Handling Handel & Other Overtures with Spectrograms

Avery Pong – AMATH 482 – 02.12.2019

0. Overview: Musical Texture through Spectrograms

Handel's Messiah and Mary Had a Little Lamb were decomposed using a Gabor Windows to extract time-series of the piece's frequency components. Three different windows—a Gaussian, a boxier (nearly) stepwise Gaussian function, and a Mexican hat filter—were used to properly frame the variable frequency content of the Handel recording with respect to time. Mary Had a Little Lamb recorders were analyzed to verify each of the notes and to elucidate the overtone-based origins of the timbre associated with each instrument. A thorough discussion of the musical origins of timbre and tonal texture follows to distinguish piano from recorder.

I. Introduction: Messiah versus Mary

Music is pitch with rhythm. In other words, the pressure waves that result from a musical piece carry variable frequency information at carefully divided temporal intervals, bringing to life pieces of music that are both rhythmically compelling with pleasant frequency components. MatLab's Messiah sample is an amalgamation of multiple singer's voices and musical instruments across an 8.92 second time interval. Different pitches are played simultaneously with instruments of varying timbre to render a rich, complex sound. This richness and complexity is reflected in the very busy spectrum that the piece produces, as each beat contains a diverse array of frequency contents. Similarly, the diverse frequency content is apparent from the public domain score, where, upon reaching the chorus, seven notes each played by multiple musicians, sound simultaneously.

Fig 1. Handel's Messiah: orchestral scores are musical spectrograms.

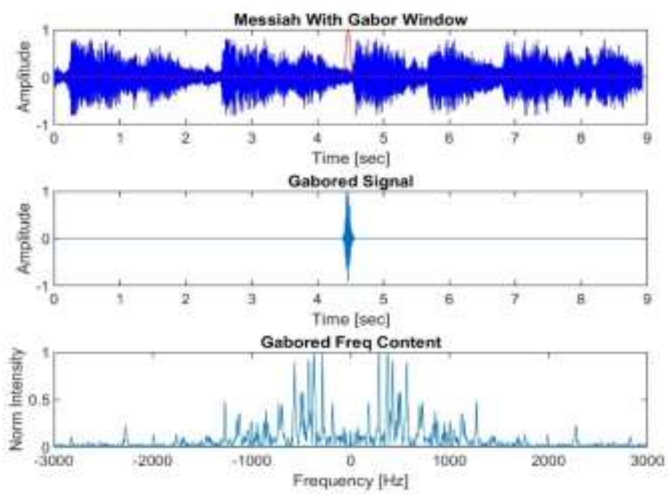
The recordings of Mary Had a Little Lamb (*Mary*, from here) are quite different: instead of being an artistic composite of multiple voices played to the tune of a complex harmonic orchestral backdrop, Mary Had a Little Lamb is comprised of a one-line melody. That is, only one fundamental frequency is being played at any given time. While this is musically disappointing, the Mary affords the opportunity to analyze the overtone (harmonic) structure of each note produced by both the piano and recorder. The overtone series of a fundamental frequency are integer multiple frequency resonances that sound every time a note is played on an instrument. This means that every note on the piano or record is not just the frequency (base note) that is being played, but a sum of the vibrations of the fundamental and overtone frequencies that sound upon exciting the instrument.

II. Theoretical Background: the Short-Time FT

We can extract frequency information from the time domain of the recording by using the Fast Fourier Transform (FFT). However, applying the FFT alone will only yield the total frequency content of the piece, the summed aggregate of all of its many, many notes. This would give a muddled image of what the piece's frequency content looks like with respect to time. It thus becomes important to understand the frequencies in the order in which they came so that we can understand the temporal component of the music. This can be accomplished by applying a sliding Gabor window to our FFT analysis, allowing us to localize a signal of interest in time and level out other signals that might interfere with the FFT analysis.

$$G[f](t,\omega) = \int_{-\infty}^{\infty} f(\tau)\cdot g(\tau - t)\, e^{-i\omega t} d\tau$$

Here, f represents the function of the signal, g represents the function that describes the particular window applied, τ represents time-axis displacement, and ω describes the window width. Most of the following analysis uses Gaussians to extract time-localized frequencies so that signals of interest can be identified while other signals are zeroed out.



$Fig\ 2. Gabor\ Window\ yields\ time - localized\ frequency\ content.$

Sliding the Gabor window developed above renders a clearer, time-resolved picture of the frequencies of the piece. $Fig\ 2.$ visually demonstrates how a small slice of the signal can be extracted via the window and Fourier Transformed to examine its frequency content. Taking the Fourier Transform of each of these windowed signals yields a spectrogram, which resolves the music in both time and frequency. This process was optimized to generate spectra for both the Messiah and Mary.

A major challenge presented by the Short-Time Fourier Transform is the fact that increased resolution in time always costs resolution in frequency as a result of the Heinsenburg relationship. This trade off results in two cases, one in which the recording can be under-sampled by using large windows. While this can capture a wide variety of wavelengths and therefore frequency content, it lowers resolution in time. The second case arises from the use of time-windows that are too small, resulting in over-sampling where longer wavelengths are omitted due to the small window size.

