AMATH 582 Homework 2

Pavel Zelinsky

Github: <https://github.com/palpalych/SpectrogramFun.git>

**Abstract**

I am analyzing the frequencies of several different musical pieces via looking at their spectrograms. Additionally I extract the music score for an unknown piece

## Introduction and Overview

I am interested in viewing the structures of musical pieces and see what information can be extracted. One difficulty is that simply taking a Fourier Transform of the whole piece provides good information about frequencies present over the entire piece, but doesn’t portray specific details about the piece. By applying time frequency analysis and plotting spectrograms we can hone in on details within the piece

## Theoretical Background

By applying a filter, for example a Gabor filter, that is centered around a particular time in the piece we can move the center of the filter across the whole time range and get more detailed information about the smaller frequencies while losing information about more global frequencies. The Gabor filter is a Gaussian, where s is the scaling factor that controls how wide the filter is and c is the time in the piece of interest to us:

Other filters to consider are the Mexican Hat Filter:

And the Step Filter:

Plotting the Fourier transform as a function of moving the center of the filter across the full time range gives us a spectrogram. By varying the width of the filter we can vary the detail level of the frequencies that we are trying to extract. Finally, once this analysis has been done, if certain frequencies show to be of interest we can apply a Gaussian filter on the Fourier transform of the dataset to hone in on particular structures, and potentially extract specific notes from the music

## Algorithm Implementation and Development

Although it is less efficient, I broke up the code into functions for the different filters, and then wrote a Specto function to calculate the spectrogram with any given filter. Throughout this I never used the Matlab built in spectrogram function as the desire was to have finer tuned control over what I’m extracting from the dataset. One important note for the calculations – the ks were not wave numbers but instead frequencies, so they had to be scaled by 1/L due to the relationship between wave number and frequencies being w=2π

