# Matt Bachmeier

## Table of Contents



ECE 203 Lab 4 Part 2: Synthesis of Sinusoidal Signals -- Music Synthesis 2/24/2017

# Section 4.3.1 (Timing)

```
close all
clear all
load bach_fugue.mat;
bpm = 80;
beats_per_second = bpm/60;
seconds_per_beat = 1/beats_per_second;
seconds_per_pulse = seconds_per_beat/4;

fs = 11025;
% This sampling frequency is high enought because even the third
% harmonic will not get aliased because fs > 2*fmax
```

# Section 4.4 (ADSR and Envelope)

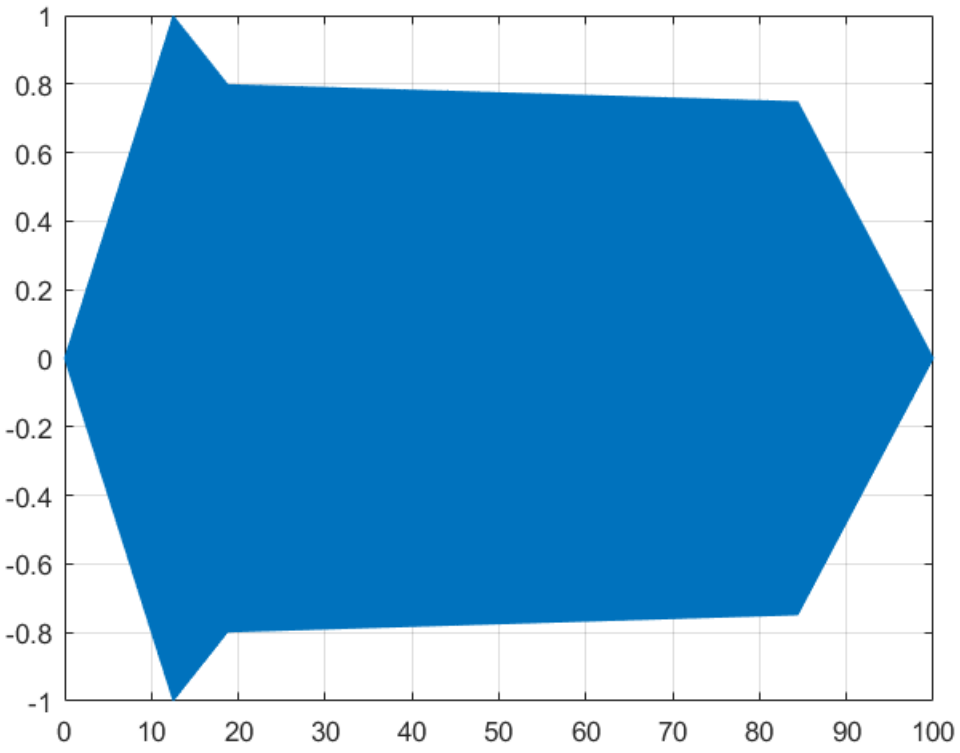
```
% arbitrary numbers chosen to create the plot of the envelope being
 used
fs = 11025;
dur = 100;
tt = 0:1/fs:dur;
wave = real(exp(j*2*pi*510*tt)); % random wave
length_of_tone = length(wave);

% approximations on what the evelope should look like based off of the
% picture
attack = linspace(0,1,length_of_tone*4/32);
delay = linspace(1,.8,length_of_tone*2/32);
sustain = linspace(.8,.75,length_of_tone*21/32);
release = linspace(.75,0,length_of_tone*5/32);

envelope = [attack,delay,sustain,release];

difference = length_of_tone - length(envelope);
envelope = [envelope, zeros(1,difference)];
envelope = envelope.*wave;

plot(tt, envelope), grid on
```