III. Algorithmic Implementation and Development

MatLab's FFT function was used for analysis. A few preliminaries require clarification. MatLab's FFT function, as a result of the FFT algorithm implementation, shifts data to the edge of the domain in which it operates. That is, all data *x* on a *[0 , L]* domain are shifted to a *[-L , 0]* domain, while all data *x* on a *[-L, 0]* domain are shifted to a *[0, L]* domain. Every other mode defined by this data resolution of *L* is also scaled to a 2π domain, which has the consequence of requiring the wavenumbers associated with the data to similarly be scaled by a factor that is dependent on both *L* and 2π. Lastly, MatLab's FFT multiplies every other mode by *-1*, requiring us to take absolute values of our transformed data when appropriate. These precautions were taken during scripting.

Handel's Messiah was recorded as a 73113 component vector the needed to be formatted in such a way to allow for the application of the FFT: the number of Fourier modes was the number of vector components recorded to reconstruct and store the piece; the domain length was defined as the number of measurements in the 73113 component long vector `v` divided by sample rate `Fs`. Having rendered the recording in the time domain, a number of Gabor Filters were developed to resolve the recording's frequencies with respect to time. Filters of the following form were used:

$$Gaussian\ Gabor = g(t) = \exp(-at^2)$$

$$Boxy\ (Step)\ Gabor = g(t) = \exp\exp(-at^{10})$$

$$Mexican\ Hat = g(t) = \left(1 - \frac{t^2}{b}\right)\exp(\frac{-t^2}{c})$$

Constant coefficient *a* was used to control the width of the window—where a larger *a* corresponds to a narrower window. Constant *b* was used to control the depth of the Mexican Hat function's satellite peaks, while *c* was used to control the width of the window. A larger *b* increases satellite concavity (depth) and a larger *c* corresponds to a wider window. Visual examples and code for each of these windows can be found in *Appendix D*.

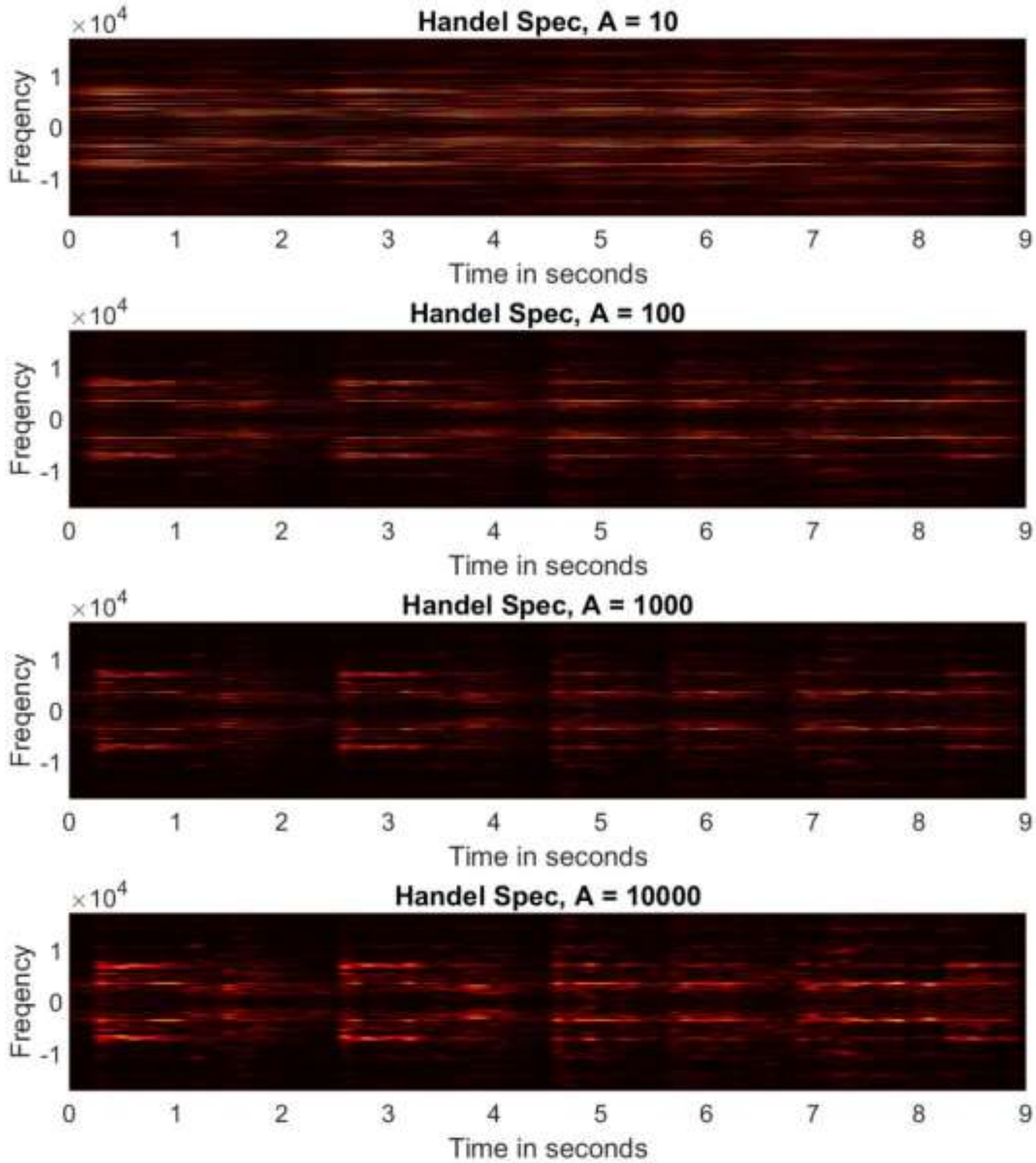
Each Gabor window was applied to the domain in increments of one-tenth of a second. *Appendix B* shows that taking the FFT of these snippets of signal renders a vector of frequency content, which was accordingly stored in `vgt_spec` to allow for the latter rendering of the spectrogram. The `pcolor` function allowed for the two-dimensional visualization of the data.