## Computational Results

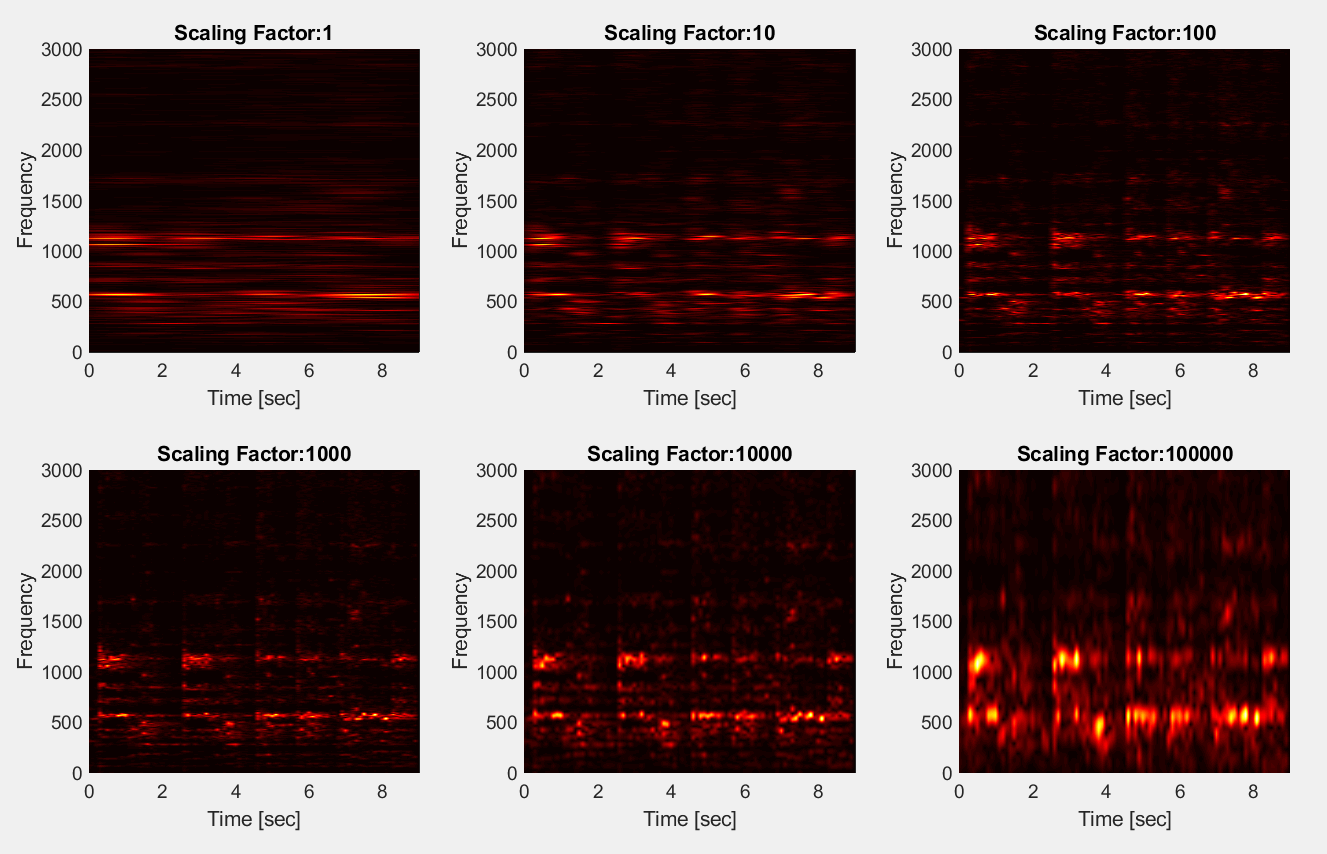
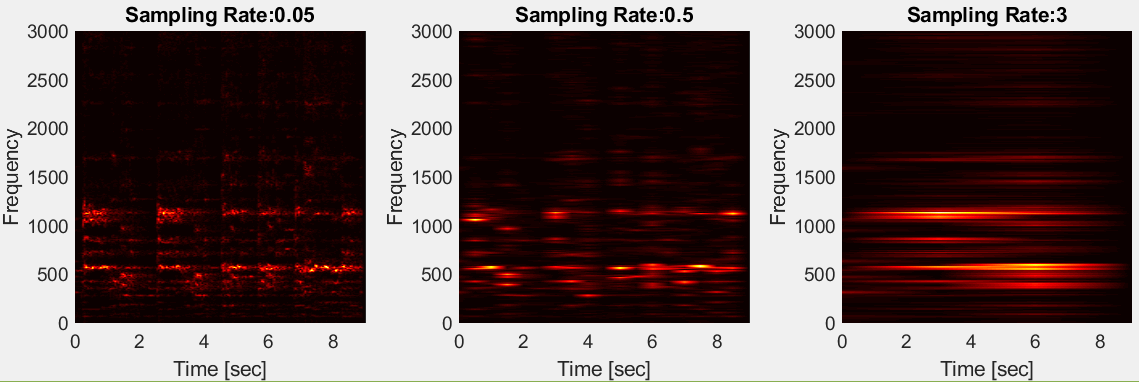
The first thing I investigated was the effect of different scaling factors on the spectrograms (Figure 1). Looking at this, when the scaling factor allows for too wide of a filter it takes in more data but then loses some details. For example when in Figure 1 we look at scaling factor of 1, although the piece has pauses in it we completely lose those. Similarly, for many of the finer details of the piece although we can tell that those frequencies exist we can’t tell when they occur. On the opposite side of the spectrum, so for scaling factor 100,000, we know where every pause occurred and many such details but we lose on the precision of the actual frequencies. For this piece a scaling factor of 1,000 seemed optimal to me as it lets us see the pauses in the music while still the details about the frequencies

