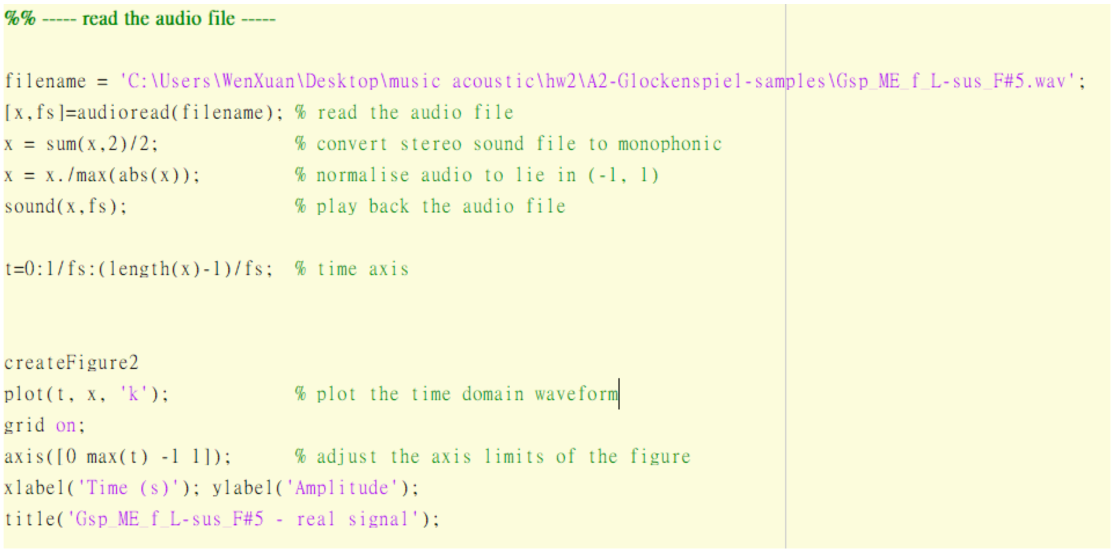
Music Acoustic - Assignment 2

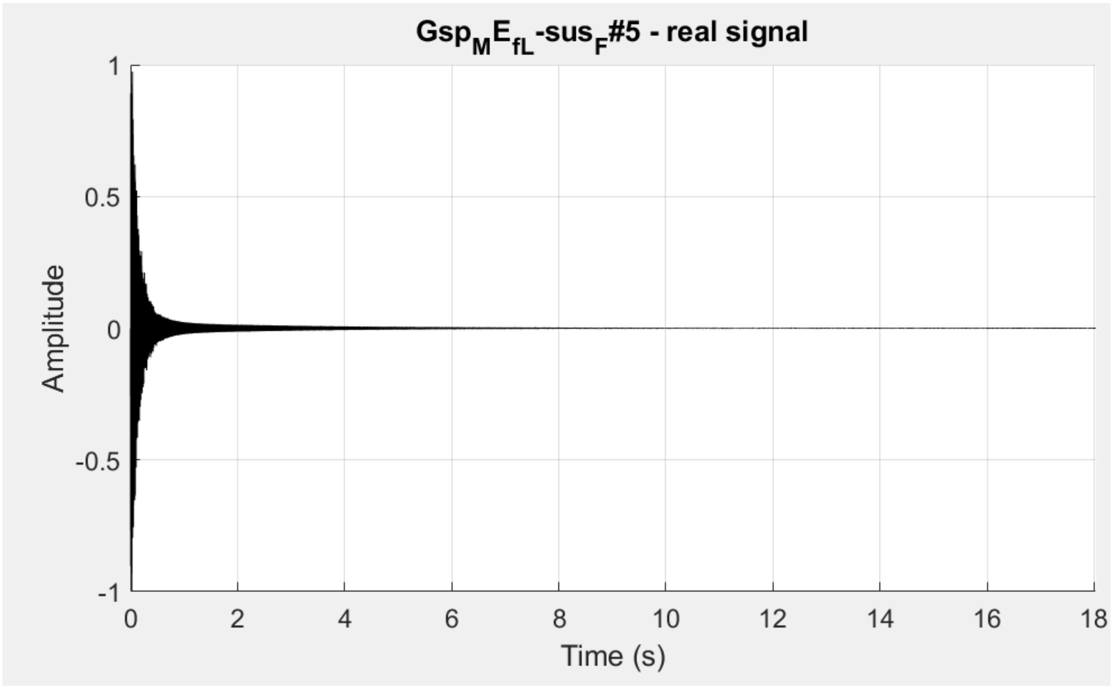
I analyzed and modeled these two samples, "Gsp_ME_f_L-sus_F#5.wav" and "Gsp_ME_p_L-sus_F#6.wav", in this assignment. Their Matlab scripts are "F_sharp5.m" and "F_sharp6.m" respectively.

