function [coeff\_mat,freq\_resp\_filt\_mat,fgrid] = constant\_Q\_fam(f0,fs)

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

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

fc\_vec = fc\*2.^[0:9]; % building series of frequencies fc\_vec, where fc\_i = 2\*(fc\_i-1)

fb = fc\_vec./Q; % vector of each octave's bandwidth

f\_low = fc\_vec - fb/2; % low -3db freqs

f\_high = fc\_vec + fb/2; % high -3db freqs

N = fs; % number of evaluation points

%%%%% listen to the series fc\_vec %%%%%

% T = 1;

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

% for freq= fc\_vec

% sound(sin(2\*pi\*freq.\*t),fs);

% pause(T);

% end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

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("Constast-Q Band Pass Filters"); axis([0,2\*10^4,0,1]); % a01 | a11 | a21...etc.

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

% DO NOT IMPELEMENT SYSTEMS WITH ALIASING:

if(f\_high(i)/fs < 0 || f\_high(i)/fs > 1 || f\_low(i)/fs < 0 || f\_low(i)/fs > 1)

break

end

[b,a] = butter(2,[f\_high(i), f\_low(i)]./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