Utilizing Gábor transforms and spectrograms to perform time-frequency analysis

Arman Naderi

Abstract:

Time-frequency analysis is essential for the characterization and modification of signals that vary in time; these techniques can even be applied to signals collected in the spatial domain. One key method by which frequency domain components are represented is through the use of spectrograms. When coupled with the Gábor transform, useful frequency and phase information can be determined from local frames of signal as those frames change over time. In this work, I demonstrate the utility of both the Gábor transform and spectrograms for the time-frequency analysis of three pieces of music performed with different instruments.

Introduction and Overview:

Time-frequency analysis consists of a broad range of techniques which have a wide variety of applications in signal processing. The fundamental principle that underlies time-frequency analysis is that rather than view, characterize, and manipulate a signal collected in one dimension, such as time or space, time-frequency analysis allows for a signal to be analyzed in two dimensions wherein the signal is a function of both time and frequency. The method by which this signal changes domains is via time-frequency (or space-frequency) transform. There are a variety of methods that allow for the visualization of a signal's frequency domain. In modern-day digital signal processing, the quickest method to transform a signal from the time or space domain to the frequency domain is via the Fast Fourier Transform (FFT). Spectrograms are a useful representation of the frequency domain of a signal as it allows the visualization of the relationship between a signal's frequency domain components, their intensity, and time.

There are a variety of methods by which the frequency domain components of a given signal can be represented and transformed from the time domain. One such method is through the use of short-time Fourier transforms (STFT). The short-time Fourier transform allows for a signal's frequency and phase contents to be analyzed as they change over time while being collected from frames of the original signal. One such case of STFTs is the Gábor transform. This transform multiples the signal of interest by a Gaussian before transforming it, which causes frequency signal collected at the midpoint of the Gaussian to be weighted higher than at the ends of the Gaussian. In itself, the Gábor transform can be used as a method for signal filtering.

In this work, I will be applying the Gábor transform to spectrograms to analyze three pieces of music: Handel's *Messiah*, Mary Had a Little Lamb performed on the piano, and Mary Had a Little Lamb performed on the recorder. I will first demonstrate how Gábor filtering can be used to produce spectrograms; then, I will explore how changing the window width effects the resultant spectrograms. In addition, I will analyze how changing parameters of the Gaussian used in the Gábor transform cause oversampling and undersampling of signal. I will also demonstrate different window functionality in the Gábor window through the implementation of the use Mexican Hat wavelet. Lastly, I will apply the Gábor transform to Mary Had a Little Lamb to determine its music score.

Theoretical Background:

The Gábor transform is in itself a version of the STFT. It allows for windowed analysis of the frequency components of a given signal. The formula for the Gábor transform is shown in Equation 1.a, wherein f(t) is the signal of interest, and g(t) is the window function.

$$\tilde{f}(\tau,\omega) = \int_{-\infty}^{\infty} f(t) * g(t-\tau) * e^{-i\omega t} * dt \qquad (1.a)$$

The most common window applied to the Gábor transform is the Gaussian shown in Equation 2.a.

$$g(t-\tau) = e^{-a(t-\tau)^2} \qquad (2.a)$$

By scaling the signal of interest by the Gaussian at various time points, we are effectively filtering the signal to favor the values of τ at which the Gaussian window is based upon to attenuate. Changing this value τ will change the windows at which signal is attenuated. By modifying the scale of a, the signal can effectively be oversampled and undersampled from the original. Oversampling a signal is only detrimental if the signal becomes so plentiful that it burdens the computation time, however it is assured that more frequency information is captured in the sample set. Undersampling can be detrimental in that the decreased amount of signal may actually end up losing valuable frequency information, however the computation time will decrease as there is less data in the sample set. There are other possible window functions that can be applied in the Gábor transform; one such function is the Mexican Hat wavelet. This wavelet is defined in Equation 3.a.

$$\psi(t-\tau) = (1-(t-\tau)^2)e^{-\frac{(t-\tau)^2}{2}} \qquad (3.a)$$

Algorithm Implementation and Development:

The primary functionality of this algorithm is to compute the spectrogram of a signal of interest (SOI) as a function of the filter function center point, τ and time, t. This was done by first computing a matrix that accounts for the number of times τ appears in the time range of the signal. This matrix then guides the number of iterations at which the filter function is multiplied with the signal at a given time point, and subsequently taken to the frequency domain via the FFT. The shifted, absolute value of frequency domain component of the signal is then added to a list of time windows at which the filter was applied. This concatenated list of frequency domain components are then plotted with a hot color map to indicate their intensities as a function of time. By modifying the value of a in the Gaussian function, I was able to modify the size of the window at which signal was being attenuated. By optimizing this window size for each of the music pieces, I was able to create distinct spectrograms that represented the frequencies at which the music was being played. These parameters were modified to demonstrate the variety of analytical outputs possible during time-frequency analysis.

