**Lab 1：Introduction**

|  |  |
| --- | --- |
| **Author** | Name：唐心宇lab6 Student ID:11911817  毕云天Lab5 12112501 |
| **Verification of used lowpass filter and bandpass filter:**  Below is the verification of filter generated by butter() used in tone-vocoder:  First is verification of low-pass filter:          Next is verification of bandpass filter:                      There is more, but too much. So, omit.  **Task4**  Following will introduce the effect of increasing cut-off frequency of low-pass filter in tone-vocoder when there is noise.  Will try cut-off frequency of 20Hz 50Hz 100Hz and 400Hz with number of bands of 6.      The graph above is spectrum of original signal and noisy signal.  The following graphs is the spectrum of noisy signal generated by the tone-vocoder  With N=6 and cut off frequency = 20 50 100 400Hz.          Analysis (Task 4):  The generated audio is hard to understand. Audio generated with higher cut-off frequency is clearer although there are still not understandable. There are 6 peeks in each side corresponding to N=6. With the cutoff frequency increasing, there appear more and more scattered signals between the peak, which happens in task2.  Application in real life (Task 4):  Increase cutoff frequency of low-pass filter can improve quality of generated sentence when there is noise.  **Conclusion:**  Audio generated with more band and higher cutoff frequency sound clearer. Although too much bands can lead to strange sound effect. When the sound has much noise and it’s hard to understand, the generated audio by tone-vocode is hard to understand too. | |
| **Experience：** | |
| **Score** | task4 唐心宇-11911817自我评分90 （了解了截止频率变化对耳蜗生成的影响 |

**Code:**

**Task4**

close all;clc;clear;

[x, fs] = audioread ("C\_01\_02.wav");

N = length(x);

f = linspace(-fs/2, fs/2, N);

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

but\_order = 4;

% SSN

noise = (1-2\*rand(N,1));

sig = repmat(x,10,1);

[Pxx,w] = pwelch(sig,[],[],512,fs);

b=fir2(3000,w/(fs/2),sqrt(Pxx/max(Pxx)));

[h, wh] = freqz(b,1,128);

noise = filter(b,1,noise);

% -5dB

Enoise = norm(noise);

Ex = norm(x);

Eknosie = 10^(1/4)\*Ex;

k = Eknosie/Enoise;

y = x + k\*noise;

% 归一化

Ey = norm(y);

m = Ey/Ex;

y\_Final = y./m;

figure

subplot(211);

x\_fft = fft(x)/length(x);

plot(f,abs(fftshift(x\_fft)));%%%

title('spectrum of original signal');

xlabel('freq/Hz');ylabel('magitude');

subplot(212);

[b,a]=butter(but\_order,100/(fs/2));

x\_enve=filter(b,a,abs(x));

plot(xt, x, xt, x\_enve);%%%

title('time domain of original signal');

xlabel('t');ylabel('original audio');

legend('时域','包络','Location', 'northeast' )

figure

subplot(211);

y\_Final\_fft = fft(y\_Final)/length(y\_Final);

plot(f,abs(fftshift(y\_Final\_fft)));%%%%

title('spectrum of SSN');

xlabel('freq/Hz');ylabel('magitude');

subplot(212);

[b,a]=butter(but\_order,100/(fs/2));

y\_Final\_enve=filter(b,a,abs(y\_Final));

plot(xt, y\_Final, xt, y\_Final\_enve);%%%%

title('time domain of SSN');

xlabel('t');ylabel('SSN audio');

legend('时域','包络','Location', 'northeast' )

n\_bands = 6;

for i = [20 50 100 400]

y = tone\_vocoder(y\_Final,n\_bands, i,but\_order,fs);

Y = fft(y)/length(y);

figure;

subplot(211);

[b,a]=butter(but\_order,100/(fs/2));

y\_enve=filter(b,a,abs(y));

plot((0:N-1)/fs, y, (0:N-1)/fs, y\_enve);

xlabel("t/s");

ylabel("audio");

legend('时域','包络','Location', 'northeast' );

title(sprintf('Q4 noisy signal N=%d f=%d Hz time domain',n\_bands, i));

subplot(212);

plot(f,abs(fftshift(Y)));

title(sprintf('Q4 noisy signal N=%d f=%d Hz spectrum',n\_bands, i));

xlabel('freq/Hz');

ylabel('magnitude');

audiowrite(sprintf('Q4 noisy signal N=%d f=%d Hz.wav',n\_bands, i),y,fs)

end

**Verification of filter used in tone-vocoder:**

%% 低通滤波器验证

clear; clc; close all;

[x, fs] = audioread ("C\_01\_02.wav");

but\_order = 4;

for ctof\_fq = [20 50 100 400]

[lpf\_b, lpf\_a] = butter(but\_order, ctof\_fq/(fs/2));

[h,f]=freqz(lpf\_b,lpf\_a,512,fs); % Digital filter frequency response

figure;

plot(f,20\*log10(abs(h))); % in dB scale

hold on;

plot(f, ones(1, length(f)).\*(-3));

axis([0 800 -50 5]);

xlabel('Frequency (Hz) ');

ylabel('Magnitude');

title(sprintf("lowpass filter of cutoff=%d", ctof\_fq))

end

%% 带通滤波器验证

clear; clc; close all;

[sig, fs] = audioread ("C\_01\_02.wav");

ctof\_fq = 50;

but\_order = 4;

fbe = 200;

fen = 7000;

for n\_bands = [4 6 8 16]

f2d = @(f) log10(f/165.4 + 1) / 0.06;

% 求耳蜗长度

d2f = @(d) 165.4 \* (10.^(0.06\*d) - 1);

% 求对应长度所对应的频率

dbe = f2d(fbe);

% 耳蜗长度最小值

den = f2d(fen);

% 耳蜗长度最大值

ds = linspace(dbe, den, n\_bands+1);

% 用n+1个数产生n个间隔，产生耳蜗长度的等间距分布

% cotf\_fq是产生包络的截止频率

t = (0:length(sig)-1)/fs;

for i = 1:n\_bands

d0 = ds(i); d1 = ds(i+1);

% 该滤波器所对应的耳蜗长度

f0 = d2f(d0); f1 = d2f(d1);

% 该耳蜗长度对应的频率分布

[bpf\_b, bpf\_a] = butter(but\_order, [f0 f1]/(fs/2));

[h,f]=freqz(bpf\_b,bpf\_a,512,fs); % Digital filter frequency response

thres = ones(1, length(f)).\*(-3);

db\_mag = 20\*log10(abs(h));

figure;

plot(f,db\_mag); % in dB scale

hold on;

plot(f, thres);

axis([0 8000 -50 5]);

xlabel('Frequency (Hz) ');

ylabel('Magnitude');

title(sprintf("fbe=%d fen=%d bandpass filter when band num is %d", f0, f1, n\_bands))

end

end