Figure 2: Effects of the sampling rate on spectrograms with the Gabor filter

Figure 1: Effects of the scaling factor on spectrograms with the Gabor filter

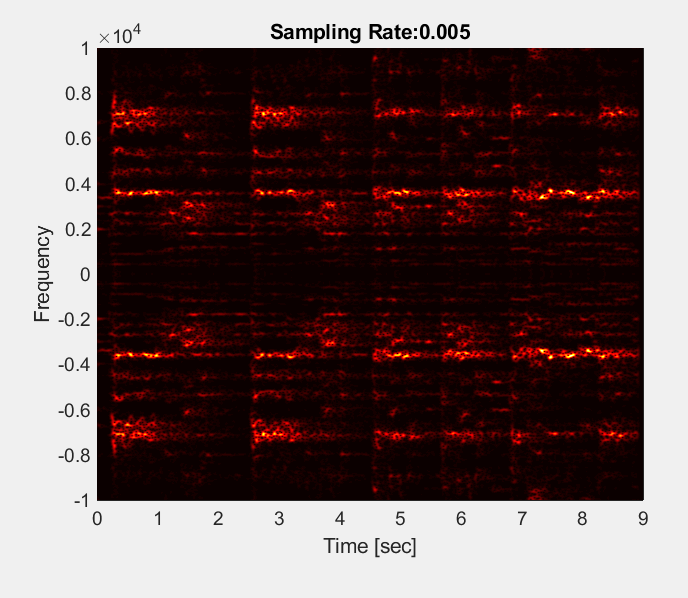
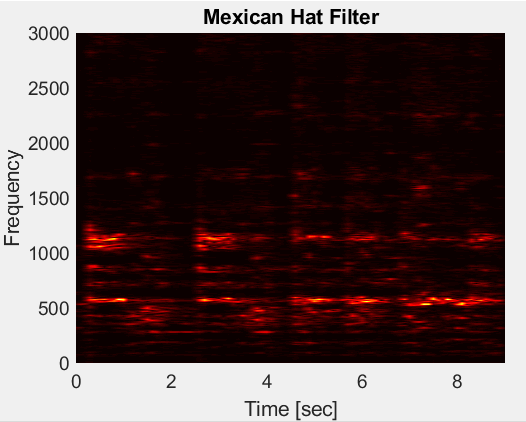
Another knob that can be utilized here is the sampling rate. By this I am meaning the distance between the locations we place our filter when generating the spectrogram. One of the biggest drawbacks of a very small sampling rate is that the computation time can easily become excessive. For example Figure 3 took several hours to generate, despite not showing any more data as compare to Figure 2 with sampling rate 0.05, which takes less than a minute. Having too high of a sampling rate can have similar effects to a wide scaling factor, where all the details of the piece are lost, for example the sampling rate of 3 in Figure 2. I continued to use a sampling rate of 0.1 as it seemed to strike a good balance between time to run and keeping details intact

Figure 3: Spectogram with very low sampling rate using the Gabor filter

The final knob to play with is what filter to use. Everything that I’ve done has been using the Gabor filter, but I talked about the Mexican Hat filter and the Step filter. Ultimately for this piece, after having found a good looking scaling they look extremely similar. One point against both Mexican Hat Filter and Step Filter in comparison to the Gabor filter though is that a smaller range of scaling values gave such well-behaved spectrograms

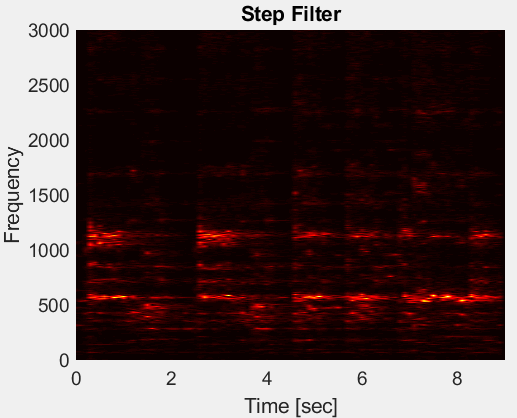
Upon seeing these spectrograms the immediate curiosity is what is happening at the different particularly active frequencies? Since I’m not a musical professional I found it difficult to be certain of what I’m hearing, but by applying a filter around 200 Hz and listening to the audio, it sounded like base notes of the piece and the tune. Doing the same process at around 550 Hz, it was the main choir singing. Lastly I listened to what is happening at around 1150 Hz, and that is either sopranos hitting high notes, or even more likely it is the timbre as it is precisely double what was clearly the main choir