Computational Results:

The first part of the algorithm analyzed signal of interest 1, Handel's *Messiah*, utilizing a variety of parameters with the Gábor transform. Figure 1 demonstrates what the signal, v(n) looks like in amplitude as a function of time, how this signal looks in the frequency domain, and what the spectrogram of this signal looks like with perfect

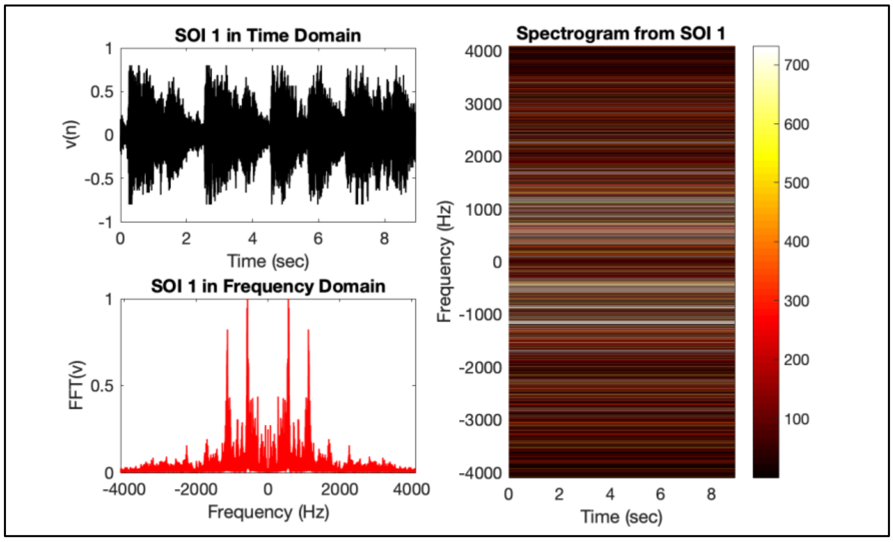

Figure 1: Handel's *Messiah,* represented in the time domain, frequency domain, and as a spectrogram.

frequency resolution, and no time resolution. Figure 2 demonstrates what the signal, v(n) looks like after transformation with a Gábor filter, with a Gaussian window function. The

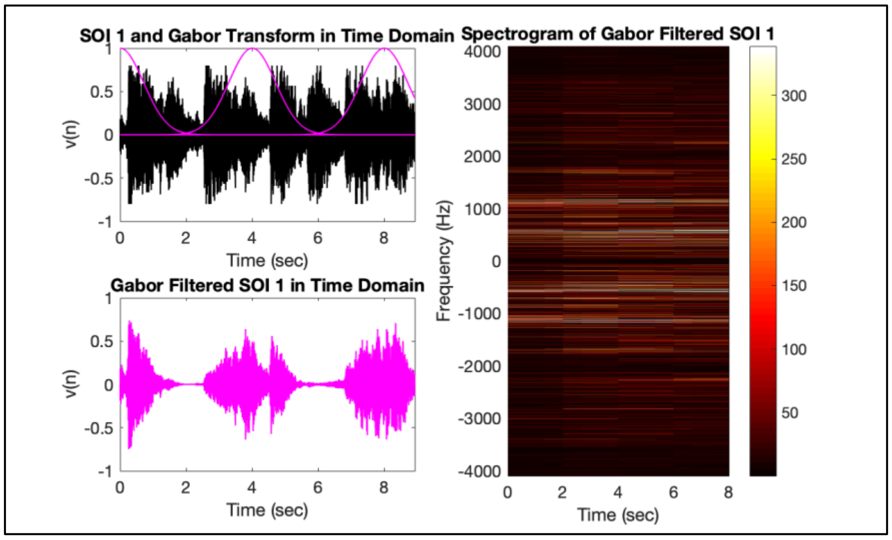

Figure 2: SOI 1, represented in the time domain with the Gabor Transform in magenta, where the Gabor transform is then applied to the signal, and the subsequent spectrogram is produced.