First of all, I read the audio file and converted them from stereo sound to monophonic sound. Before plotting the time domain waveform, I normalized the audio data to lie in (-1, 1).

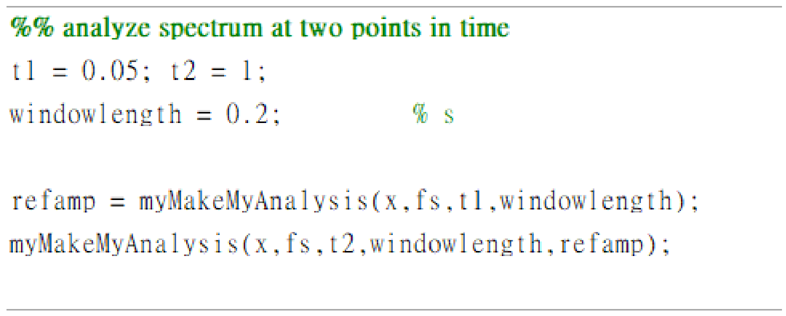


```
%% ----- read the audio file -----

filename = 'C:\Users\WenXuan\Desktop\music acoustic\hw2\A2-Glockenspiel-samples\Gsp_ME_f_L-sus_F#5.wav';
[x,fs]=audioread(filename); % read the audio file
x = sum(x,2)/2;             % convert stereo sound file to monophonic
x = x./max(abs(x));         % normalise audio to lie in (-1, 1)
sound(x,fs);                % play back the audio file

t=0:1/fs:(length(x)-1)/fs;  % time axis


createFigure2
plot(t, x, 'k');            % plot the time domain waveform
grid on;
axis([0 max(t) -1 1]);      % adjust the axis limits of the figure
xlabel('Time (s)'); ylabel('Amplitude');
title('Gsp_ME_f_L-sus_F#5 - real signal');
```

The output time domain waveform is like below.

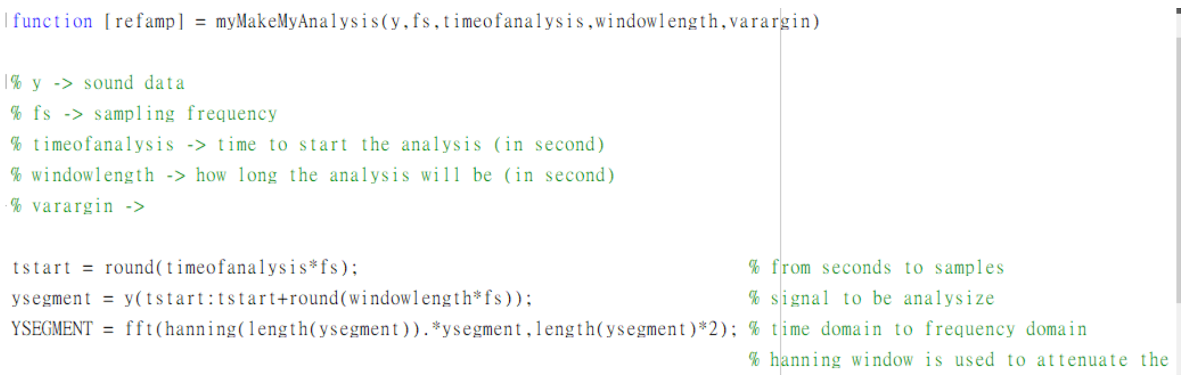


I drew the spectrums for two time intervals, (0.05, 0.25) and (1, 1.2).