Figure 4: Different Filters

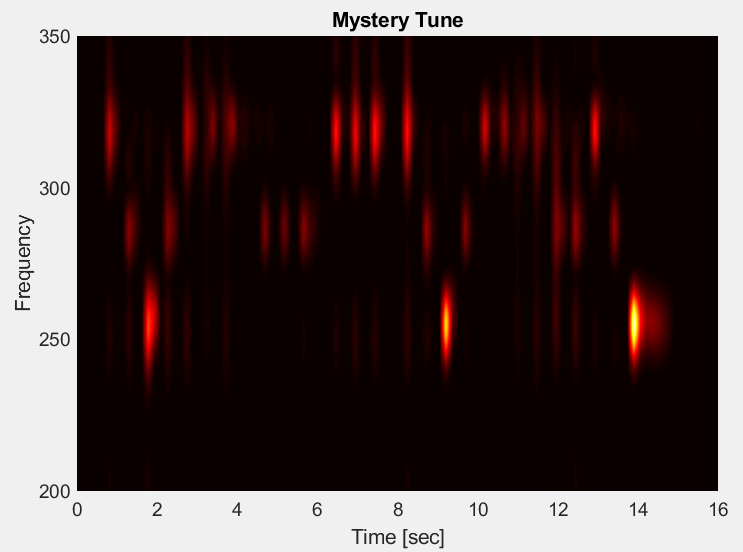
Next, the most fun portion of the project, trying to extract music. I followed a similar process of first figuring out what seemed to be the best scaling for the spectrogram of the piece, then focusing on the frequencies I was interested in (Figure 5). I ignored the timbre of the piano by focusing on the lowest but strongest frequencies and cutting out the rest of the data. Once I knew the range I was looking at, we can apply filters for the frequencies of particular piano notes of interest. In this case C: ~261 Hz, D: ~290 Hz, and E: ~329 Hz. Figure 6 shows the final result, and we can clearly tell that the tune is EDCDEEE pause DDD pause EEE pause EDCDEEEEDDEDC