Gaussian utilized for this transformation has a tau of 4 and a of 1. Figure 3 demonstrates what SOI 1 looks like with a variety of a values applied to the Gaussian used in the Gábor filter, with a tau of 0.1. With larger values of a there is a finer time resolution, but coarser frequency resolution, which is indicated by more frequencies in the 2000-4000 Hz range

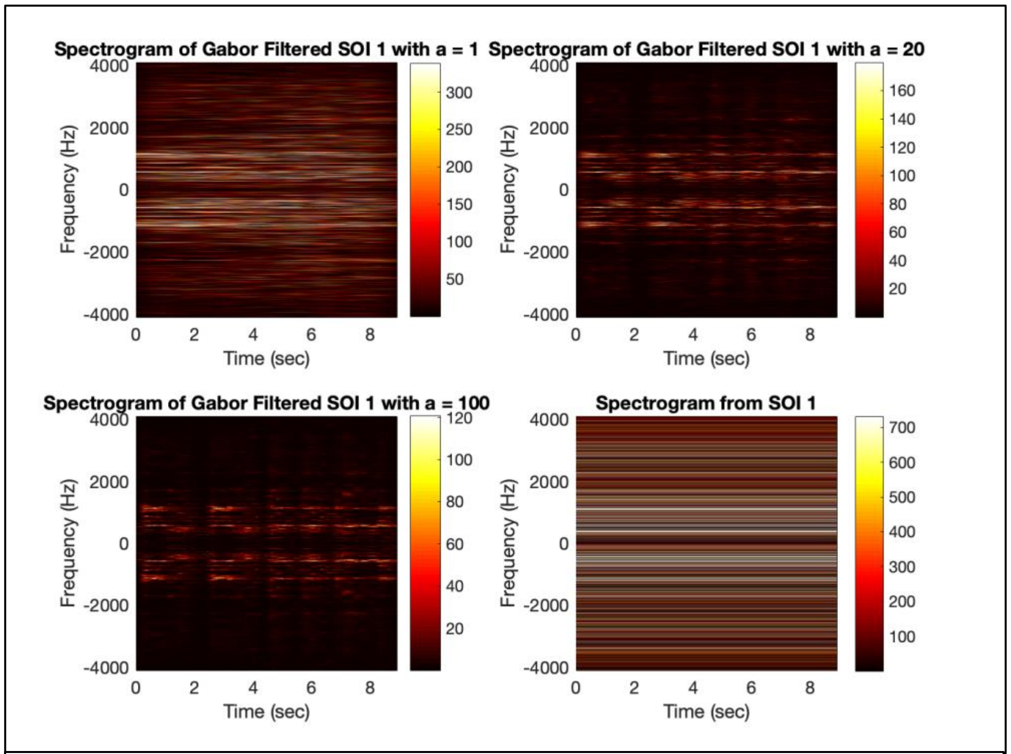

Figure 3: The spectrograms of SOI 1, wherein different a values in the Gábor filter's Gaussian are applied to result in different frequency representations of the signal.

