141
Note 3	257.7697	804.2809

It is interesting to note that, as can be seen in Figure 7, the recorder had more prominent overtones for each note.

5 Summary and Conclusions

The Gabor Transform is useful for identifying the active frequencies at different points in time in a signal. When applied to a piece of music, this can effectively allow us to “reverse-engineer” a recording back into its component notes.

6 Appendix A

We used the standard MATLAB functions for fft manipulation, `fft` and `fftshift`. We also used `pcolor` to create spectrograms.

7 Appendix B

```
%% Part 1 - Handel
clear variables; close all; clc;
load handel

n=length(y); L=n/Fs;
t=(1:n)/Fs;
k=(Fs/n)*[0:(n-1)/2 -(n-1)/2:-1];
ks=fftshift(k);

S = y';
St = fft(S);

% Plot waveform and fft
subplot(2, 1, 1);
plot(t, S);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Waveform Signal');

subplot(2, 1, 2);
plot(ks, abs(fftshift(St))/max(abs(St)), 'r');
set(gca, 'XLim', [0 2e3]);
xlabel('Frequency [Hz]');
ylabel('Amplitude');
title('Fast Fourier Transform');

sgtitle('Handel Waveform and FFT');

saveas(gcf, 'Handel_Waveform_FFT.jpg');
close(gcf);

% Plot spectrogram over varying window sizes
a = [1; 20; 200];
tslide = 0:0.1:L;
Sgt_spec = zeros(length(tslide), n);

for j=1:length(tslide)
    g = exp(-a .* (t - tslide(j)) .^ 2);
    Sg = g .* S;
    for k=1:length(a)
        Sgt = fft(Sg(k, :));
        Sgt_spec(j, :, k) = fftshift(abs(Sgt));
    end
end
```

```
for j=1:length(a)
    f = figure(j);
    pcolor(tslide, ks, Sgt_spec(:, :, j).');
    shading interp;
    set(gca,'Ylim',[0 2e3]);
    colormap(hot);
    xlabel('Time [sec]');
    ylabel('Frequency [Hz]');
    title_text = 'Handel Spectrogram with Window Size';
    title(sprintf('%s %d', title_text, a(j)));
    saveas(f, sprintf('Handel_a=%d.jpg', a(j)));
    close(f);
end

a = 20;
dt = [0.1; 1];
for i=1:length(dt)
    tslide = 0:dt(i):L;
    Sgt_spec = zeros(length(tslide), n);

    for j=1:length(tslide)
        g = exp(-a * (t - tslide(j)) .^ 2);
        Sg = g .* S;
        Sgt = fft(Sg);
        Sgt_spec(j, :) = fftshift(abs(Sgt));
    end

    f = figure(i);
    pcolor(tslide, ks, Sgt_spec(:, :).');
    shading interp;
    set(gca,'Ylim',[0 2e3]);
    colormap(hot);
    xlabel('Time [sec]');
    ylabel('Frequency [Hz]');
    title_text = 'Handel Spectrogram with Translations';
    title(sprintf('%s of %0.1f', title_text, dt(i)));
    saveas(f, sprintf('Handel_dt=%0.1f.jpg', dt(i)));
    close(f);
end

%% Part 2.1 - Mary had a Little Lamb (piano)
clear variables; close all; clc;

[y,Fs] = audioread('music1.wav');
```

```
L = length(y)/4/Fs; n = length(y)/4;
t = (1:n)/Fs;
k=(Fs/n)*[0:n/2-1 -n/2:-1];
ks=fftshift(k);

S = y(1:n).';

f = figure(1);
plot((1:length(y))/Fs, y);
xlabel('Time [sec]');
ylabel('Amplitude');
title('Mary had a Little Lamb (piano)');
saveas(f, 'Piano_Waveform.jpg');
close(f);

% Filter to just first 3 notes