Figure 5: Spectogram of the Mystery Tune

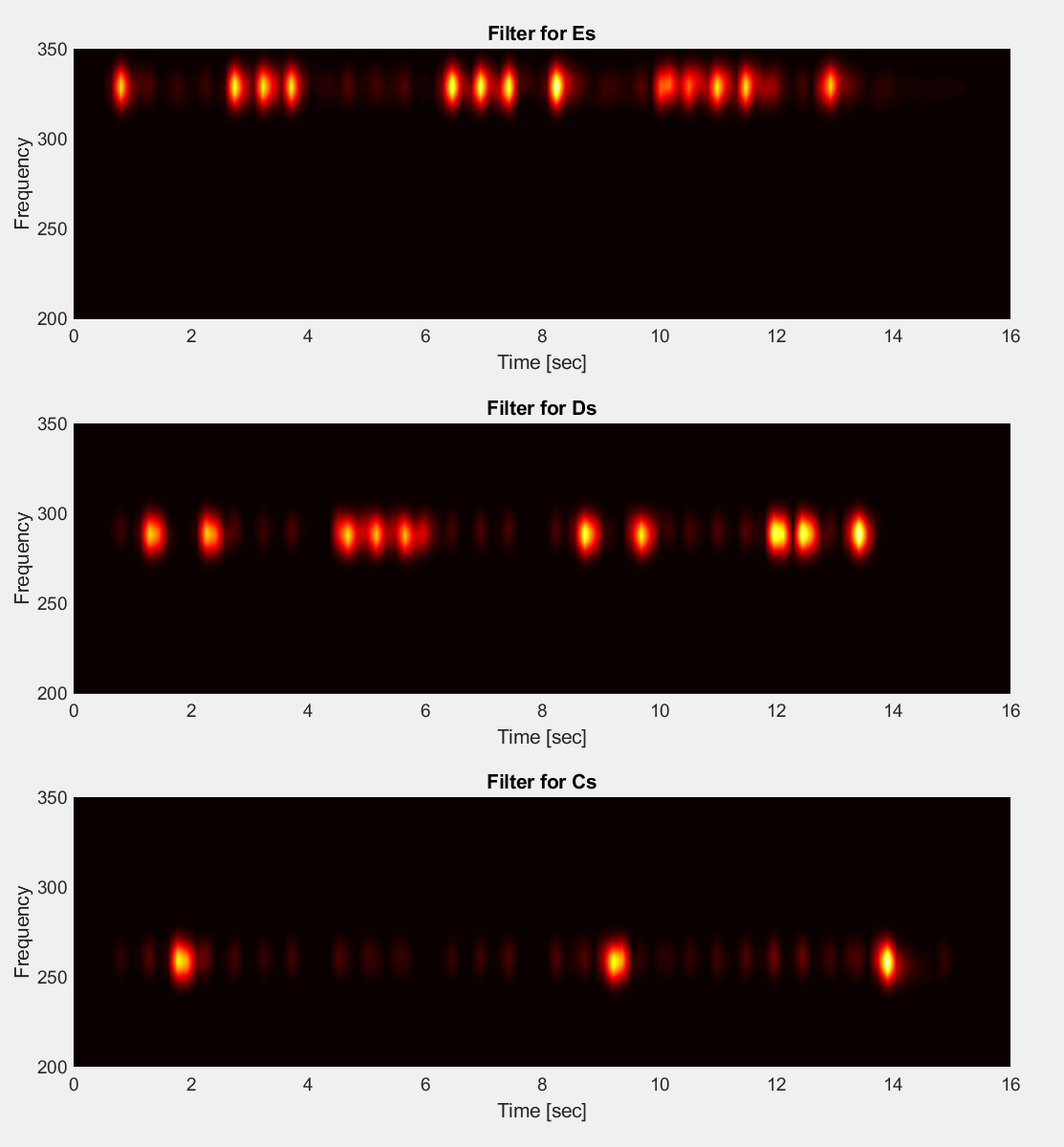
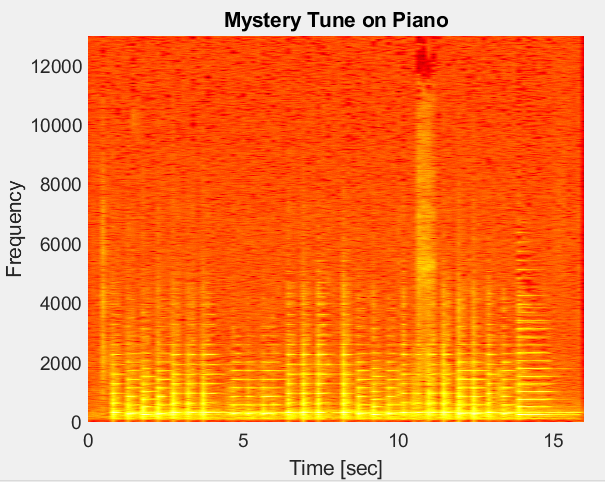


Figure 6: Mystery tune filtered by piano notes

The last thing of particular interest comes from looking at the log of the spectrogram for the tune played on the recorder vs. the piano (Figure 7). The reason for the log is to ensure that the timbre is visible