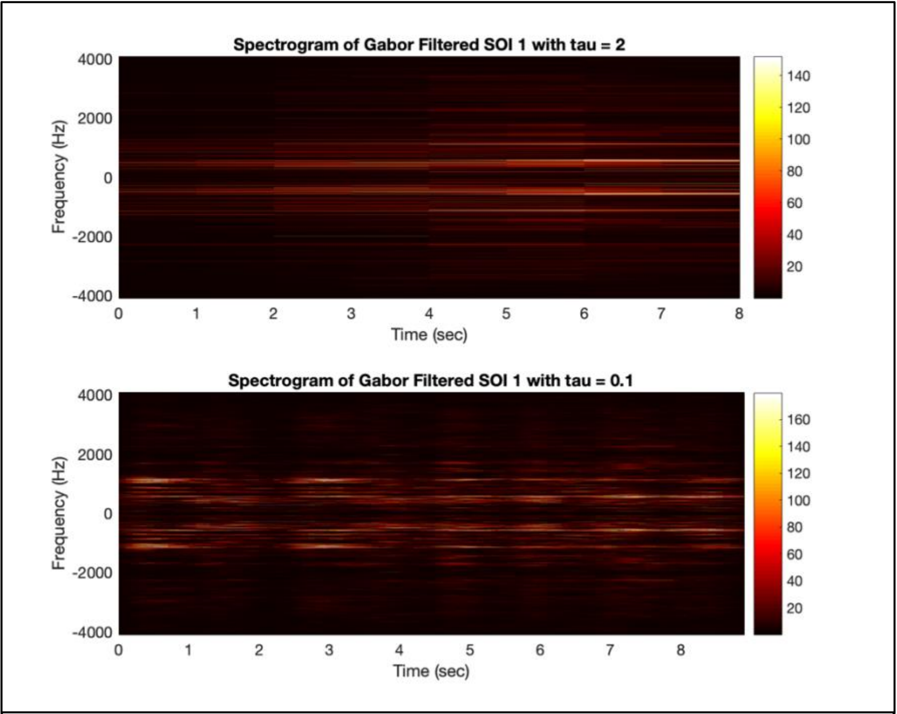

Figure 4: The spectrograms of SOI 1, wherein different tau values in the Gábor filter's Gaussian are applied to result in different frequency representations of the signal.

being attenuated or lessened in intensity with a values equal to 100 as opposed to values equal to 1. Figure 4 demonstrates the use of varying window sizes on SOI 1 to demonstrate the effects of window size on oversampling and undersampling of signal. Oversampled signal is useful for ensuring a more accurate representation of the original signal, but may be detrimental in that modifications of this sample set will require a longer computation time. Undersampled signal is useful for faster computation time, but may be detrimental as it may not accurately represent the original signal. Figure 5 demonstrates the use of a Mexican Hat wavelet in the Gábor filter. The Mexican Hat wavelet is better able to

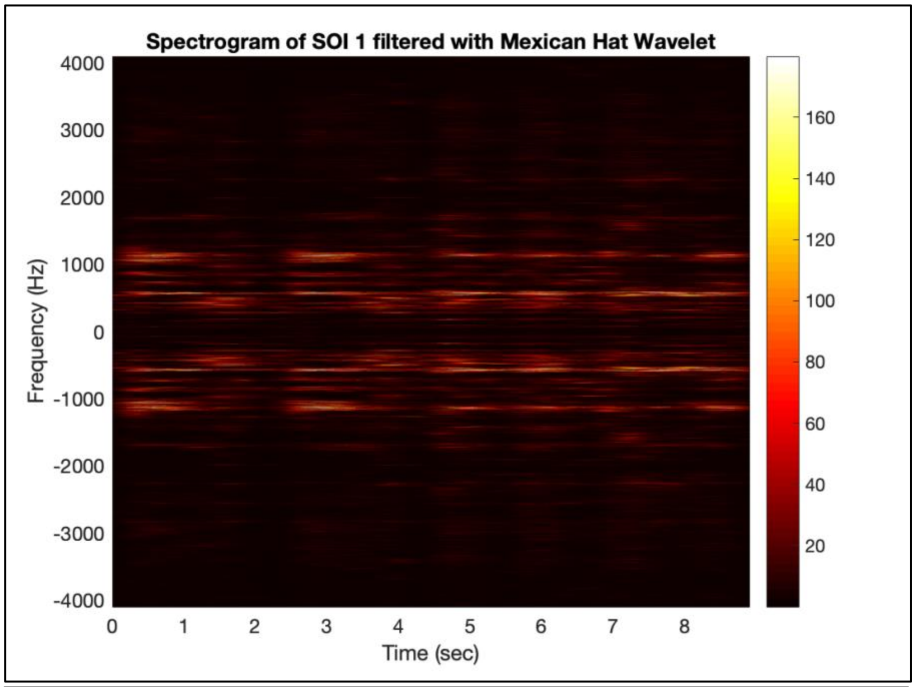

Figure 5: The spectrogram of SOI 1 with a Gábor filter applied that utilizes a Mexican Hat wavelet as a window function. The tau for the wavelet was set to 0.1.