A similar approach was developed for Mary Had a Little Lamb. A Gaussian Gabor window was applied to the signal along the time domain at increments that allowed for sufficient frequency content to be captured. Again, a `vgt_spec` matrix was developed from each time-step to render the spectrogram issuing the `pcolor` function. At this juncture, it became necessary to divide the wavenumber `k` by 2π because we were aiming to elucidate the pitches rendered from each recorded note. This important step essentially rescaled the scaling step required for proper FTT functioning.

While the spectrogram allowed for an approximation of the frequency (hz) of the piece, a more rigorous approach was taken to deliver more precise pitch measurements. To accomplish this, the same process of applying a Gabor Window to the signal was taken, but in time-domain increments that matched the beat of the music. This allowed each time slice to capture the frequency of approximately one note. Upon taking the FFT of each time slice, the index of the maximally intense frequency was taken. Importantly, this step was taken so that the fundamental frequency could be identified—simply taking the `max()` of the frequency vector would return the highest frequency, which was typically a high pitch overtone. Finally, with all the precautions out of the way, the maximum frequency was identified by finding the `k` value that corresponded to the index of this most intense peak. These values were identified per time slice, and were collected in vector `maxfreq` to ease the process of reporting values and to ease plotting.

IV. Computational and Visual Results, Handel

The Messiah was analyzed with the Gabor filters described in section III. The Mexican Hat analysis, while shown in the code in *Appendix B,* was omitted for space because its results are similar to what I call the 'boxy' Gaussian.



$Fig\ 3.\ Sectrograms\ with\ decreasing\ window\ size; increasing\ A\ decreases\ window\ width.$

The top graph in *Fig 3.* shows that undersampling—keeping the windows large—can indeed capture all the frequencies associated with the piece, almost to the resolution of the image-processing software. However, there is limited resolution in time because of the course overlap of the windows across the time steps. The bottom figure, which had the greatest coefficient and thus the narrowest window width, shows the best resolution in time, but shows poor frequency resolution. This is demonstrative of the Heinsenburg relationship, showing that decreasing window widths indeed improve resolution in time with costs to frequency resolution.