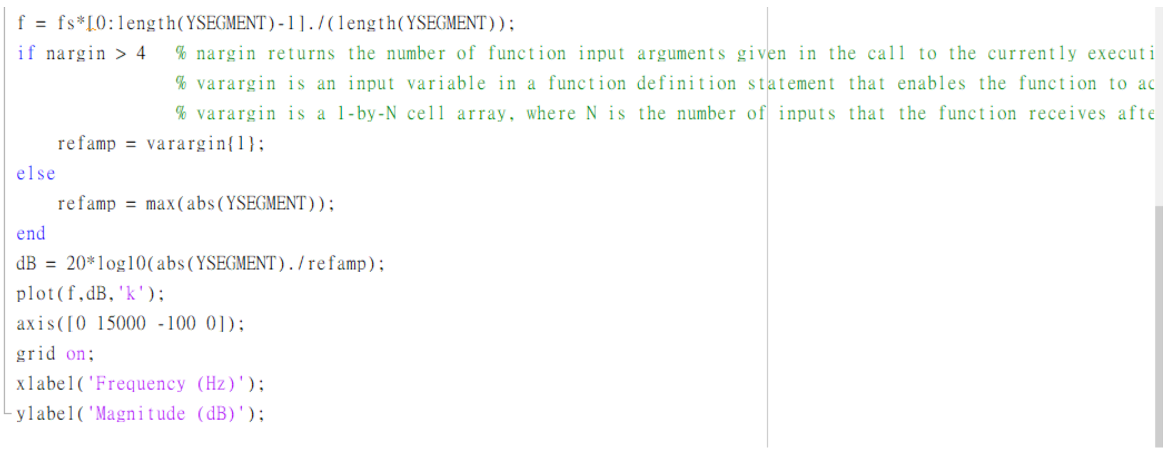
```
%% analyze spectrum at two points in time
t1 = 0.05; t2 = 1;
windowlength = 0.2;          % s

refamp = myMakeMyAnalysis(x,fs,t1,windowlength);
myMakeMyAnalysis(x,fs,t2,windowlength,refamp);
```

In the function, myMakeMyAnalysis, I made a FFT analysis in which a hanning window is multiplied to attenuate the sides of the window of the signal. This is done because transients at the limits of the window will produce undesired artifacts in the spectrum.



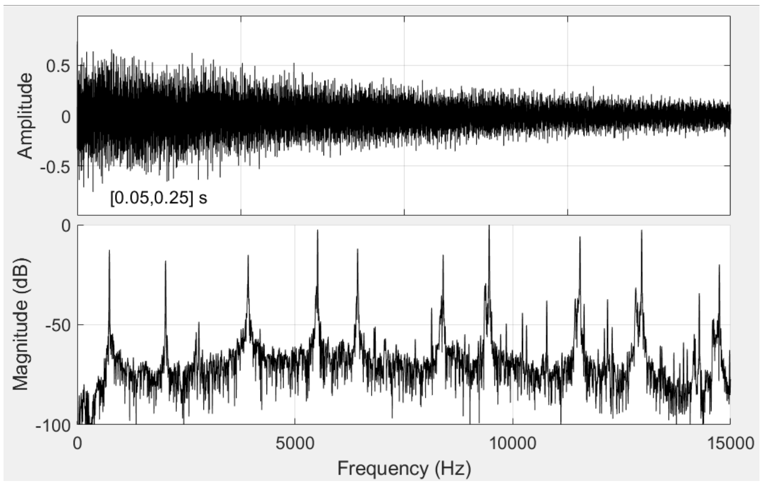
```
function [refamp] = myMakeMyAnalysis(y,fs,timeofanalysis,windowlength,varargin)

% y -> sound data
% fs -> sampling frequency
% timeofanalysis -> time to start the analysis (in second)
% windowlength -> how long the analysis will be (in second)
% varargin ->

tstart = round(timeofanalysis*fs);                                    % from seconds to samples
ysegment = y(tstart:tstart+round(windowlength*fs));                   % signal to be analysize
YSEGMENT = fft(hanning(length(ysegment)).*ysegment,length(ysegment)*2); % time domain to frequency domain
                                                                      % hanning window is used to attenuate the
```