represent the frequency signal at the ends of the window as the function itself is sharper. However, this results in more ringing, and thus is slightly more coarse in the frequency domain. Figure 6 demonstrates the Gábor filtering, with a tau of 0.1 and an a value of 100, of SOI 2 and 3, Mary Had a Little Lamb performed on piano and recorder, respectively. By looking at the produced spectrograms I was able to reproduce the sheet music for the SOI 2 recording: E4, D4, C4, D4, E4, E4, E4, D4, D4, D4, E4, E4, E4, E4, D4, C4, D4, E4, E4, E4, E4, D4, D4, E4, D4, C4. The SOI 3 recording follows the same pattern but with C6, A6, and G5 as the highest through lowest notes respectively. The primary differences between these two recordings is that the instrument used for each piece demonstrates the different overtones they are able to produce. The piano produces very strong overtones, and thus if notes are played at frequency ω_0, the piano's timbre will produced overtones at $2\omega_0$, $3\omega_0$ and so on. The recorder does not have as strong of timbre so these overtones are not as visible, but they are still present. Figure 7 demonstrates the same spectrogram with these overtones filtered out.

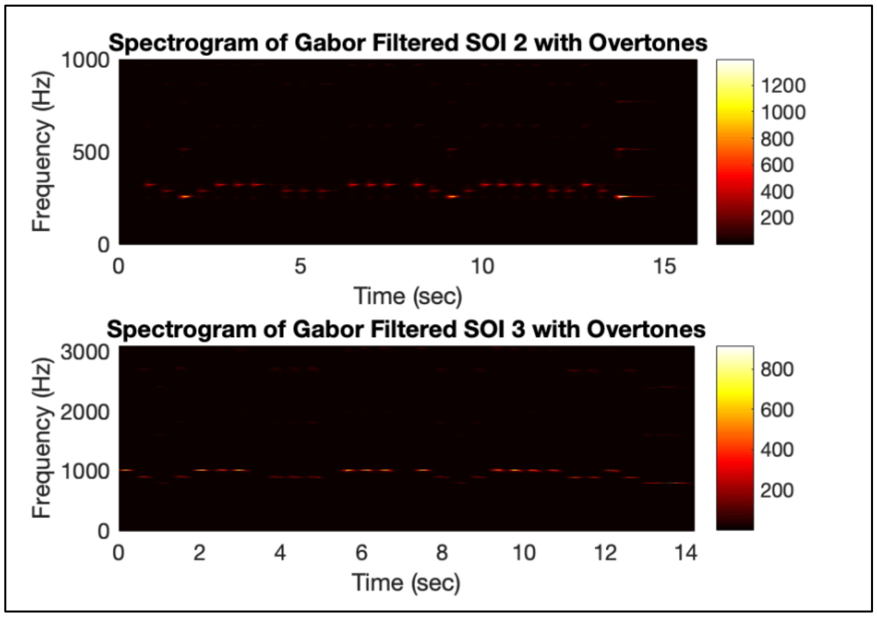

Figure 6: Spectrograms of the Gábor filtered SOI 2 and 3, with overtones included.

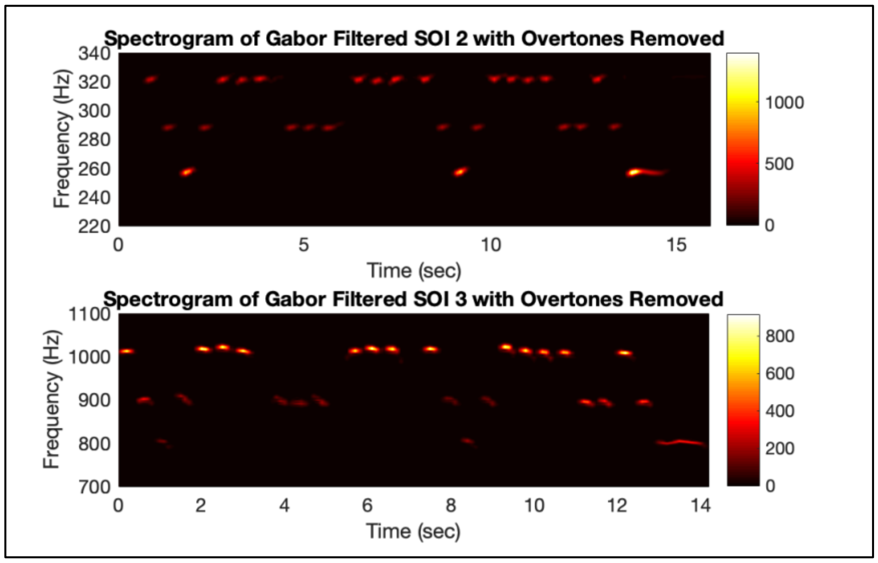

Figure 7: Spectrograms of the Gábor filtered SOI 2 and 3, with overtones filtered out.