# Section 4.5

```
theVoices(1)
theVoices(2)
theVoices(3)

% The data in the voices holds the key number that is being played at
 a
% certain pulse in the song and how many pulses that note is held

tot_samp = 0;

% loop through the three voices
for k = 1:length(theVoices)

    % loop through the notes
    for p = 1:length(theVoices(k).durations)

        % find the max number of samples from the voices
        tot_samp = max(tot_samp, (sum(theVoices(k).durations(p) +
 theVoices(k).startPulses(p))*seconds_per_pulse*fs));
    end
end
```

```
% determine the total number of samples to create an array of the
 correct
% size
tot_samp = ceil(tot_samp);

% the following builds the voices and puts them together
% BuildVoice and BuildMusic
xx = zeros(1,tot_samp);
xx1 = zeros(1, 135);
xx2 = zeros(1, 135);
xx3 = zeros(1, 135);

for i = 1:length(theVoices(1).durations)

        keynumx = theVoices(1).noteNumbers(i);
        startpulsex =
 round(theVoices(1).startPulses(i)*seconds_per_pulse*fs);
        durationx = theVoices(1).durations(i)*seconds_per_pulse;

        xxp = key2noteModified(1,keynumx,durationx,fs);
        tt = 0:(1/fs):durationx;

        stopx = startpulsex + length(xxp) - 1;
        xx1(startpulsex:stopx) = xxp + xx1(startpulsex:stopx);

end

    % plots each voices wave form
    plot(xx1)
    title(['Wave form of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Amplitude')

    %plots each voices spectrogram
    specgram(xx1)
    title(['Spectrogram of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Frequency')

for i = 1:length(theVoices(2).durations)

    keynumx = theVoices(2).noteNumbers(i);
    startpulsex =
 round(theVoices(2).startPulses(i)*seconds_per_pulse*fs);
    durationx = theVoices(2).durations(i)*seconds_per_pulse;

    xxp = key2noteModified(1,keynumx,durationx,fs);
    tt = 0:(1/fs):durationx;

    stopx = startpulsex + length(xxp) - 1;
    xx2(startpulsex:stopx) = xxp + xx2(startpulsex:stopx);

end
```

```
    % plots each voices wave form
    plot(xx2)
    title(['Wave form of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Amplitude')

    %plots each voices spectrogram
    specgram(xx2)
    title(['Spectrogram of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Frequency')

for i = 1:length(theVoices(3).durations)

    keynumx = theVoices(3).noteNumbers(i);
    startpulsex =
 round(theVoices(3).startPulses(i)*seconds_per_pulse*fs);
    durationx = theVoices(3).durations(i)*seconds_per_pulse;

    xxp = key2noteModified(1,keynumx,durationx,fs);
    tt = 0:(1/fs):durationx;

    stopx = startpulsex + length(xxp) - 1;
    xx3(startpulsex:stopx) = xxp + xx3(startpulsex:stopx);

end

    % plots each voices wave form
    plot(xx3)
    title(['Wave form of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Amplitude')

    %plots each voices spectrogram
    specgram(xx3)
    title(['Spectrogram of voice ' num2str(q)]);
    xlabel('Time')
    ylabel('Frequency')


xx = xx1 + xx2 +xx3;

%for q = 1:length(theVoices)

 %   for i = 1:length(theVoices(q).durations)

  %      keynumx = theVoices(q).noteNumbers(i);
   %     startpulsex =
 round(theVoices(q).startPulses(i)*seconds_per_pulse*fs);
    %    durationx = theVoices(q).durations(i)*seconds_per_pulse;

     %   xxp = key2noteModified(1,keynumx,durationx,fs);
      %  tt = 0:(1/fs):durationx;
```

```
        % stopx = startpulsex + length(xxp) - 1;
         %xx(startpulsex:stopx) = xxp + xx(startpulsex:stopx);

    %end

    % plots each voices wave form
    %plot(xx)
    %title(['Wave form of voice ' num2str(q)]);
    %xlabel('Time')
    %ylabel('Amplitude')

    %plots each voices spectrogram
    %specgram(xx)
    %title(['Spectrogram of voice ' num2str(q)]);
    %xlabel('Time')
    %ylabel('Frequency')

%end

% the key2note used to construct the voices and music together. Very
% similar to what was put in section 4.4

%function xx = key2noteModified(X, keynum, dur,fs)
% KEY2NOTE Produce a sinusoidal waveform corresponding to a
%       given piano key number
%
%  usage: xx = key2note (X, keynum, dur)
%
%      xx = the output sinusoidal waveform
%       X = complex amplitude for the sinusoid, X = A*exp(j*phi).
%  keynum = the piano keyboard number of the desired note
%     dur = the duration (in seconds) of the output note
%

%if nargin < 4
%    fs = 11025;
%end
%tt = 0:(1/fs):dur;
%freq = 440*2^((keynum-49)/12);
%no_envelope = real(X*exp(j*2*pi*freq*tt) +
 0.75*X*exp(j*2*pi*2*freq*tt) + 0.5*X*exp(j*2*pi*3*freq*tt) +
 0.25*X*exp(j*2*pi*4*freq*tt));
%length_of_tone = length(no_envelope);
%attack = linspace(0,1,length_of_tone*4/32);
%delay = linspace(1,.8,length_of_tone*2/32);
%sustain = linspace(.8,.75,length_of_tone*21/32);
%release = linspace(.75,0,length_of_tone*5/32);

%envelope = [attack,delay,sustain,release];

%difference = length_of_tone - length(envelope);
%envelope = [envelope, zeros(1,difference)];
%xx = envelope.*no_envelope;
```

```
% Plays the music in matlab after it scales it to between 1 and -1
soundsc(xx,fs);


ans =

    startPulses: [1x123 double]
      durations: [1x123 double]
    noteNumbers: [1x123 double]


ans =

    startPulses: [1x130 double]
      durations: [1x130 double]
    noteNumbers: [1x130 double]


ans =

    startPulses: [1x88 double]
      durations: [1x88 double]
    noteNumbers: [1x88 double]


Index exceeds matrix dimensions.

Error in Lab_04Part2 (line 87)
        xx1(startpulsex:stopx) = xxp + xx1(startpulsex:stopx);
```

*Published with MATLAB® R2016a*