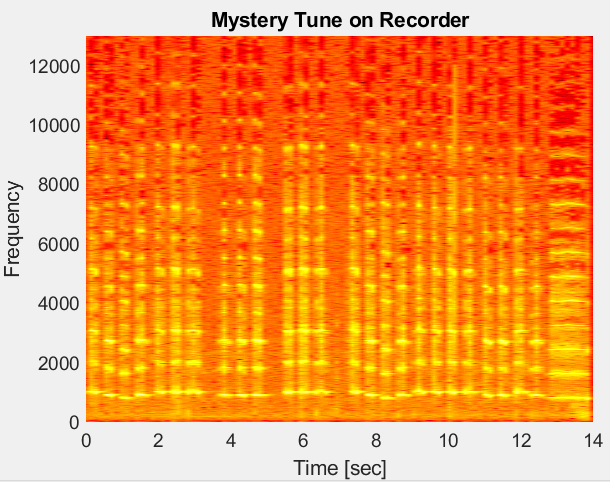


Figure 7: Timbre of Recorder vs. Piano

Looking at this we can see that the timbre on the recorder is lighting up many more of the frequencies in the spectrogram as compared to the piano

## Summary and Conclusion

Through time frequency analysis we can see many more detailed structures in data as compared to only performing a Fourier transform. After doing some analysis, we can even apply filters to extract the particular data desired from a piece

Appendix A – MATLAB Functions and Code Notes:

Fft, fftshift, ifft – functions for doing a Fast Fourier Transform, shifting it for readability, and Inverse Fast Fourier Transform. I used the built in functionality for perform the FFT for performance

Pcolor, shading, set, xlabel, ylabel, title, colormap, figure, subplot – functions used for plotting, organizing, and labeling the data

Audioplayer, playblocking – functions for taking a data set and playing it as music

Audioread – function for loading a sound dataset

Notes on code:

I implemented the filters as separate functions and a Specto function that takes any filter and creates the spectrogram. This is inefficient as calling into any function causes processing overhead, but made the code much easier for me to write and follow

On several occasions before playing filtered music I would multiply it by a seemingly random number. This is in order to increase the volume of the data – the filters cut out most of the sounds being played so when I focused on a small band of frequencies I needed to increase the volume to make any sense of what I’m hearing

I broke the code into 3 separately runnable sections, one for each dataset

Appendix B – MATLAB Code:

GaborFilter.m

function [g] = GaborFilter(t, scaling, center)

%GABOR Apply the gabor filter at time(s) t

% The Gabor filter is a Gaussian centered at a particular point

% with some given scaling. This function is meant to work for

% a single particular time or a vector of times

g=exp(-scaling.\*(t-center).^2);

end

MexicanHatFilter.m

function [m] = MexicanHatFilter(t, scaling, center)

%MEXICANHATFILTER Apply the mexican hat filter at time(s) t

m=(1-((t-center)/scaling).^2).\*exp(-(((t-center)/scaling).^2)/2);

end

StepFilter.m

function [s] = StepFilter(t, scaling, center)

%STEPFILTER Returns 1 in a range defined by center and scaling, 0

%everywhere else

s = double(t >= (center - scaling/2) & t <= (center + scaling/2));

end

Specto.m

function [spectr] = Specto(transformFilter, signal, t, tslide, scaling)

%Specto Calculates a spectogram of the signal using the given transform

% Transform is any transform function that calculates an FFT

spectr=[];

for j=1:length(tslide)

gt=fft(transformFilter(t, scaling, tslide(j)).\*signal);

spectr=[spectr;

% The transform function computes the FFT, so we need to shift it

% to be correct when trying to view it

abs(fftshift(gt))];

end

end

Homework2.m

%% Investigating the Handel dataset

clear all; close all; clc

% Load dataset

load handel

v = y'/2;

t = (1:length(v))/Fs;

n = length(v);

L = n/Fs;

% Basic look at dataset

% figure(1)

% plot(t,v);

% xlabel('Time [sec]');

% ylabel('Amplitude');

% title('Signal of Interest, v(n)');

%p8 = audioplayer(v,Fs);

%playblocking(p8);