Summary and Conclusions:

Spectrograms produced via the Gábor transform have a wide variety of applications and uses in the processing of various signals. In this paper, I was able to demonstrate the flexibility and versatile application of the Gábor transform by modifying the parameters of the Gaussian window function, and by utilizing another window function, the Mexican Hat wavelet. By using the Gábor transform to conduct time-frequency analysis, we can control which aspects of signal we would like to collect, analyze, and precisely manipulate.

Appendix A:
fft() – Fast Fourier Transform
pcolor() – Produces a colormap used for a spectrogram
colormap(hot) – Provides a colormap for a spectrogram where black is zero intensity to red to yellow and eventually white at the highest intensity.
Repmat() – Repeats copies of an array

Appendix B:
```
%% PART 1
clear all; close all; clc;
load handel

% Signal of interest, v(n)
v = y';

% Number of sampled points
n = length(v);

% Length of the signal as determined by number of sampled points divided
% by the rate at which they were sampled
L = n/Fs;

% Time component of the signal
v_time_span2 = linspace(0, L, n+1);
v_time_span = v_time_span2(1:n);

%% Frequency Analysis
% Frequency components
k = (2*pi/L)*[0:(length(v)-1)/2 -(length(v)-1)/2:-1];

% Shifted freq components to be mathematically correct; scaled for Hz
% instead of angular frequencies
k_shift_v = fftshift(k)/(2*pi);

% FFT of signal of interest
v_fft = fft(v);

% Plot the signal of interest's amplitude as a function of time
figure(1)
subplot(2,2,1)
plot(v_time_span,v,'k','Linewidth',1); axis([0 L -1 1])
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('v(n)'), title("SOI 1 in
Time Domain");

% Plot the signal of interest's FFT
subplot(2,2,3)
plot(k_shift_v,abs(fftshift(v_fft))/max(abs(v_fft)),'r','Linewidth',1);
axis([-4100 4100 0 1])
set(gca,'Fontsize',12)
xlabel('Frequency (Hz)'), ylabel('FFT(v)'), title("SOI 1 in Frequency
Domain")

% Plot the signal of interest's spectrogram
```

```
subplot (2,2,[2,4])
% Time component of the signal of interest
tslide = 0:0.1:L;
% Frequency component of the signal of interest remapped to an array that
% matches the time component's length
v_fft_spec = repmat(fftshift(abs(v_fft)),length(tslide),1);
pcolor(tslide,k_shift_v,v_fft_spec.');
shading interp
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)'),
title('Spectrogram from SOI 1')
colormap(hot), colorbar

%% Gabor window added to time domain plot
% Gabor transform parameters:
% Time steps at which the filter is applied
tau = 4;
% Window width
a = 1;
% Filter definition
g = exp(-a*(v_time_span-tau).^2);
% Time components of the filter
tslide = 0:tau:L;
% Allocate space for spectrogram
vg_fft_spec = zeros(length(tslide), length(v));

figure(2)
%Calculate the spectrogram matrix
for j=1:length(tslide)
    g=exp(-a*(v_time_span-tslide(j)).^2);
    vg=g.*v;
    subplot(2,2,1)
    plot(v_time_span,v,'k','Linewidth',1);
    set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('v(n)'), title("SOI
1 and Gabor Transform in Time Domain");
    hold on
    subplot(2,2,3)
    plot(v_time_span,vg,'m','Linewidth',1), axis([0 L -1 1]);
    set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('v(n)'),
title("Gabor Filtered SOI 1 in Time Domain")