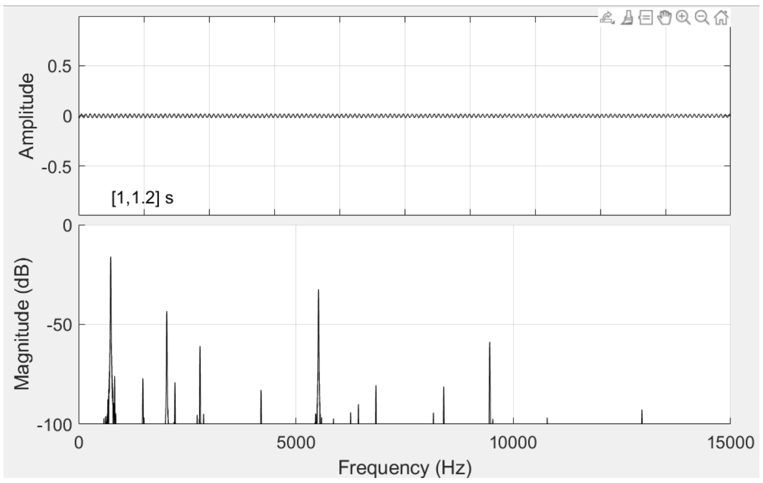
Also, the y-axis of the spectrum is computed to dB values.



```
f = fs*[0:length(YSEGMENT)-1]./(length(YSEGMENT));
if nargin > 4   % nargin returns the number of function input arguments given in the call to the currently executi
                % varargin is an input variable in a function definition statement that enables the function to ac
                % varargin is a 1-by-N cell array, where N is the number of inputs that the function receives afte
    refamp = varargin{1};
else
    refamp = max(abs(YSEGMENT));
end
dB = 20*log10(abs(YSEGMENT)./refamp);
plot(f,dB,'k');
axis([0 15000 -100 0]);
grid on;
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
```

The generated spectrum is as below. The upper subfigure is the time domain waveform of the selected time interval and the lower subfigure is the frequency domain spectrum which y-axis is dB value.





Through observing the peaks in these two spectrums, we can easily estimate each overtone frequency and its dB value. Then we can use these two formulas to compute the exponential decay parameter and the amplitude for each overtone.

Solving for the exponential decay parameter gives:

$$\gamma_n = \frac{\log_e 10}{20}\frac{\mathrm{dB}_n(t_1) - \mathrm{dB}_n(t_2)}{t_2 - t_1}. \tag{8}$$