% 1/L because we are looking at frequencies instead of steps

k=(1/L)\*[0:n/2 -n/2:-1];

ks=fftshift(k);

tslide=0:0.1:9;

% What is the effect of different scaling factors?

figure(1);

scalings = [1 10 100 1000 10000 100000];

for j = 1:length(scalings)

scaling = scalings(j);

Sgt\_spec = Specto(@GaborFilter, v, t, tslide, scaling);

subplot(2,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Scaling Factor:', num2str(scaling)));

colormap(hot)

end

% What is the effect of different sampling rates?

figure(2);

bestScaling = 1000;

sampling = [0.05 0.5 3];

for j = 1:length(sampling)

sampleRate = sampling(j);

tslide=0:sampleRate:9;

Sgt\_spec=Specto(@GaborFilter, v, t, tslide, bestScaling);

subplot(1,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Sampling Rate:', num2str(sampleRate)));

colormap(hot)

end

% How does a Mexican Hat Filter look in comparison?

tslide=0:0.1:9;

figure(3);

scalings = [0.05];

for j = 1:length(scalings)

scaling = scalings(j);

Sgt\_spec = Specto(@MexicanHatFilter, v, t, tslide, scaling);

%subplot(2,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Mexican Hat Filter');

colormap(hot)

end

% How does a Step Filter look in comparison?

tslide=0:0.1:9;

figure(4);

scalings = [0.1];

for j = 1:length(scalings)

scaling = scalings(j);

Sgt\_spec = Specto(@StepFilter, v, t, tslide, scaling);

%subplot(2,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Step Filter');

colormap(hot)

end

% Plot the best Step Filter alone for a deeper look into frequencies

figure(5)

scaling = 0.1;

Sgt\_spec = Specto(@StepFilter, v, t, tslide, scaling);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Scaling Factor:', num2str(scaling)));

colormap(hot)

% We're always seeing some interesting behaviors at around 550 Hz and

% around 1100 Hz, we can hone in on those and listen to them

% original for reference

p8 = audioplayer(v,Fs);

playblocking(p8);

% around 550 sounds like the main choir

tau = 0.00005;

volumeScale = 10;

filter = volumeScale \* exp(-tau\*((ks - 550).^2));

newV = ifft(fftshift(fftshift(fft(v)).\*filter));

p8 = audioplayer(newV,Fs);

playblocking(p8);

% around 1150 that sounds like the sopranos hitting high notes or it is the

% timbre

tau = 0.00005;

volumeScale = 10;

filter = volumeScale \* exp(-tau\*((ks - 1150).^2));

newV = ifft(fftshift(fftshift(fft(v)).\*filter));

p8 = audioplayer(newV,Fs);

playblocking(p8);

% this sounds like one of the base instruments

tau = 0.00005;

volumeScale = 10;

filter = volumeScale \* exp(-tau\*((ks - 200).^2));

newV = ifft(fftshift(fftshift(fft(v)).\*filter));

p8 = audioplayer(newV,Fs);

playblocking(p8);

% Sanity check - what are the effects of applying a filter on the

% Spectogram?

figure(6)

scaling = 0.1;

Sgt\_spec = Specto(@StepFilter, newV, t, tslide, scaling);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Effects of filter on Spectogram'));

colormap(hot)

%% Investigating "Mary had a little lamb" on piano dataset

clear all; close all; clc

tr\_piano=16; % record time in seconds

y=audioread('music1.wav'); Fs=length(y)/tr\_piano;

y = y';

% Basic look at dataset

plot((1:length(y))/Fs,y);

xlabel('Time [sec]'); ylabel('Amplitude');

title('Mary had a little lamb (piano)'); drawnow

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

n = length(y);

t = (1:n)/Fs;

k=(1/tr\_piano)\*[0:(n/2-1) -n/2:-1];

ks=fftshift(k);

tslide=linspace(0, tr\_piano, 50);