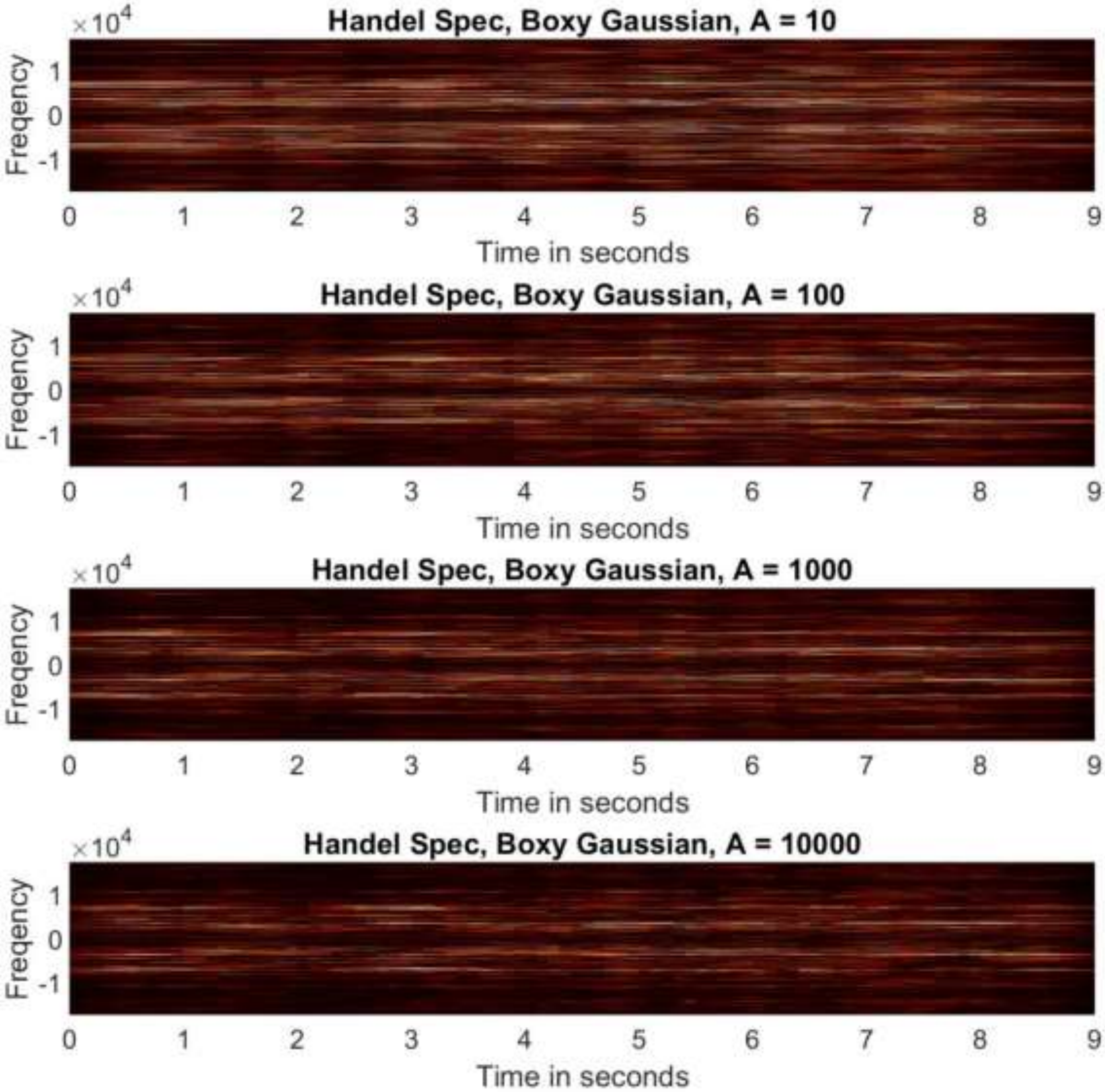


Fig. 4 shows the spectrograms rendered from the pseudo − stepwise function.

Fig 4. Shows that the broader 'boxy' Gaussian function maintains frequency resolution. This is because the broad shoulders at the points of concavity capture more frequency content per time step. Because of this breadth—as opposed to the tapering nature of the Gaussian—the boxy function tends to capture more frequency content. Even large *A* coefficients are not sufficient to properly render the signal in time. Comparisons between Fig. 3 and Fig. 4 show the values and drawbacks of oversampling and undersampling. They also serve to verify the Heinsenburg relationship as it applies to the transform used in this analysis: the more we know about the frequency, the less certain we are about the time in which those frequencies occurred. Likewise, the more we know about time-localization, the less we know about the exact frequencies that occur at that time.

V. Computational and Visual Results, Mary Had a Little Lamb

Mary Had a Little Lamb, recorded on both the piano and recorder. The frequencies and corresponding pitches for each can be found in *Appendix E,* where there is a thorough discussion about the challenges of discriminating singly played notes and sustained notes. This section, instead, focuses more on the spectrograms rendered from each recording.

