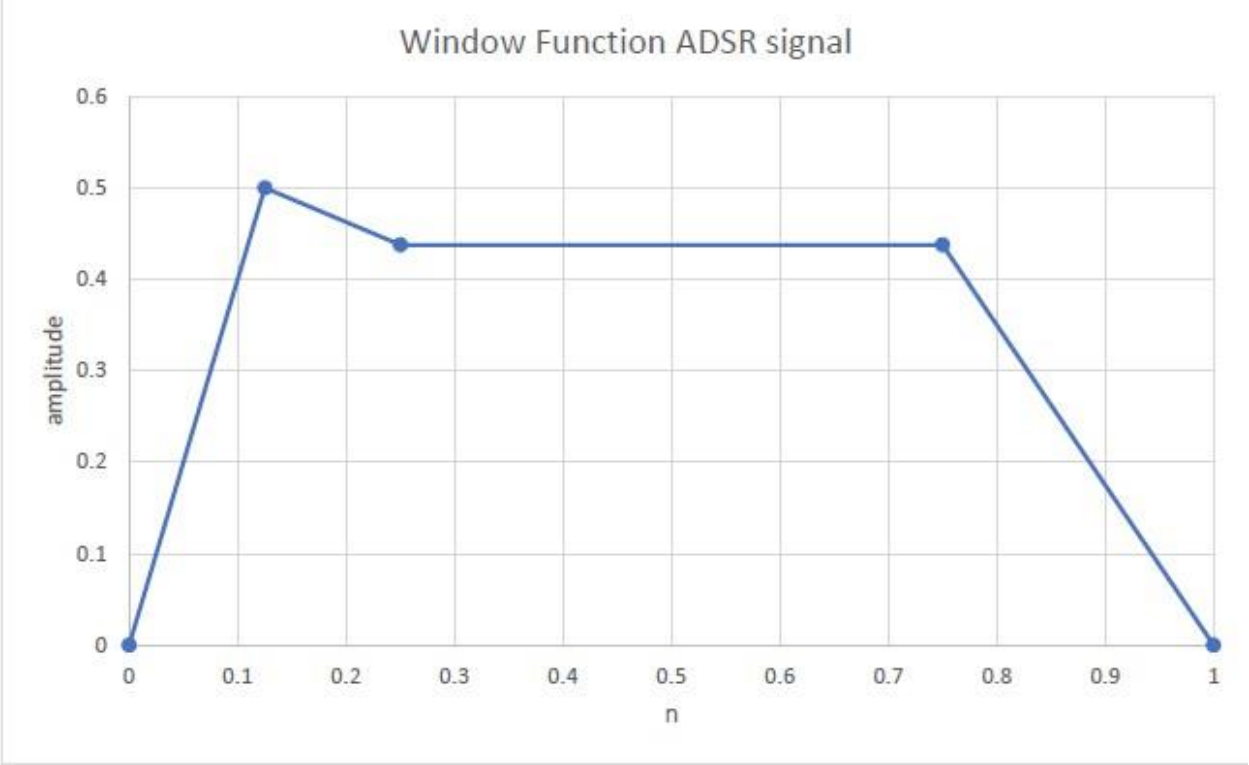
Zhimingyuan Liu LAB2 REPORT

This lab, we learned the signal transformation and synthesis skills we learned previously to create some basic sounds. By synthesizing the sound notes and apply some special signal processing method, we create relatively smooth audio to simulate instrumental and natural sounds.

In step 1, we create the different sound notes using cos wave. According to the frequency of each note, we first initialize all the notes separately. Then by concatenation, we synthesize the musical score sound.

In step 2, we extract each note and process it to imitate instrumental effects. To achieve this, we use a window function that modify the amplitude of the note to create and ASDR level difference. The ADSR signal will multiply in amplitude with the note for use. The ADSR signal in the window function is set to have following characteristics. The A part has 1/8 length of the note, the amplitude goes from 0 to 0.5 to imitate the start of a sound. Then D part has 1/8 length of the note, amplitude drops from 0.5 to 0.4375. The S part stabilize at 0.4375 for ½ length of the note. Finally, the R part drops from 0.4375 to 0 for ¼ length of the note. Following is a plot of it.



Window Function ADSR signal

And the overlap effect of neighbor notes is achieved by creating signals with all the same length as the whole song, but each signal only has one note while the rest of the length are 0 in amplitude. By moving them into certain overlap position and summing them in amplitude, we get the overlap effect audio signal. We use 1000 samples for overlapping range of each neighbor note.

In step 3. First, we multiply the noise signal with a 1000-frequency cosine wave in amplitude to create an ocean sound signal. During the modification, higher the frequency is, the more consistent the noise of ocean would be. When using low frequency, we tend to hear difference in amplitude more. But when the frequency is too low, the noise is more like artificial sound with amplitude level rather than a simulating natural sound of the sea. Basically, the noise of sea should have some minor

difference in amplitude level. So, I choose 1000 frequency as a balance of sound level and natural flavor. Then we multiply the signal with a decay exponential to further process the signal to sound more like natural one. The decay factor I use is -2, with an initial amplitude factor of 0.2. The smaller the decay factor in magnitude, the slower the sound will fade away. After testing multiple time, -2 provide the best result with reasonable length of sound. And the 0.2 amplitude factor is for writing the audio properly without clipping.