    hold on
    vg_fft=fft(vg);
    vg_fft_spec(j,:) = fftshift(abs(vg_fft));
end

% Plotting the Gabor Transforms on top of SOI 1 in Time Domain
for j=1:length(tslide)
    g=exp(-a*(v_time_span-tslide(j)).^2);
    vg=g.*v;
    subplot(2,2,1)
    plot(v_time_span, g,'m','Linewidth',1); axis([0 L -1 1]);
end

% Plotting the spectrogram matrix
subplot(2,2,[2,4])
pcolor(tslide,k_shift_v,vg_fft_spec.'),
shading interp
```

```
title('Spectrogram of Gabor Filtered SOI 1'), xlabel('Time (sec)'),
ylabel('Frequency (Hz)')
set(gca,'Fontsize',12)
colormap(hot), colorbar


%% Gabor Transform and spectrogram - window width comparison
a = 100;

figure(3)
a_vec = [1 20 100];
for jj = 1:length(a_vec)
    a = a_vec(jj);
    tslide=0:0.1:L;
    vg_fft_spec = zeros(length(tslide),n);
    for j=1:length(tslide)
        g=exp(-a*(v_time_span-tslide(j)).^2);
        vg=g.*v;
        vg_fft=fft(vg);
        vg_fft_spec(j,:) = fftshift(abs(vg_fft));
    end

    subplot(2,2,jj)
    pcolor(tslide,k_shift_v,vg_fft_spec.'),
    shading interp
    title(['Spectrogram of Gabor Filtered SOI 1 with a =
',num2str(a)],'Fontsize',12)
    set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)')
    colormap(hot), colorbar
end

% Plot the signal of interest's spectrogram
subplot (2,2,4);
% Time component of the signal of interest
tslide = 0:0.1:L;
v_fft_spec = repmat(fftshift(abs(v_fft)),length(tslide),1);
pcolor(tslide,k_shift_v,v_fft_spec.');
shading interp
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)'),
title('Spectrogram from SOI 1')
colormap(hot), colorbar

%% Oversampling and Undersampling - Tau change
a = 20;

figure(4)
tau_over = 2;
tslide = 0:tau_over:L;
vg_fft_spec = zeros(length(tslide), length(v));
for j=1:length(tslide)
    g=exp(-a*(v_time_span-tslide(j)).^2);
    vg=g.*v;
    vg_fft=fft(vg);
    vg_fft_spec(j,:) = fftshift(abs(vg_fft));
end
```

```
subplot(2,1,1)
pcolor(tslide,k_shift_v,vg_fft_spec.'),
shading interp
title(['Spectrogram of Gabor Filtered SOI 1 with tau =
',num2str(tau_over)],'Fontsize',12)
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)')
colormap(hot), colorbar

tau_under = 0.1;
tslide = 0:tau_under:L;
vg_fft_spec = zeros(length(tslide), length(v));
for j=1:length(tslide)
    g=exp(-a*(v_time_span-tslide(j)).^2);
    vg=g.*v;
    vg_fft=fft(vg);
    vg_fft_spec(j,:) = fftshift(abs(vg_fft));
end

subplot(2,1,2)
pcolor(tslide,k_shift_v,vg_fft_spec.'),
shading interp
title(['Spectrogram of Gabor Filtered SOI 1 with tau =
',num2str(tau_under)],'Fontsize',12)
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)')
colormap(hot), colorbar

%% Mexican hat
tau = 0.1;
t_slide = 0:tau:L;
sigma = 1; %width paramater
vm_fft_spec = zeros(length(tslide), length(v));
for j=1:length(tslide)
    m_hat = (1-((v_time_span-tslide(j))/sigma).^2).*exp((-(v_time_span-
tslide(j)).^2)/(2*sigma.^2));
    vm=m_hat.*v;
    vm_fft=fft(vm);
    vm_fft_spec(j,:) = fftshift(abs(vm_fft));
end

figure(5)

pcolor(tslide,k_shift_v,vg_fft_spec.'),
shading interp
title('Spectrogram of SOI 1 filtered with Mexican Hat Wavelet','Fontsize',12)
set(gca,'Fontsize',12), xlabel('Time (sec)'), ylabel('Frequency (Hz)')
colormap(hot), colorbar

%% PART 2
clear all; close all; clc;

[y1,Fs] = audioread('music1.wav');
tr_piano=length(y1)/Fs; % record time in seconds