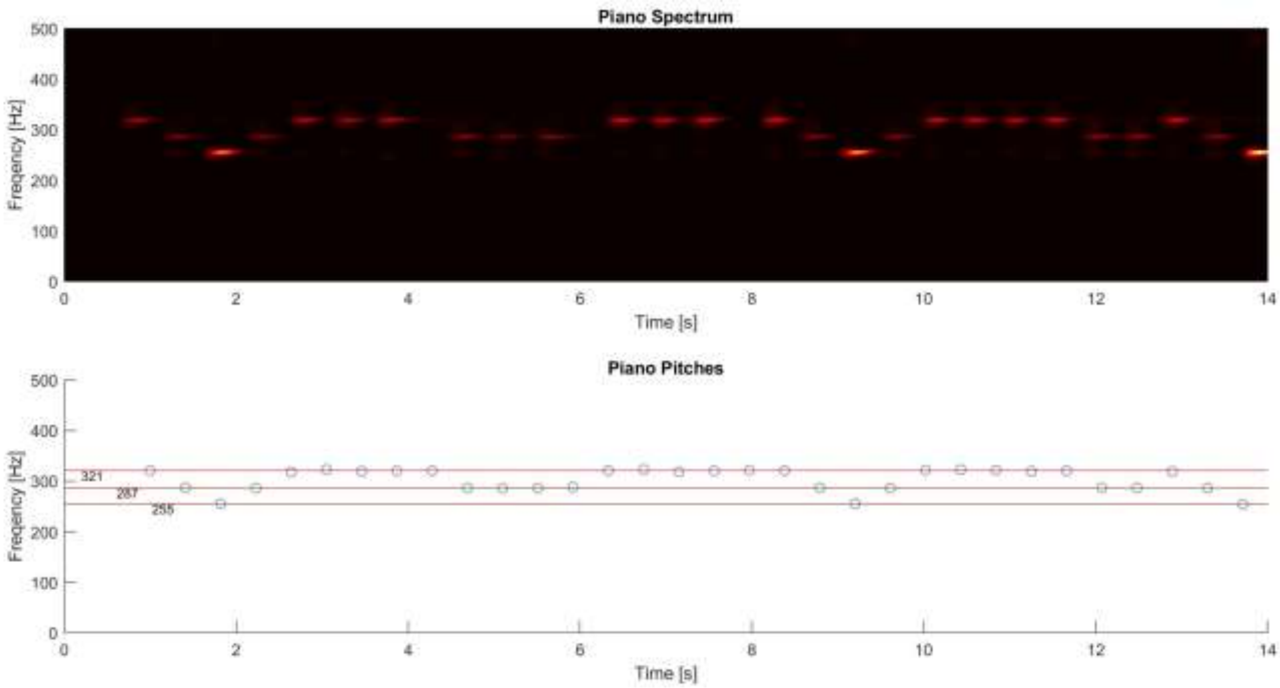


Fig 5. *Spectrogram and notes for Mary Had a Little Lamb on piano.*

Figure 5 shows fundamental frequencies of the piano notes, with higher frequencies filtered out. This required both the adjustment of the `pcolor` map color to maximize contrast and the use of a filter width that was best suited to capture frequency content in the 200 – 400 Hz range. Red lines mark the three approximate pitches (255, 287, 321 Hz) that are played throughout the piece, with frequency labels at left. These notes correspond to middle C, D and E on the piano.

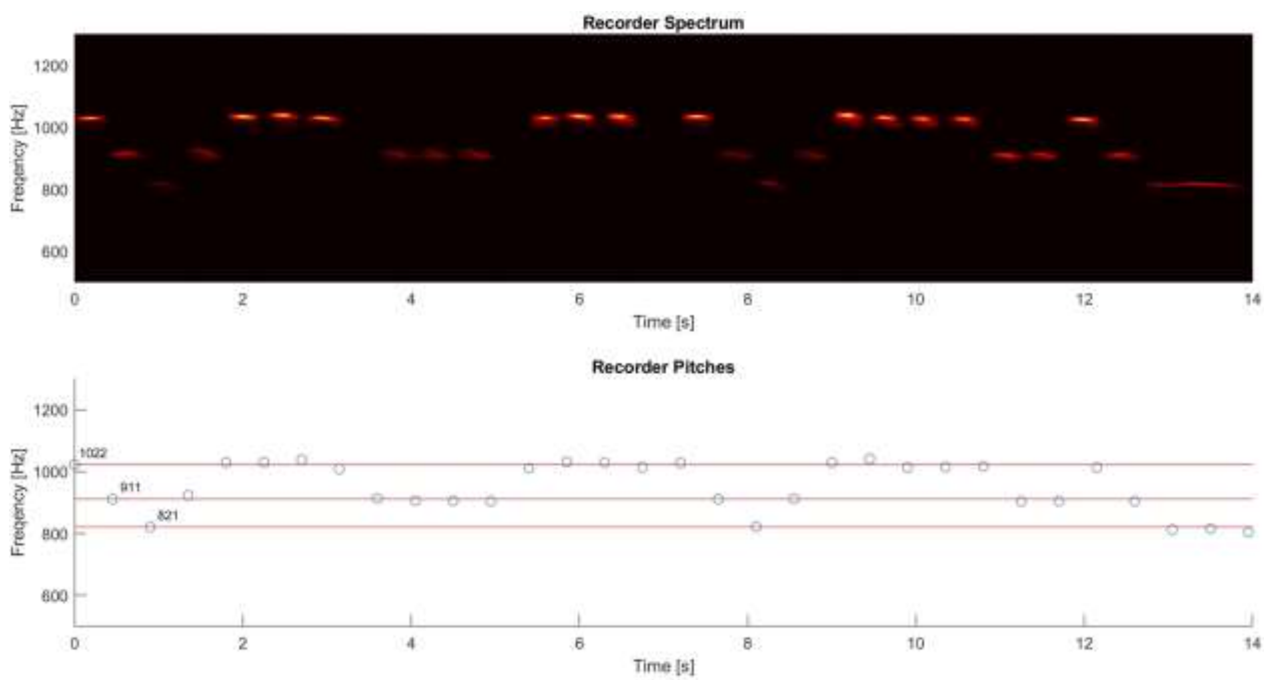


Fig 6. *Spectrogram and notes for Mary Had a Little Lamb on Recorder.*

Figure 6 shows fundamental frequencies for the recorder's rendition of Mary Had a Little Lamb. Red lines mark the three pitches (821, 911, 1022 Hz) that are played throughout the piece, with frequency labels at left. These pitches correspond to A flat, B flat, and C.

Three important musical observations can be made here. First, the sequences of notes between the piano and recorder remain the same, though the absolute frequencies of their pitches differ. This indicates that the piece was played in different keys in different registers. Hence, the recorder recording sounds higher in pitch than does the piano. Thus, the overall shape of the spectrograms look very similar. Second, in direct reference to the frequencies reported in *Appendix E,* it is clear that the piano is out of tune. None of the frequencies truly correspond to pure tone C, D, or E notes. They all miss the mark by about 7 Hz. Because the notes are consistently less than the frequencies they should be according to the piano's tuning, we refer to the piano as out of tune. Specifically, this piano is *flat,* likely because of a long period of use without having been retuned. Third, also in reference to the tables in *Appendix E*, the variance of recorder notes from their first sounded pitch is greater than the piano. That's to say that the piano's notes are more consistent than are the recorders. This is because the piano has fixed tuning of the strings, where the player is not in direct control of the frequencies that sound from hitting the piano's keys. The recorder's pitches, however, are completely dependent on the depth of the breath of the player, and is thus prone to greater inconsistency in the pitches it sounds. These noted inconsistencies between players is the reason why musical ensembles tend to tune their string instruments to the wind players: wind instruments are variable based on the player and on playing conditions more so than string instruments, and so the string players typically defer to the winds.