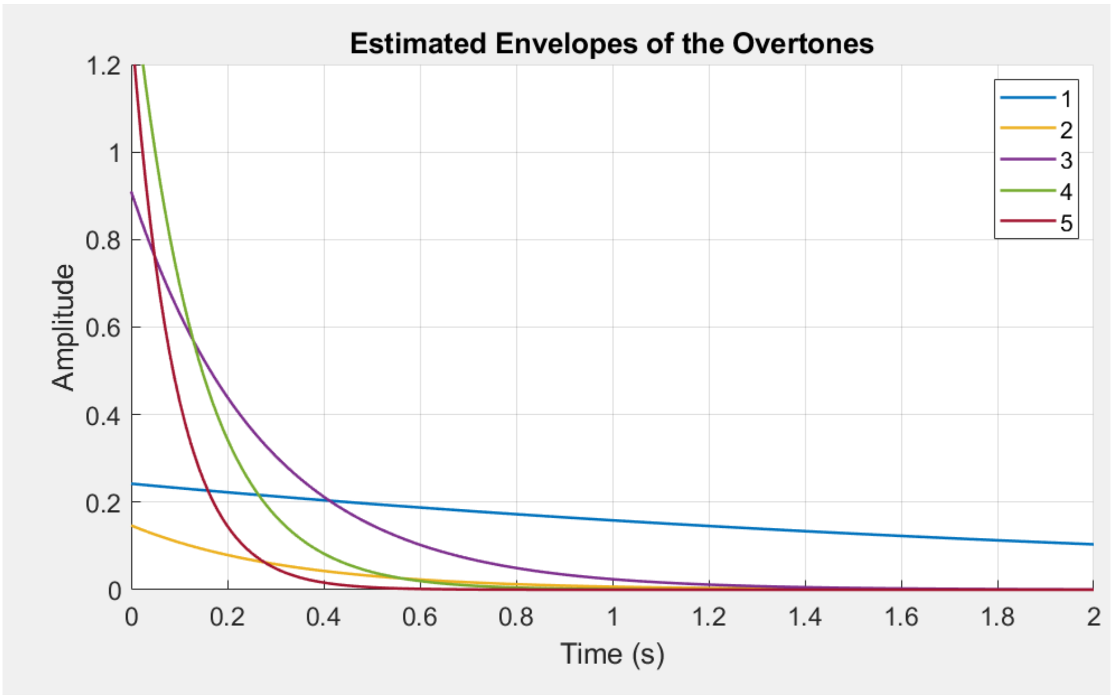
Having computed γ_n, we can then plug it into (5) or (6) and solve for the amplitude:

$$A_n = 10^{\mathrm{dB}_n(t_1)/20} 10^{\gamma_n t_1/\log_e 10}. \tag{9}$$

```
%%
% I took the following measurements from the above magnitude spectra
fest = [740 2030 5519 9459 12959];
dB1 = [-12.5 -18 -2.4 0 -2.5];
dB2 = [-16 -43.4 -32.4 -58.8 -92.8];

% estimate parameters
gammaest = (dB1-dB2)./(20/log(10))/(t2-t1);
Ampest = 10.^(dB1/20 + gammaest*t1/log(10));
```