% First lets figure out a good scaling factor

figure(3);

scalings = [100 1000 10000];

for j = 1:length(scalings)

scaling = scalings(j);

Sgt\_spec = Specto(@GaborFilter, y, t, tslide, scaling);

subplot(1,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Scaling Factor:', num2str(scaling)));

colormap(hot)

end

% Scaling factor 1000 looks best

bestScaling = 1000;

% Now we want to filter for particular notes. Looking at the image from

% scaling factor 1000 it seems that we have 3 notes between 250 and 350 Hz

figure(5)

tslide=linspace(0, tr\_piano, 100);

scaling = bestScaling;

Sgt\_spec = Specto(@GaborFilter, y, t, tslide, scaling);

pcolor(tslide,ks,log(Sgt\_spec.')),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Mystery Tune on Piano');

colormap(hot)

% Since we know the general range of every note (C: ~261 Hz, D: ~290 Hz,

% E: ~329 Hz), we almost don't even need to run any additional analysis to

% extract the notes being played. But lets put a small filter around

% every note and extract the data

tau = 0.1;

volumeScale = 10;

Cfilter = volumeScale \* exp(-tau\*((ks - 261).^2));

Cs = ifft(fftshift(fftshift(fft(y)).\*Cfilter));

Dfilter = volumeScale \* exp(-tau\*((ks - 290).^2));

Ds = ifft(fftshift(fftshift(fft(y)).\*Dfilter));

Efilter = volumeScale \* exp(-tau\*((ks - 329).^2));

Es = ifft(fftshift(fftshift(fft(y)).\*Efilter));

% Look at what the signal looks like post filter

figure(5)

tslide=linspace(0, tr\_piano, 100);

scaling = bestScaling;

Sgt\_spec = Specto(@GaborFilter, Es, t, tslide, scaling);

subplot(3,1,1);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[200 350],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Filter for Es');

colormap(hot)

Sgt\_spec = Specto(@GaborFilter, Ds, t, tslide, scaling);

subplot(3,1,2);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[200 350],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Filter for Ds');

colormap(hot)

Sgt\_spec = Specto(@GaborFilter, Cs, t, tslide, scaling);

subplot(3,1,3);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[200 350],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Filter for Cs');

colormap(hot)

%% Investigating "Mary had a little lamb" on recorder dataset

clear all; close all; clc

tr\_rec=14; % record time in seconds

y=audioread('music2.wav'); Fs=length(y)/tr\_rec;

y = y';

figure(1)

% Basic look at dataset

plot((1:length(y))/Fs,y);

xlabel('Time [sec]'); ylabel('Amplitude');

title('Mary had a little lamb (recorder)');

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

n = length(y);

t = (1:n)/Fs;

k=(1/tr\_rec)\*[0:(n/2-1) -n/2:-1];

ks=fftshift(k);

tslide=linspace(0, tr\_rec, 50);

% First lets figure out a good scaling factor

figure(3);

scalings = [100 1000 10000];

for j = 1:length(scalings)

scaling = scalings(j);

Sgt\_spec = Specto(@GaborFilter, y, t, tslide, scaling);

subplot(1,3,j);

pcolor(tslide,ks,Sgt\_spec.'),

shading interp

set(gca,'Ylim',[0 3000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title(strcat('Scaling Factor:', num2str(scaling)));

colormap(hot)

end

% Scaling factor 1000 looks best

bestScaling = 1000;

% Now we want to filter for particular notes. Looking at the image from

% scaling factor 1000 it seems that we have 3 notes between 750 and 1100 Hz

figure(4)

tslide=linspace(0, tr\_rec, 100);

scaling = bestScaling;

Sgt\_spec = Specto(@GaborFilter, y, t, tslide, scaling);

pcolor(tslide,ks,log(Sgt\_spec.')),

shading interp

set(gca,'Ylim',[0 13000],'Fontsize',[14])

xlabel('Time [sec]');

ylabel('Frequency');

title('Mystery Tune on Recorder');

colormap(hot)