One last interesting discussion concerns instrument timbre. Timbre is usually an intangible quality that instrumentalist speak of that differentiates the tonal quality of a note played on a particular instrument. This timbre, in fact, has a very concrete source: the overtone series of a note played on a particular instrument. Analysis of the spectrograms of both instruments illuminate the contributions of frequency content of the overtones to each note played.

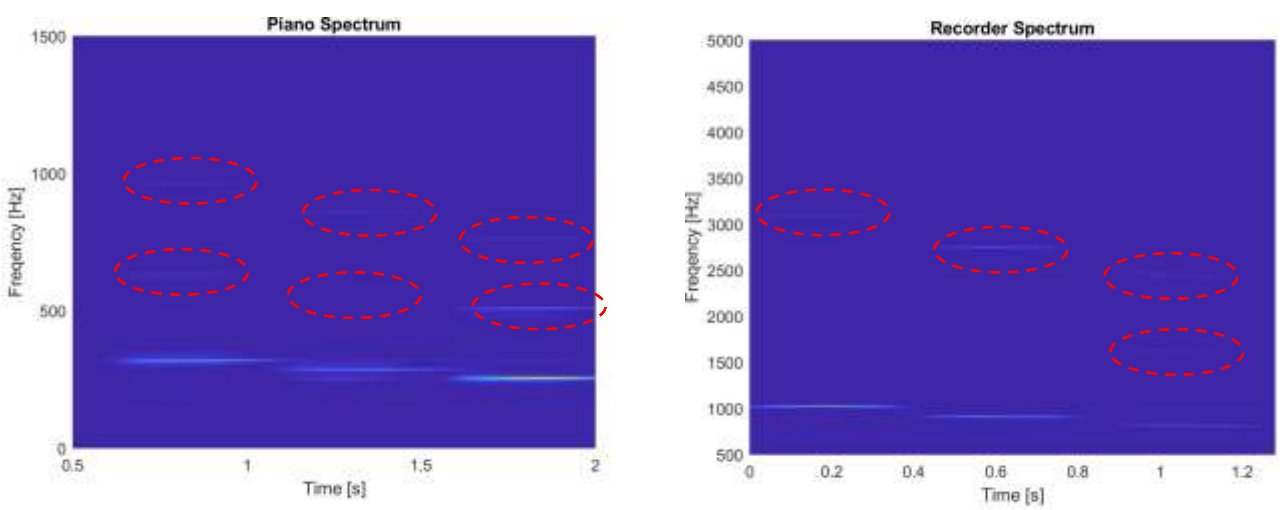


Fig 7. Overtone series for the piano is richer than the recorder, observable overtones in red.

Looking at the frequency content of the first three notes of both the piano and recorder spectrum shows that the piano's notes are far richer in low frequency overtone contributes. This gives the piano its relatively rich, complex sound: when you play one note on the keyboard, many more sound. Because of the relatively low pitch of these notes, the piano's notes tend to produce overtones that are in the more soothing, lower end of the audible spectrum. The recorder's notes, on the other hand, have relatively damp, sparse overtone series that accompany each fundamental frequency. Moreover, the overtone contributes are high enough in pitch to render themselves unpleasant to the ear. Overall, this analysis demonstrates why the piano has a more complex, rich timbre than the piercing notes of the recorder. No wonder more people know how to play the piano, even though they're 1000 times more expensive than even a reasonably good recorder.

IV. Conclusions

Handel's Messiah and Mary Had a Little Lamb were analyzed using Gabor filters to better understand the temporal dynamics of their frequency contents. Handel's piece, played by multiple instruments, gave a rich portrait of frequencies with which to work, and afforded a great example to parse the Heinsenburg relationship: it demonstrated that spectrograms are fundamentally a compromise between good frequency and temporal resolution of a signal. Mary Had a Little Lamb only recorded a single melody, but aptly demonstrated the value of the overtone series in contributing to the overall timbre of the instruments. Richer overtone series in the author's opinion tend to make the individual notes sounds more complex—this is reasonable, since a better developed overtone series essentially gives the instrument extra notes.

Appendix A: MatLab functions

1) `Fft(data)` – This allowed for a multidimensional application of the fast fourier transform to the `v` data. This extracted the frequency content from the spatial data I originally had.
2) `fftshift(data)` – These functions reversed the effect of using the FFT function, which shifts the domain one which it works. This fftshift counteracts the shift, and all data must be correspondingly shifted in the same way, in the same domain in order to work together.
3) `Max(vector)` – This found the maximum value of the vector or data in question.
4) `ind2sub(dim, index)` – This allowed for easier indexing of our data, specifically when I tried to find the indices of the 3D U data.
5) `Pcolor(matrix)` – Produces a checkerboard, 2D display of the input matrix.