s1 = zeros(1, n);
s1(32784:49745) = 1; % got indices from analyzing the plot

s2 = zeros(1, n);
s2(49657:75279) = 1;

s3 = zeros(1, n);
s3(76073:95301) = 1;

f = figure(2);
plot(t, s1 .* S, t, s2 .* S, t, s3 .* S);
xlabel('Time [sec]');
ylabel('Amplitude');
title('First 3 Notes (piano)');
saveas(f, 'Piano_3_Notes_Waveform.jpg');
close(f);

S1t = fft(s1 .* S);
S2t = fft(s2 .* S);
S3t = fft(s3 .* S);

f = figure(3);
subplot(3,1,1);
plot(ks, abs(fftshift(S1t))/max(abs(S1t)), 'LineWidth', 2);
axis([0 2e3 0 1]);
xlabel('Frequency [Hz]');
ylabel('Amplitude');
title('First Note Frequency');
subplot(3,1,2);
```

```
plot(ks, abs(fftshift(S2t))/max(abs(S2t)), ‘r’, ‘LineWidth’, 2);
axis([0 2e3 0 1]);
xlabel(‘Frequency [Hz]’);
ylabel(‘Amplitude’);
title(‘Second Note Frequency’);
subplot(3,1,3);
plot(ks, abs(fftshift(S3t))/max(abs(S3t)), ‘g’, ‘LineWidth’, 2);
axis([0 2e3 0 1]);
xlabel(‘Frequency [Hz]’);
ylabel(‘Amplitude’);
title(‘Third Note Frequency’);
sgtitle(‘Frequencies of First 3 Notes (piano)’);
saveas(f, ‘Piano_Frequencies.jpg’);
close(f);

[~, s1_max_idx] = max(abs(S1t)); s1_freq = k(s1_max_idx)
[~, s2_max_idx] = max(abs(S2t)); s2_freq = k(s2_max_idx)
[~, s3_max_idx] = max(abs(S3t)); s3_freq = k(s3_max_idx)

%% Part 2.2 Mary had a Little Lamb (recorder)
clear variables; close all; clc;

[y,Fs] = audioread(‘music2.wav’);

L = length(y)/4/Fs; n = length(y)/4;
t = (1:n)/Fs;
k=(Fs/n)*[0:n/2-1 -n/2:-1];
ks=fftshift(k);

S = y(1:n).’;

f = figure(1);
plot((1:length(y))/Fs, y);
xlabel(‘Time [sec]’);
ylabel(‘Amplitude’);
title(‘Mary had a Little Lamb (recorder)’);
saveas(f, ‘Recorder_Waveform.jpg’);
close(f);

% Filter to just first 3 notes

s1 = zeros(1, n);
s1(1621:20877) = 1; % got indices from analyzing the plot

s2 = zeros(1, n);
s2(20877:40815) = 1;
```

```
s3 = zeros(1, n);
s3(40815:59359) = 1;

f = figure(2);
plot(t, s1 .* S, t, s2 .* S, t, s3 .* S);
xlabel('Time [sec]');
ylabel('Amplitude');
title('First 3 Notes (recorder)');
saveas(f, 'Recorder_3_Notes_Waveform.jpg');
close(f);

S1t = fft(s1 .* S);
S2t = fft(s2 .* S);
S3t = fft(s3 .* S);

f = figure(3);
subplot(3,1,1);
plot(ks, abs(fftshift(S1t))/max(abs(S1t)), 'LineWidth', 2);
axis([0 3.5e3 0 1]);
xlabel('Frequency [Hz]');
ylabel('Amplitude');
title('First Note Frequency');
subplot(3,1,2);
plot(ks, abs(fftshift(S2t))/max(abs(S2t)), 'r', 'LineWidth', 2);
axis([0 3.5e3 0 1]);
xlabel('Frequency [Hz]');
ylabel('Amplitude');
title('Second Note Frequency');
subplot(3,1,3);
plot(ks, abs(fftshift(S3t))/max(abs(S3t)), 'g', 'LineWidth', 2);
axis([0 3.5e3 0 1]);
xlabel('Frequency [Hz]');
ylabel('Amplitude');
title('Third Note Frequency');
sgtitle('Frequencies of First 3 Notes (recorder)');
saveas(f, 'Recorder_Frequencies.jpg');
close(f);

[s1_max, s1_max_idx] = max(abs(S1t)); s1_freq = k(s1_max_idx)
[s2_max, s2_max_idx] = max(abs(S2t)); s2_freq = k(s2_max_idx)
[s3_max, s3_max_idx] = max(abs(S3t)); s3_freq = k(s3_max_idx)
```