UNIVERSITY OF WASHINGTON

AMATH 482 A WI 20: COMPUTATIONAL METHODS FOR DATA ANALYSIS

Homework 2

Abstract

The goal of this project is to analyze and filter different audio signals.
In the first part we take a look at a short portion of Handel's Messiah and analyze it with different Gabor Filter to produce spectrograms of the piece of work. The quality of the spectrograms is obtained with different shapes, sizes and density for the filters. In the second part we analyze another signal. We take a look at the overtones produced by a piano and try to filter them out and compare the signal structure to one recorded with a recorder.

Ather Ahmad

Instructor: Craig GIN

1 Introduction and Overview

To deconstruct a signal we can use a Fourier transform and look at the frequencies produced by such. Now if we just transform the whole signal we will see all the frequencies but we won't have any information about the time to tell at which point which frequency was obtained. To solve this problem we can take portions of the signal with (Gabór-)filters and let these filter travel over the signal. This way we won't have an overview of all the Frequencies at once, but will have some information about the time.

2 Theoretical Background

2.1 Fourier Transform

The general idea behind the Fourier Transform is to obtain a spectrum of a given function. The most common use is to analyze a signal over time and transform it into the frequency space to find all frequencies which build that signal.

2.1.1 Continuous Fourier Transform

Definition of a Fourier Transform for a given function

$$\hat{f}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x)\, e^{-ikx} dx$$

and can be inverted back with

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(k)\, e^{ikx} dk\ .$$

Instead of transforming the given function it can be decomposed in *sin* and *cos* terms with **Fourier coefficients (Fourier series)**

$$f(x) = \frac{a_0}{2} \sum_{k=1}^{\infty} \left[a_2\, cos(kx) + b_k\, sin(kx)\right]$$

with

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\, cos(kx)\, dx \quad k \geq 0$$
$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x)\, sin(kx)\, dx \quad k > 0\ .$$

2.1.2 Discrete Fourier Transform

To calculate our function we have to discretize our space:

$$x \in \mathbb{R} \rightarrow x \in \{x_0,\ x_1,\ x_2,\ ...,\ x_{N-1}\}$$

this gives us new terms for the Fourier transform

$$\hat{x}_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi ikn}{N}}$$
$$x_k = \sum_{n=0}^{N-1} \hat{x_k} e^{\frac{2\pi ikn}{N}}$$

2.2 Different filters used

Here is a summary of the filters used in this project:

2.2.1 Gauß filter

$$f(x) = e^{-a(t-\tau)^2}$$

2.2.2 Mexican hat wavelet filter

$$f(x) = (1 - \ (a \cdot (t-\tau))^2\ )\ e^{-\frac{(a \cdot (t-\tau))^2}{2}}$$

2.2.3 Morlet wavelet filter

$$f(x) = e^{-\frac{1}{2}\ a(t-\tau)^2} \cos\left(2 \cdot \sqrt{a}(t-\tau)\right)$$

a: filter width
τ: The time at which the mean of the function is located

3 Algorithm Implementation and Development

3.1 Part 1

To obtain good results for the Gabór spectograms a **for loop** was used calculating the heat map for different widths for the filters. Also for loops were used to plot an animation showing different portions of the signal and their Fourier transform to compare them to the sound for a qualitative verification of the transforms.

4 Computational Results

4.1 Part 1

4.1.1 Gauß filter

The spectogram was obtained using a Gauß filter which was moved over the signal

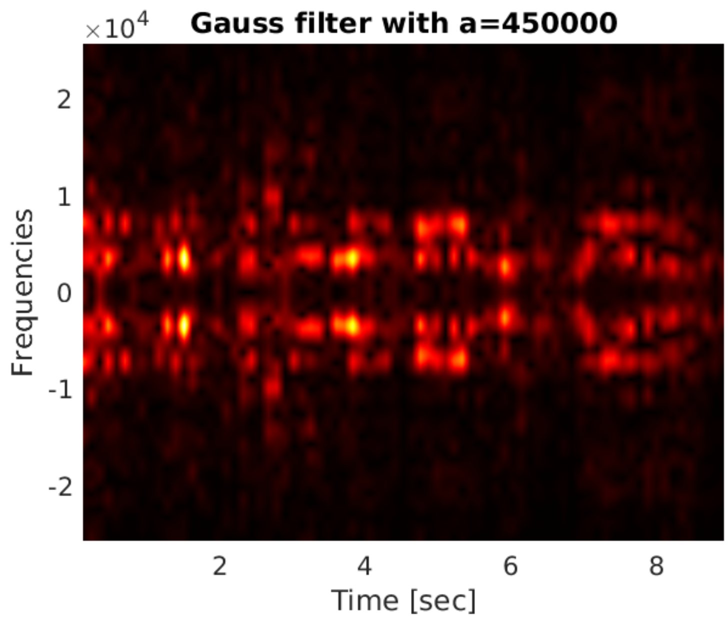

Figure 1: Gabór spectogram for a Gauß filter

4.1.2 Mexican hat wavelet filter

The spectogram was obtained using a Mexican hat wavelet filter which was moved over the signal