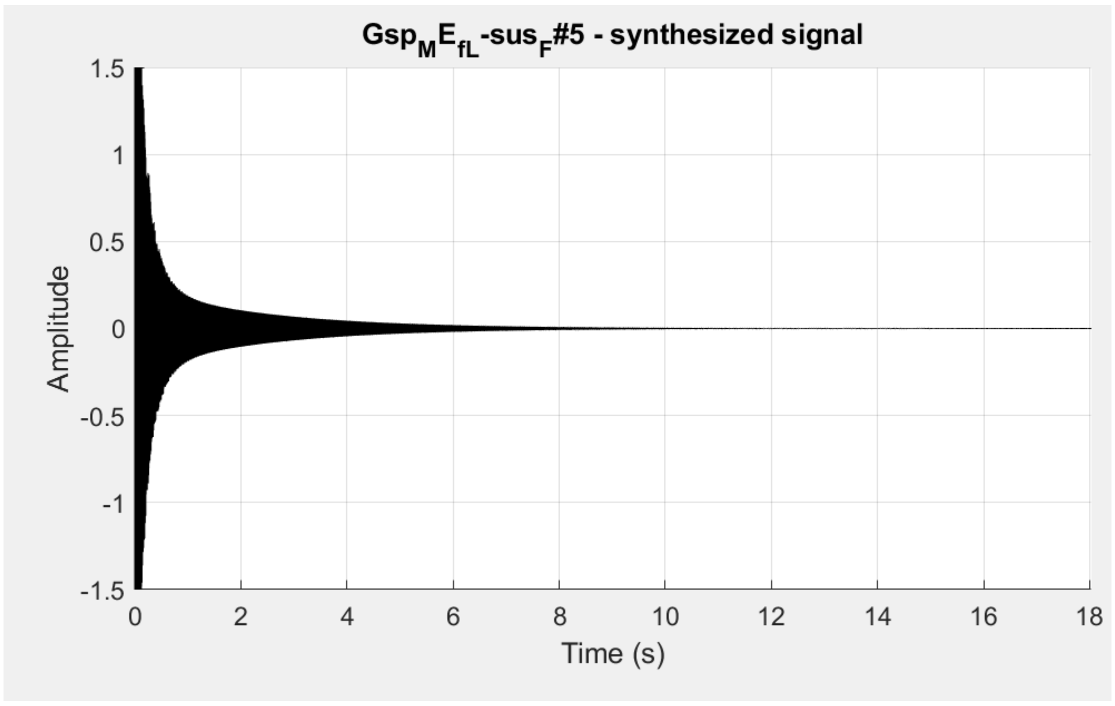
After knowing the exponential decay parameter and the amplitude, I can compute the envelope for each overtone. Below is the estimated envolope for each overtone.

```
createFigure2
t = [0:0.01:2];
colors = ["#0072BD" "#EDB120" "#7E2F8E" "#77AC30" "#A2142F"];
for ii=1:length(Ampest)
    env = Ampest(ii)*exp(-gammaest(ii)*t);
    plot(t,env, 'color', colors(ii), 'Linewidth', 1.5);
    % gtext(num2str(ii),'FontSize',16)
end
```



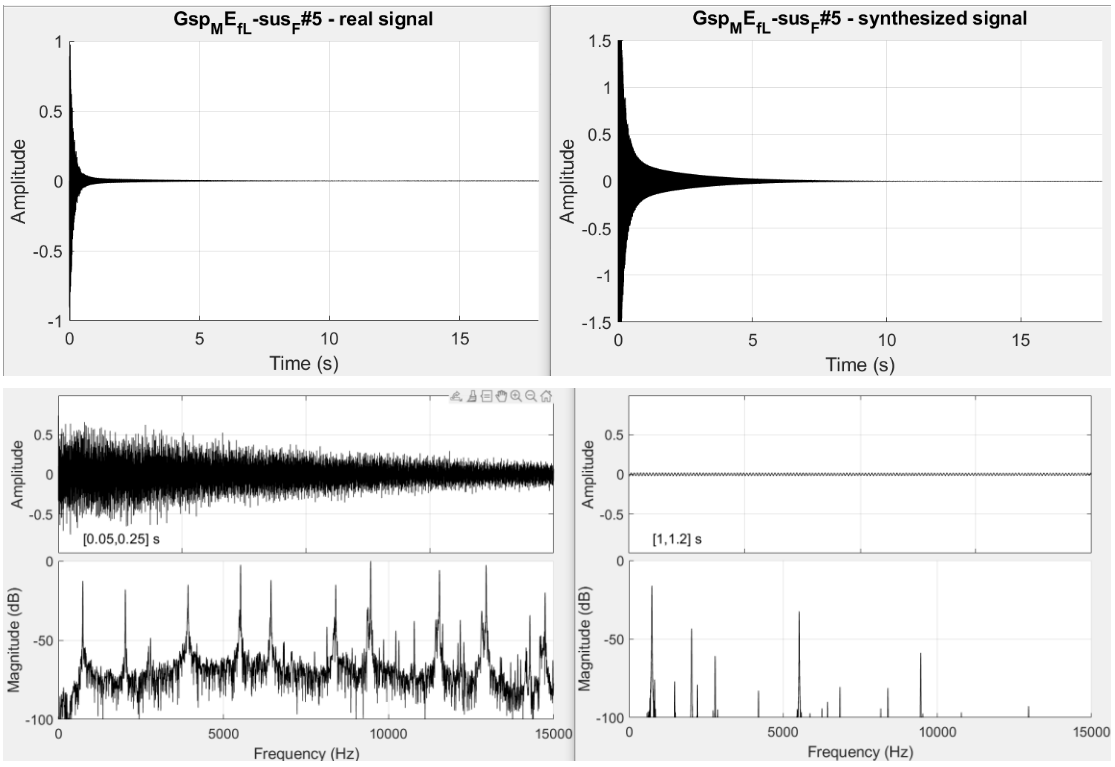
With the envelopes and overtones, we can make our additive model to synthesize this sound. The result is as below.