Appendix B: Handel Code

```
clear all; close all; clc

load handel                  % Load data
v = y';                      % Plotting, v this is the number of pressure readings
n = length(v);               % Fourier modes, number of measurements from music
L = n/Fs;                    % Measurement/(Measurement/second) = seconds
t2 = linspace(0, L, n);
t = t2(1:n);
k = (2*pi/L)*[0:(n/2-1) -n/2:-1]; % freq comp for fft.
% 2pi/L scales domain of 10 to domain of 2pi
ks = fftshift(k);

kplot = max(k);
ksplot = max(ks);

% Gabor Filter
a = [.1 .3 .5 .7];                   % smaller a = smaller window
vgt_spec = [];                       % store freq data

tslide = 0:0.1:L;

for i = 1:4
    for j = 1:length(tslide)
        % g = exp(-a(i).*(t-tslide(j)).^2); % Gabor, Gaussian
        % g = exp(-a(i).*(t-tslide(j)).^10); % Gabor, step
        g = (1-((t-tslide(j))/a(i)).^2).*exp(-((t-tslide(j))/a(i)).^2); % Mexican Hat
        vg = v.*g;
        vgt = fft(vg);
        vgt = vgt(1:(end-1));    % make even number points
```

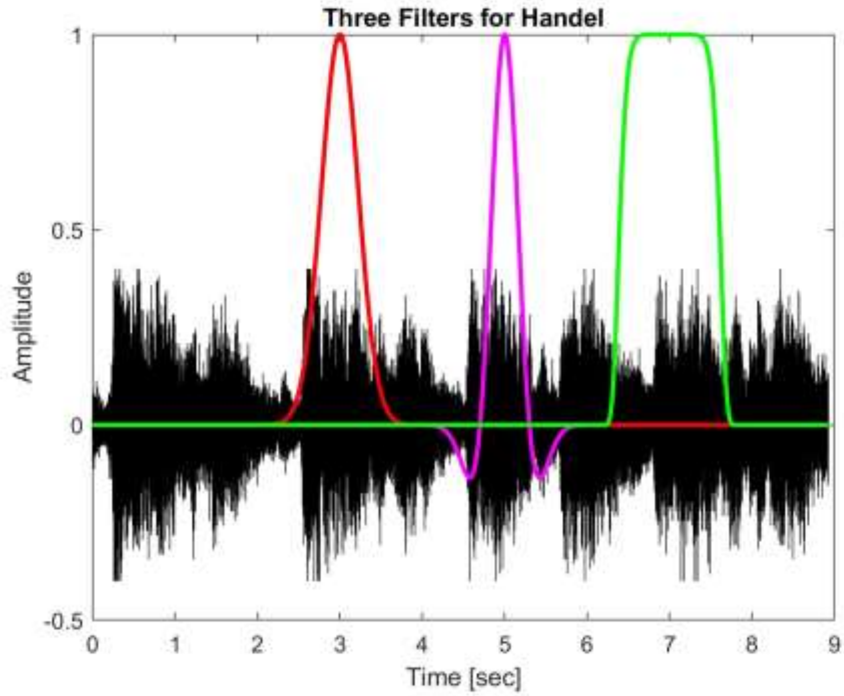
```
        vgt_spec = [vgt_spec; fftshift(abs(vgt))];
    end
end

% Prep plotting for spectrograms
graph1 = vgt_spec(1:91, :);
graph2 = vgt_spec(92:182, :);
graph3 = vgt_spec(183:273, :);
graph4 = vgt_spec(274:end, :);

graph1 = graph1.';
graph2 = graph2.';
graph3 = graph3.';
graph4 = graph4.';

% Figure 1 Spectrogram
figure(1)
title('Handel Spec, Mexican Hat')

subplot(4,1,1)
pcolor(tslide, ks, graph1), shading interp
set(gca, 'Ylim', [-ksplot*(2/3) ksplot*(2/3)])
title('Handel Spec, Mexican Hat, A = 0.1');
xlabel('Time in seconds')
ylabel('Freqency')
colormap(hot);

subplot(4,1,2)
pcolor(tslide, ks, graph2), shading interp
set(gca, 'Ylim', [-ksplot*(2/3) ksplot*(2/3)])
title('Handel Spec, Mexican Hat, A = 0.3');
xlabel('Time in seconds')
ylabel('Freqency')

subplot(4,1,3)
pcolor(tslide, ks, graph3), shading interp
set(gca, 'Ylim', [-ksplot*(2/3) ksplot*(2/3)])
title('Handel Spec, Mexican Hat, A = 0.5');
xlabel('Time in seconds')
ylabel('Freqency')

subplot(4,1,4)
pcolor(tslide, ks, graph4), shading interp
set(gca, 'Ylim', [-ksplot*(2/3) ksplot*(2/3)])
title('Handel Spec, Mexican Hat, A = 0.7');
xlabel('Time in seconds')
ylabel('Freqency')
```