[y2,Fs] = audioread('music2.wav');
tr_rec=length(y2)/Fs; % record time in seconds
```

```
% Signal of interest 2
soi2=y1';

% Signal of interest 3
soi3=y2';

% Number of sampled points from SOI 2
n2=length(y1);

% Number of sampled points from SOI 3
n3=length(y2);

% Time component of SOI 2
t_soi2_2 = linspace(0, tr_piano, n2+1);
t_soi2 = t_soi2_2(1:n2);

% Time component of SOI 3
t_soi3_2 = linspace(0, tr_rec, n3+1);
t_soi3 = t_soi3_2(1:n3);

% Frequency components
k2 = (1/tr_piano)*[0:n2/2-1 -n2/2:-1];
k3 = (1/tr_rec)*[0:n3/2-1 -n3/2:-1];

% Shifted freq components to be mathematically correct; scaled for Hz
% instead of angular frequencies
k2_shift = fftshift(k2);
k3_shift = fftshift(k3);
%% With Overtones
% Gabor filter
tau = 0.1;
a = 100;

tslide2 = 0:tau:tr_piano;
soi2g_fft_spec = zeros(length(tslide2), n2);
for j=1:length(tslide2)
    g=exp(-a*(t_soi2-tslide2(j)).^2);
    soi2g=g.*soi2;
    soi2g_fft=fft(soi2g);
    soi2g_fft_spec(j,:) = fftshift(abs(soi2g_fft));
end

figure(6)
subplot(2,1,1)
pcolor(tslide2,k2_shift,soi2g_fft_spec.'),
shading interp
title('Spectrogram of Gabor Filtered SOI 2 with Overtones'), xlabel('Time
(sec)'), ylabel('Frequency (Hz)')
set(gca,'Ylim', [0 1000],'Fontsize',12)
colormap(hot), colorbar

tau = 0.1;
tslide3 = 0:tau:tr_rec;
soi3g_fft_spec = zeros(length(tslide3), n3);
```

```
for j=1:length(tslide3)
    g=exp(-a*(t_soi3-tslide3(j)).^2);
    soi3g=g.*soi3;
    soi3g_fft=fft(soi3g);
    soi3g_fft_spec(j,:) = fftshift(abs(soi3g_fft));
end
subplot(2,1,2)
pcolor(tslide3,k3_shift,soi3g_fft_spec.'),
shading interp
title('Spectrogram of Gabor Filtered SOI 3 with Overtones'), xlabel('Time
(sec)'), ylabel('Frequency (Hz)')
set(gca,'Ylim', [0 3100],'Fontsize',12)
colormap(hot), colorbar

%% Filtering Overtones
% Gabor filter
tau = 0.1;
a = 100;

% Gaussian filter parameters
tau_gaus = 0.2;

tslide2 = 0:tau:tr_piano;
soi2g_fft_spec = zeros(length(tslide2), n2);
for j=1:length(tslide2)
    g=exp(-a*(t_soi2-tslide2(j)).^2);
    soi2g=g.*soi2;
    soi2g_fft=fft(soi2g);
    [maxfreq, maxindex] = max(abs(soi2g_fft(:)));
    center_freq = k2(maxindex);
    gausfilter = exp(-tau_gaus*(((k2-center_freq).^2)));
    filtered_soi2g_fft = soi2g_fft.*gausfilter;
    soi2g_fft_spec(j,:) = fftshift(abs(filtered_soi2g_fft));
end

figure(7)
subplot(2,1,1)
pcolor(tslide2,k2_shift,soi2g_fft_spec.'),
shading interp
title('Spectrogram of Gabor Filtered SOI 2 with Overtones Removed'),
xlabel('Time (sec)'), ylabel('Frequency (Hz)')
set(gca,'Ylim', [220 340],'Fontsize',12)
colormap(hot), colorbar

% Gaussian filter parameters
tau_gaus = 0.01;

tslide3 = 0:tau:tr_rec;
soi3g_fft_spec = zeros(length(tslide3), n3);
for j=1:length(tslide3)
    g=exp(-a*(t_soi3-tslide3(j)).^2);
    soi3g=g.*soi3;
    soi3g_fft=fft(soi3g);
    [maxfreq, maxindex] = max(abs(soi3g_fft(:)));
    center_freq = k3(maxindex);
    gausfilter = exp(-tau_gaus*(((k3-center_freq).^2)));
    filtered_soi3g_fft = soi3g_fft.*gausfilter;
    soi3g_fft_spec(j,:) = fftshift(abs(filtered_soi3g_fft));
end
subplot(2,1,2)
pcolor(tslide3,k3_shift,soi3g_fft_spec.'),
shading interp
title('Spectrogram of Gabor Filtered SOI 3 with Overtones Removed'),
xlabel('Time (sec)'), ylabel('Frequency (Hz)')
set(gca,'Ylim', [700 1100],'Fontsize',12)
colormap(hot), colorbar
```