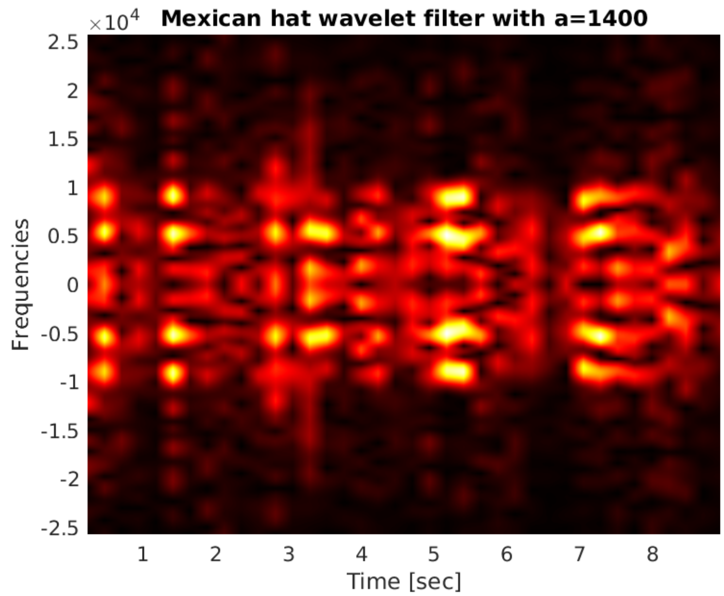

Figure 2: Gabór spectogram for a mexican hat wavelet filter

5 Summary and Conclusions

6 Appendix A: MATLAB functions used and brief implementation explanation

7 Appendix B: MATLAB codes

%% Homework 2
close all, clc, clear all
%% Define colors
orange = [1 0.7 0]; orange2 = [0.5 0.6 0.5]; red = [0.8 0.1 0.1]; %% Data load handel
%% Define signal properties
signal = y';
signal_data_points = length(signal); signal_time = signal_data_points / Fs;
%% Define time
data_vector = 1:signal_data_points; time_vector = data_vector / Fs;