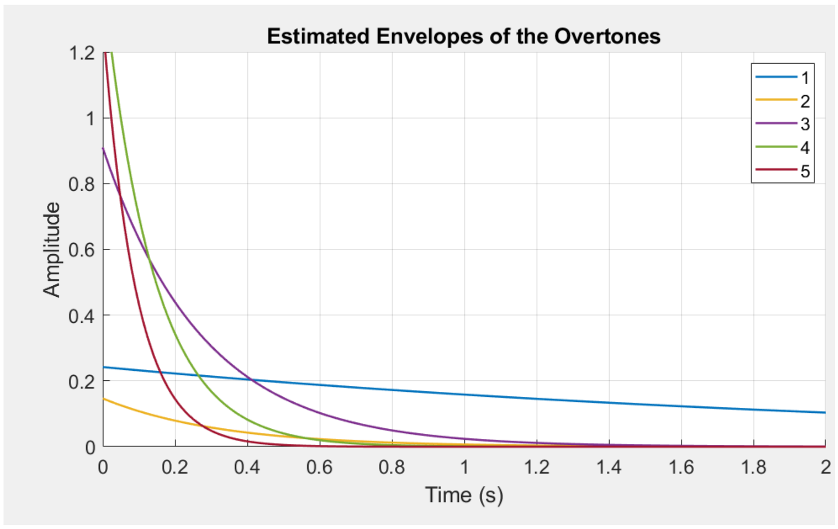
```
dur = (length(x)-1)/fs;
t = [0:dur*fs-1]'/fs;
xsynth = zeros(length(t),1);
for ii=1:length(Ampest)
    xsynth = xsynth + Ampest(ii)*exp(-gammaest(ii)*t).*sin(2*pi*t*fest(ii));
end
```



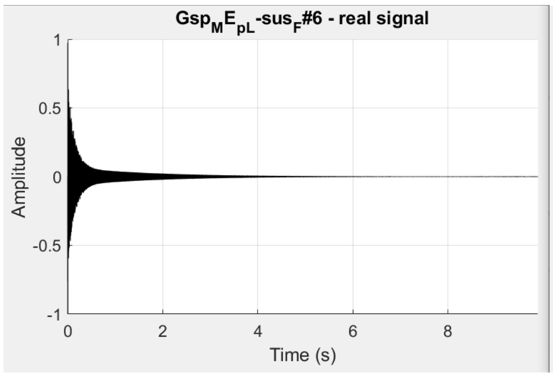
Comparison between the Audio Samples and the Synthesized Signals

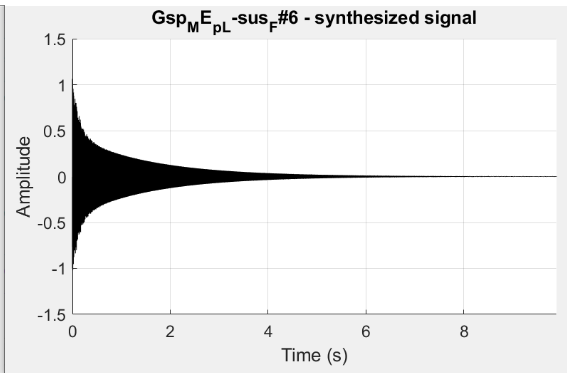
1. Result of F_sharp5 (hard)