Appendix C: Mary Had a Little Lamb Code

```
clear all; close all; clc;

tr_rec=14; % record time in seconds
y=audioread('music2.wav'); Fs=length(y)/tr_rec;

v = y';                              % make row vector
n = length(v);
L = n/Fs;                            % Measurement/(Measurement/second) = seconds
t2 = linspace(0, L, n);              % n+1 because first point same as last
t = t2(1:n);
k = (2*pi/L)*[0:(n/2-1) -n/2:-1];    % freq comp for fft.
% 2pi/L scales domain of 10 to domain of 2pi
ks = fftshift(k); % just keep all unshifted.
```

```matlab
a = 12000; % smaller a, wider window

vgt_spec  = [];
maxfreq = [];
tslide1 = 0:0.1:14;                 % resolution required for spectrogram rendering
tslide2 = 0:0.45:14;                % resolution required to match rhythmic beat of piece

% Make Spectrogram
for i = 1:length(tslide1)
    g = exp(-a.*(t-tslide1(i)).^4); % plotting gabor requires you make domain same for
                                      both sig and function
    vg = v.*g;                      % multiply
    vgt = fft(vg);                  % take fft of windowed signal
    vgt_spec = [vgt_spec; fftshift(abs(vgt))];  % store vgt in matrix for spectrogram
end

% Make values for finding exact frequency
for j = 1:length(tslide2)
    g = exp(-a.*(t-tslide2(j)).^4); % plotting gabor requires you make domain same for
                                      both sig and function
    vg = v.*g;                      % multiply
    vgt = fft(vg);                  % take fft of windowed signal
    [m, ind] = max(abs(vgt));       % find index of max int freq component
    [x_ind, y_ind] = ind2sub([n,n],ind); % find index
    maxfreq = [maxfreq k(x_ind)];   % apply first component to k
                                      to find frequency the index corresponds to
end

subplot(2,1,1)
vgt_spec = vgt_spec.';
pcolor(tslide1, ks./(2*pi), vgt_spec), shading interp   % reconvert to freq in hz
set(gca, 'Ylim', [500 1300])                            % set y axis
title('Recorder Spectrum');
xlabel('Time [s]')
ylabel('Freqency [Hz]')
colormap(hot);

subplot(2,1,2)
maxfreq = maxfreq./(2*pi);                              % plot max frequency in hertz
scatter(tslide2, maxfreq);                              % display scatter plot of
                                                          recorded freq
a = round(maxfreq(1:3).');                              % map lines for three pitches
b = num2str(a);                                         % convert to str for text label
c = cellstr(b);                                         % make text
text(tslide2(1:3) + .05, maxfreq(1:3) + 40, c, 'Fontsize', [8]);
title('Recorder Pitches')
xlabel('Time [s]')
ylabel('Freqency [Hz]')
ylim([500 1300]);

hold on

% Plots the three lines corresponding to the distinct pitches.
x = linspace(0, 14, length(tslide2));                   % define domain
y = ones(length(tslide2));

y1 = maxfreq(1).*y;
y2 = maxfreq(2).*y;
y3 = maxfreq(3).*y;

plot(x, y1, 'r', 'linewidth', [.5])                     % plots first freq in hz
plot(x, y2, 'r', 'linewidth', [.5])                     % plots second freq in hz
plot(x, y3, 'r', 'linewidth', [.5])                     % plots third freq in hz
```

Appendix D: Gabor Windows



Three forms of windows were used. The red corresponds to the Gaussian window, the magenta corresponds to the Mexican Hat wavelet used, and the green corresponds to the pseudo-step function used for the analysis.

```
clear all; close all; clc

load handel
v = (y'/2);                   % Plotting, v this is the number of pressure readings
n = length(v);                % Fourier modes, number of measurements from music
L = n/Fs;                     % Measurement/(Measurement/second) = seconds
t2 = linspace(0, L, n);      % n+1 because first point same as last
t = t2(1:n);

Gabor windows
h = exp(-10.*(t-3).^2);
g = (1-((t-5)/.3).^2).*exp(-((t-5)/.3).^2);
k = exp(-100.*(t-7).^10);

plot(t, v, 'k')
hold on
plot( t, g, 'm', t, h, 'r', t, k, 'g', 'linewidth', [2]);
xlim([0 9]); ylim([-.5 1]);
xlabel('Time [sec]'); ylabel('Amplitude'); title('Three Filters for Handel');
```

Appendix E

Reported are the pitches of the fundamental frequency that sounds during the Piano recording, in the order in which they appear. Because the piece was played on an out of tune piano—the piano's pitches are flat, or of lower overall frequency due to use—the pitches do not exactly correspond to the values outlined on the HW2 documentation. Additionally, because the rhythm with which the piece was played was not exactly in tempo—or of even musical increments—the time stepping scheme could not exactly capture each note on its beat. For this reason, it was difficult to capture only true notes, instead of sustained–held–notes. The discrimination between sustained notes and singly played notes was difficult, and thus done by ear. Red notes correspond to these sustained pitches.

Piano Data	
F, Hz	Note
321	E
287	D
255	C
286	D
318	E
323	E
319	E
320	E
320	E
287	D
286	D
286	D
288	D
321	E
323	E
318	E
320	E
322	E
321	E
287	D
255	C
286	D
322	E
323	E
321	E
319	E
287	D
287	D
319	E
286	D
254	C

Recorder Data	
F in Hz	Note
1022	C
911	B flat
821	A flat
923	B flat
1028	C
1030	C
1038	C
1008	C
913	B flat
906	B flat
906	B flat
904	B flat
1011	C
1031	C
1029	C
1014	C
1028	C
911	B flat
821	A flat
912	B flat
1029	C
1041	C
1013	C
1015	C
904	B flat
904	B flat
1013	C
904	B flat
813	A flat