%% Sound plot
figure(1)
subplot(4,1,1) plot(time_vector,signal); axis([0 signal_time -1 1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Signal of Interest, v(n)');
saveas(gcf, 'handel.png')
%% play music %p8 = audioplayer(v,Fs); %playblocking(p8);
%% play music %sound(y,Fs);
%% Fourier
frequencies_space = (2*pi/signal_time)*[0:(signal_data_points/2) -signal_data_points/2:-1]; frequencies_space_shifted = fftshift(frequencies_space);
transformed_signal = fft(signal);
subplot(4,1,2) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); xlabel('Frequncies'); ylabel('Amplitude'); title('Frequencies of Interest');
saveas(gcf, 'handelfft.png')
%% Gauss filter tau = 4.5; a = 1; gauss_filter = exp(-a*(time_vector-tau).^2);
subplot(4,1,3) plot(time_vector,gauss_filter, 'Linewidth', 2, 'Color', orange) axis([0 signal_time 0 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Gauss Filter');
%% adding gauss filter
vg = gauss_filter.*signal;
vgf = fft(vg);
subplot(4,1,4) plot(frequencies_space_shifted,fftshift(abs(vgf)), 'Color', red); xlabel('Frequncies'); ylabel('Amplitude'); title('Frequencies of Interest');
pause(7) %% Gauss Animation (width)
tau = 4.5; number_of_filters = 8;
tau0 = 0;
for a = 1:number_of_filters
ar = 5^(a/2);
gauss_filter = exp(-ar *(time_vector - tau).^2);
vg = gauss_filter.*signal;
vgf = fft(vg);
subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of whole signal');
subplot(3,1,2) plot((1:signal_data_points)/Fs,signal); hold on plot(time_vector,gauss_filter, 'Linewidth', 2, 'Color', orange) plot(time_vector,vg, 'Color', orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Gauss Filter');
subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vgf)), 'Color', red); xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of filtered signal');
pause(3)
end

```
pause(5)
%% Gauss Animation (time)
steps_per_second = 20; steps = steps_per_second * signal_time; tau0 = 0; a = 100;
for tau = 1:steps
gauss_filter = exp(-a*(time_vector - (tau0 + tau/steps_per_second)).^2);
vg = gauss_filter.*signal;
vgf = fft(vg); subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), ’Color’, red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel(’Frequencies’); ylabel(’Amplitude’); title(’Frequencies of whole signal’);
subplot(3,1,2) plot((1:signal_data_points)/Fs,signal); hold on plot(time_vector,gauss_filter, ’Linewidth’, 2, ’Color’, orange) plot(time_vector,vg, ’Color’, orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel(’Time [sec]’); ylabel(’Amplitude’); title(’Gauss Filter’);
subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vgf)), ’Color’, red); axis([-signal_data_points/2 signal_data_points/2 0 100]) xlabel(’Frequencies’); ylabel(’Amplitude’); title(’Frequencies of filtered signal’);
pause(0.000001)
end
pause(2)
%% Gabor spectogram for Gauss filter close all
for a = 1:20
ar = 30*10^(4)+ a*10^(4); tau0 = 0; steps = ceil((ar)^(1/2)); time_gabor = (1:steps) * (signal_time / steps); steps_per_second = steps / signal_time;
vgf = zeros(steps, signal_data_points);
for k = 1:steps
gauss_filter = exp(-ar*(time_vector - (tau0 + k/steps_per_second)).^2);
vg = gauss_filter.*signal;
vgf(k, :) = fft(vg);
count = [a, k / steps]
end
vgf = vgf’;
figure(a+1)
pcolor(time_gabor, frequencies_space_shifted, fftshift(abs(vgf))); shading interp colormap(hot) xlabel(’Time [sec]’); ylabel(’Frequencies’); title_name = sprintf(’Gauss filter with a=%d’, ar); title(title_name); file_name = sprintf(’a_%dgauss.png’,ar); saveas(gcf, file_name)
end
pause(5) %% Mexican hat wavelet
t = 4.5; sigma = 2; mexican_filter = (1-(sigma*(time_vector-t)).^2) .* exp(-(sigma*(time_vector-t)).^2/2); subplot(4,1,3) plot(time_vector,mexican_filter, ’Linewidth’, 2, ’Color’, orange) axis([0 signal_time -0.6 1.1]) xlabel(’Time [sec]’); ylabel(’Amplitude’); title(’Mexican hat wavelet filter’);
```

```
%% adding mexican filter
vh = mexican_filter.*signal;
vhf = fft(vh);
subplot(4,1,4) plot(frequencies_space_shifted,fftshift(abs(vhf)), 'Color', red); xlabel('Frequncies'); ylabel('Amplitude'); title('Frequencies of Interest');
pause(5)
%% Mexican hat Animation (width)
t = 4.5; number_of_filters = 16;
for sigma = 1:number_of_filters
sigma = sigma - 5; mexican_filter = (1-(2^(sigma)*(time_vector-t)).^2) .* exp(-(2^(sigma)*(time_vector-t)).^2/2);
vh = mexican_filter.*signal;
vhf = fft(vh);
subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of whole signal');
subplot(3,1,2) plot(time_vector,signal); hold on plot(time_vector,mexican_filter, 'Linewidth', 2, 'Color', orange) plot(time_vector,vh, 'Color', orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Mexican hat wavelet Filter');
subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vhf)), 'Color', red); xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of filtered signal');
pause(3)
end
pause(5)
%% Mexican hat Animation (time)
steps_per_second = 20; steps = steps_per_second * signal_time; t0 = 0; sigma = 30;
for t = 1:steps
mexican_filter = (1-(sigma*(time_vector-(t0 + t / steps_per_second))).^2) .* exp(-(sigma*(time_vector-(t0 + t / steps_per_second))).^2/2);
vh = mexican_filter.*signal;
vhf = fft(vh);
subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of whole signal');
subplot(3,1,2) plot(time_vector,signal); hold on plot(time_vector,mexican_filter, 'Linewidth', 2, 'Color', orange) plot(time_vector,vh, 'Color', orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Mexican hat wavelet Filter');
subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vhf)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 100]) xlabel('Frequencies'); ylabel('Amplitude');
```

```
title('Frequencies of filtered signal');
    pause(0.000001)
    end
    pause(2)
    %% Gabor spectogram for Mexican hat wavelet filter close all
    t0 = 0;
    for sigma = 1:10
    sigmar = sigma * 10^2 + 500; steps = ceil((sigmar)^(1/2)); time_gabor = (1:steps) *
(signal_time / steps); steps_per_second = steps / signal_time;
    vgf = zeros(steps, signal_data_points);
    for k = 1:steps
    mexican_filter = (1-(sigmar*(time_vector-(t0 + k / steps_per_second))).^2) .* exp(-
(sigmar*(time_vector-(t0 + k / steps_per_second))).^2/2);
    vg = mexican_filter.*signal;
    vgf(k, :) = fft(vg);
    count = [sigma, k / steps]
    end
    vgf = vgf';
    figure(sigma+1)
    pcolor(time_gabor, frequencies_space_shifted, fftshift(abs(vgf))); shading interp
%set(gca,'Ylim',[-50 50],'Fontsize',16) colormap(hot) xlabel('Time [sec]'); ylabel('Frequencies');
title_name = sprintf('Mexican hat wavelet filter with a=%d', sigmar); title(title_name);
file_name = sprintf('a_%dmexican.png',sigmar); saveas(gcf, file_name)
    end
    pause(5) %% Morlet wavelet
    t = 4.5; sigma = 5; morlet_filter = exp(-1/2.*(time_vector-t).^2).*cos(sigma .* (time_vector-
t)); subplot(4,1,3) plot(time_vector,morlet_filter, 'Linewidth', 2, 'Color', orange) axis([0
signal_time -1 1.2]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Morlet filter');
    %% adding morlet filter
    vm = morlet_filter.*signal;
    vmf = fft(vm);
    subplot(4,1,4) plot(frequencies_space_shifted,fftshift(abs(vmf)), 'Color', red); xla-
bel('Frequncies'); ylabel('Amplitude'); title('Frequencies of Interest');
    pause(5)
    %% Morlet Animation (width)
    t = 4.5; number_of_filters = 16;
    for sigma = 1:number_of_filters
    morlet_filter = exp(-1/2.*10*1.5^((sigma-number_of_filters/2)*4)*(time_vector-t).^2).*cos(20*1.5^((si
number_of_filters/2)*4)^(1/2).*(time_vector-t));
    vm = morlet_filter.*signal;
    vmf = fft(vm);
```

subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of whole signal');

subplot(3,1,2) plot(time_vector,signal); hold on plot(time_vector,morlet_filter, 'Linewidth', 2, 'Color', orange) plot(time_vector,vm, 'Color', orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Morlet Filter');

subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vmf)), 'Color', red); xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of filtered signal');

pause(3)

end

pause(5)

%% Morlet Animation (time)

steps_per_second = 20; steps = steps_per_second * signal_time; t0 = 0; sigma = 50;

for t = 1:steps

morlet_filter = exp(-1/2.*10*sigma*(time_vector-(t0 + t/steps_per_second)).^2).*cos(20*sigma^(1/2). (t0 + t/steps_per_second)));

vm = morlet_filter.*signal;

vmf = fft(vm);

subplot(3,1,1) plot(frequencies_space_shifted,fftshift(abs(transformed_signal)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 800]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of whole signal');

subplot(3,1,2) plot(time_vector,signal); hold on plot(time_vector,morlet_filter, 'Linewidth', 2, 'Color', orange) plot(time_vector,vm, 'Color', orange2) hold off axis([0 signal_time -1.1 1.1]) xlabel('Time [sec]'); ylabel('Amplitude'); title('Morlet Filter');

subplot(3,1,3) plot(frequencies_space_shifted,fftshift(abs(vmf)), 'Color', red); axis([-signal_data_points/2 signal_data_points/2 0 100]) xlabel('Frequencies'); ylabel('Amplitude'); title('Frequencies of filtered signal');

pause(0.000001)

end

pause(2)

%% Gabor spectogram for Mexican hat wavelet filter close all

t0 = 0;

for sigma = 9:10

sigmar = sigma*10000; steps = ceil((sigmar)^(1/2)); time_gabor = (1:steps) * (signal_time / steps); steps_per_second = steps / signal_time;

vgf = zeros(steps, signal_data_points);

for k = 1:steps

morlet_filter = exp(-1/2.*sigmar*(time_vector-(t0 + k/steps_per_second)).^2).*cos(2*sigmar^(1/2).*((t0 + k/steps_per_second)));

vg = morlet_filter.*signal;

vgf(k, :) = fft(vg);

```
count = [sigma, k / steps]
end
vgf = vgf';
figure(sigma+1)
pcolor(time_gabor, frequencies_space_shifted, fftshift(abs(vgf))); shading interp
%set(gca,'Ylim',[-50 50],'Fontsize',16) colormap(hot) xlabel('Time [sec]'); ylabel('Frequencies');
title_name = sprintf('Morltet filter with a=%d', sigmar); title(title_name); file_name =
sprintf('a_%dmorlet.png',sigmar); saveas(gcf, file_name)
end
pause(5)
```