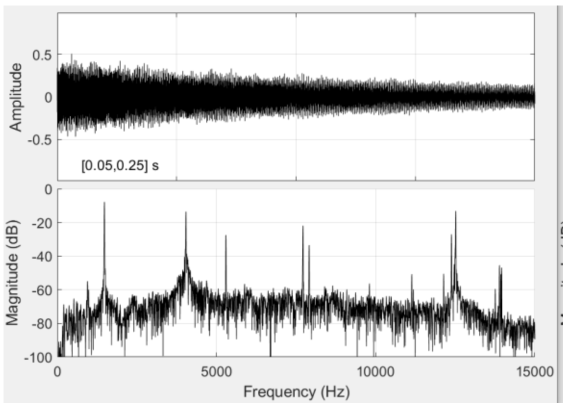


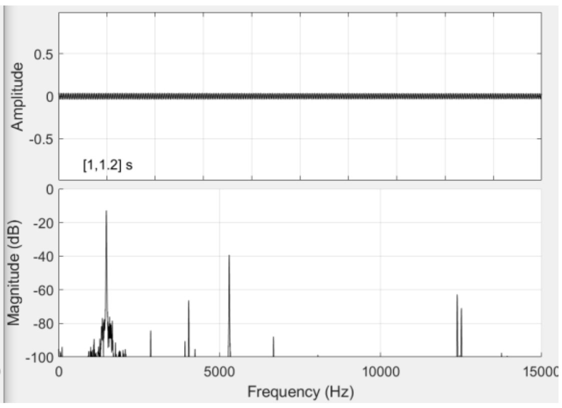


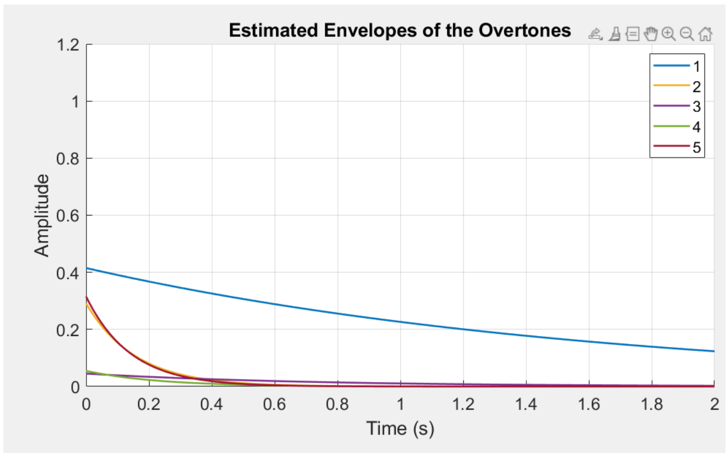
2. Result of F_sharp6 (soft)











When selecting the five partials having the most energy just after the attack below 15kHz to use in synthesis, I find that it is a little bit difficult to select the five partials which are prominent at both 0.05 second and 1 second after the attack because their decay rate is not the same. Sometimes, they are prominent at 0.05 seconds but not prominent at all at 1 second. Thus, my selected partials are not actually the first five partials having the most energy at 0.05 seconds. I chose the ones prominent at both 0.05 second and 1 second.

From the results of both F_sharp5 and F_sharp6, I find that the waveform of the synthesized signals always sustain longer than the original samples. I think this may be because of my selection of partials which are prominent at both 0.05 and 1 second after the attack. Also, from the graph of envelopes, I find the envelope of the fundamental frequency decays especially slower than the other partials, which reflects the theory.

Also, because the original samples consist of much more partials than the synthesized signals, the original samples sound better than the synthesized signals, especially for f_L-sus_F#5. For the synthesized f_L-sus_F#5, there is a little bit of roughness at the beginning. I may not describe how it sounds accurately, but this synthesized signal gives me a hoarse feeling. I think this may be improved by choosing other partials to make a synthesis: choose the ones that actually have the most energy just after the attack, because the obvious roughness is at the beginning of the sound. It may be improved by using more partials to synthesize the sound.