function [my\_saw,t] = my\_sawtooth(freq,fs)

if(nargin == 0)

fs = 48000; % default sampeling freq

freq = [261.6 329.6 392]; % default chord: c major

elseif(nargin == 1)

fs = 48000;

end

if(fs<2\*max(freq))

disp("ERROR: my\_sawtooth return an empty value")

disp("fs doesn't follow nyquist");

return

end

T = 1; % 1 sec sound signal

t = [0:1/fs:T - 1/fs]; % time grid

my\_saw\_func =@(f0) sawtooth(2\*pi\*f0\*t); % anonymous func to calculate sawtooth

my\_saw = 0;

for f\_i = freq

my\_saw = my\_saw + my\_saw\_func(f\_i); % calculate signal

end

sound(my\_saw,fs);

pause(1.2\*T);

% plot signal:

N = fs\*T;

saw\_f = fft(my\_saw,N);

sawtooth\_f\_mag = 20\*log(abs(saw\_f));

sawtooth\_f\_mag = sawtooth\_f\_mag - max(sawtooth\_f\_mag);

fgrid = fs\*(0:(N-1))/(N);

figure(1);

semilogx(fgrid,sawtooth\_f\_mag); grid on; axis([20, 20\*10^3 -100 0]);

title("Sawtooth signal Spectrum:"); xlabel("f[Hz]{\copyright} ROT"); ylabel("X(f)");

end

function [coeff\_mat,freq\_resp\_filt\_mat,fgrid] = octave\_filters(f0,fs)

% ALMOST IDENTICAL TO A CONST Q FAM SYSTEM

% THIS SYSTEM IS IMPLEMENTING A MATRIX OF OCTAVE FILTERS

% A FUNCTION TO BE CALLED FROM "octave\_analysis.m", to analyze all spectrun

% pieces around center frequency f0 and it's octaves.

if(nargin == 0)

fs = 48000; % function sets fs = 48kHz as default value (in case wasn't passed)

fc = 27.5; % function sets fc = 27.5 (the frequency corresponding to musical note A0) if wasn't passed

elseif(nargin == 1)

fs = 48000; % sampeling freq

fc = f0; % fc0

else

fc = f0;

end

f\_bottom = 20; % bottom of spec

f\_top = 20\*10^3; % top of spec

fc\_vec = fc\*2.^[0:9];

Q = 1/((2)^(1/2)); % butterworth quality factor;

N = fs; % number of evaluation points

freq\_resp\_filt\_mat = []; % a matrix to hold frequency response of each filter

coeff\_mat = []; % a matrix to hold each filter's coeffs: h0: b00 | b10 | b20...

clf; figure(1); % ----------------

fgrid = fs\*(0:(N-1))/(N); % a00 | a10 | a20...

hold on; % ----------------

xlabel("f[Hz] {\copyright}ROT"); ylabel("|H(f)|"); grid on; % h1: b01 | b11 | b21...

title("Octave-spread Butterworth Band Pass Filters"); axis([f\_bottom,f\_top,0,1]); % a01 | a11 | a21...etc.

for i = [1:length(fc\_vec)]

% DO NOT IMPELEMENT SYSTEMS WITH ALIASING:

if(fc\_vec(i)\*Q/fs < 0 || fc\_vec(i)\*Q/fs > 1 || fc\_vec(i)/(fs\*Q) < 0 || fc\_vec(i)/(fs\*Q) > 1)

break

end

[b,a] = butter(2,[fc\_vec(i)/Q, fc\_vec(i)\*Q]./fs,"bandpass");

coeff\_mat = [coeff\_mat;b;a]; % add vectors b,a to the coeff matrix

[h,~] = freqz(b,a,fs); % calc frequency response of filter

freq\_resp\_filt\_mat = [freq\_resp\_filt\_mat;h']; % add frequency resp vector to the filter matrix

plot(fgrid,abs(h),'black');

end

xline(fc\_vec,':'); yline(0.707, ':'); % mark center frequencies and -3db

hold off;

end

function octave\_analysis()

figure\_counter = 1;

c\_major\_freqs = [261.6 329.6 392];

fs = 48000;

T = 1;

[x,t] = my\_sawtooth(c\_major\_freqs,fs);

[coeff\_mat, freq\_resp, fgrid] = octave\_filters(27.5,fs);

[n,m] = size(coeff\_mat);

for i = [1:2:n]

figure(figure\_counter); subplot(211);plot(t,x); grid on;

y = filter(coeff\_mat(i,:),coeff\_mat(i+1,:),x);

figure(figure\_counter); subplot(212); plot(t,y); grid on;

sound(y,fs);

pause(1.1\*T);

end

end