In step 4, we use time scaling to change how the audio sounds like. To make the tiger sound like cat, we want to speed up the audio to make the tiger less fierce. To speed up, we need a scaling factor which is bigger than 1. After test, I decided to use 5 as the scaling factor. This makes the tiger sound way gentler. And for my own signal, I used a quote from the TV series Heroes. After a little bit speed up with 1.4 factor, the quote sound funny while still audible.

```
% FILE: Assignment1and2.m
& DESCRIPTION: Create and synthesize the musical notes and
concatenating % into song signal. Apply sound effect of ASDR and
overlap to imptove  sound % quaility to imitate instrumental sound

% Clear all variables and close all
windows clear all; close all;

% Initialize the sample vector for differnt note lengths
Fs = 8000; Ts =
1/Fs; n_4 =
0:1:(0.5/Ts); n_2
= 0:1:(1/Ts); n =
0:1:(2/Ts);

% Produce the sound notes
A_2 = cos(2*pi*220*n_2*Ts);
A_4 = cos(2*pi*220*n_4*Ts);
E_4 = cos(2*pi*220*2^(7/12)*n_4*Ts);
B_4 = cos(2*pi*220*2^(2/12)*n_4*Ts);
C_4 = cos(2*pi*220*2^(3/12)*n_4*Ts);
A = cos(2*pi*220*n*Ts);

% Create space between notes
space = zeros(1,0.25*Fs);

% Synthesize song song =
[A_2,space,A_4,space,E_4,space,E_4,space,E_4,space,B_4,space,C_4,space,B_4,space,A

% Modify each note to have ASDR
effect mod_A2 = window(A_2); mod_A4 =
window(A_4); mod_E4 = window(E_4);
mod_B4 = window(B_4); mod_C4 =
window(C_4); mod_A = window(A);

% Produce long signal for summation into final overlap song
signal ex_0 = [mod_A2,zeros(1,4001*7+16001)]; ex_1 =
[zeros(1,7001),mod_A4,zeros(1,6*4001+17001)]; ex_2 =
[zeros(1,6001+4001),mod_E4,zeros(1,5*4001+18001)]; ex_3 =
[zeros(1,5001+4001*2),mod_E4,zeros(1,4*4001+19001)]; ex_4 =
[zeros(1,4001+4001*3),mod_E4,zeros(1,3*4001+20001)]; ex_5 =
[zeros(1,3001+4001*4),mod_B4,zeros(1,2*4001+21001)]; ex_6 =
[zeros(1,2001+4001*5),mod_C4,zeros(1,1*4001+22001)]; ex_7 =
[zeros(1,1001+4001*6),mod_B4,zeros(1,23001)]; ex_8 =
[zeros(1,1+4001*7),mod_A,zeros(1,8000)];
```

```
% Summation the overlap song signal mod_song =
ex_0+ex_1+ex_2+ex_3+ex_4+ex_5+ex_6+ex_7+ex_8;

% Play the final signal
sound(mod_song,Fs);

% Save the audio files
audiowrite('assignment1.wav',song,Fs);
audiowrite('assignment2.wav',mod_song,Fs);
```

Published with MATLAB® R2017a

```
% WINDOW: Generate ASDR signal based on the length of the input
% signal,
% then modify the input signal with the ASDR effect
% USASGE: mod = ASDR(x) outputs ASDR version of input signal
function mod = ADSR(x)

n = length(x)-1
A = linspace(0,2,n/8);
D = linspace(2,1.75,n/8);
S = linspace(1.75,1.75,n/2);
E = linspace(1.75,0,(n/4+1));

ADSR = [A,D,S,E]

mod = x.*ADSR
```

```
Not enough input arguments.

Error in ADSR (line 7)
n = length(x)-1
```

Published with MATLAB® R2017a

```
% FILE: Assignment3.m
% NAME: Zhimingyuan Liu
% DESCRIPTION: Create noise of ocean from random noise signal. Then
% modify it
% with decay effect.
% Clear all variables and close all windows
clear all; close all;
Fs = 8000;
ocean = randn(1, 10001);

% Initialize the random noise signal
t = 0:1/Fs:(10000/Fs);
y = cos(2*pi*1000*t);

% Produce the 500 frequency noise signal
y1 = ocean.*y;

sound(y1, Fs);
pause(5);

% Produce the decay version
y2 =0.2*exp(-2*t).*cos(2*pi*1000*t);
y3 = ocean.*y2;

sound(y3,Fs);
% Save the audio file
audiowrite('Assignment3.wav', y3,Fs);
```

Published with MATLAB® R2017a

```
% FILE: Assignment4.m
% NAME: Jiachen Zou
% DESCRIPTION: Using time scaling to tranform tiger sound to cat. Then
% modify autobotic voice with the same method.
% Clear all variables and close all windows
clear all;
close all;
% Read the tiger sound signal
[y,Fs] = audioread('tiger.wav');
% Speed up the signal by 5
[y1,t1] = timescale(y,Fs,5);
% Read the quote from Heroes
[y2,Fs2] = audioread('autobots-2.wav');
% Speed up the signal by 1.4
[y3,t3] = timescale(y2,Fs2,1.4);
% play the after-effect tiger sound
sound(y1,Fs);
pause(2);
sound(y3,Fs);
% Save the audio files
audiowrite('assignment4_cat.wav',y1,Fs);
audiowrite('assignment4_heroes.wav',y3,Fs2);
```

```
Warning: Data clipped when writing file.
```

Published with MATLAB® R2017a

```
% TIMESCALE: Scale the input signal by the given factor in time
% USAGE: [y1, t1] = timescale(signal,Fs, factor);

function [y, t] = timescale(x, Fs, a)

    % Get numerator and denominator of scaling factor
    [n, d] = rat(a);

    % rescale the time axis
    y = resample(x, d, n);

    % Produce time samples vector
    t = (0:length(y)-1) * (1/Fs);

end
```

```
Not enough input arguments.

Error in timescale (line 8)
    [n, d] = rat(a);
```

Published with MATLAB® R2017a