

Project 1 report

张骥霄 杨牧之 冯柏钧 周安然

12012924 12010422 12011124 12011214

May 20, 2022

1 Introduction

To Help deaf people to hear again, today, a prosthetic device, called the cochlear implant, can be implanted in the inner ear and can restore partial hearing to profoundly deaf people.

To verify the principle of the cochlear implant, we do this project - Speech synthesis and perception with envelope cue.

The main steps of our implementation is to filter the original signal with different bandpass filters, do full-wave rectification and pass a low-pass filter, multiple with a sine wave and finally sum up each components and get the result. Specified in the following figure.

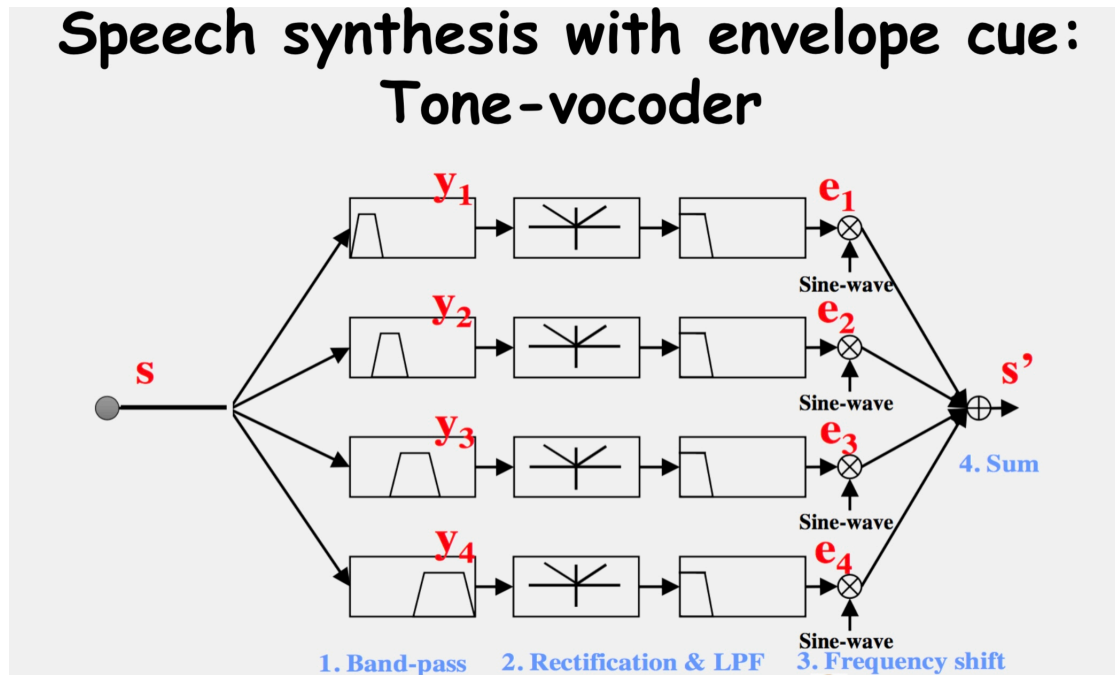


Figure 1: The process of our experiment

2 Task 1(张骥霄)

2.1 Task

Project tasks -1

- Sentences for pro 1: 'C_01_01.wav' & 'C_01_02.wav'
- Task 1
 - Set LPF cut-off frequency to 50 Hz.
 - Implement tone-vocoder by changing the number of bands to $N=1$, $N=2$, $N=4$, $N=6$, and $N=8$.
 - Save the wave files for these conditions, and describe how the number of bands affects the intelligibility (i.e., how many words can be understood) of synthesized sentence.

2.2 Result

As required, I first set the LPF cut-off frequency to 50Hz as shown below, which suit our needs.

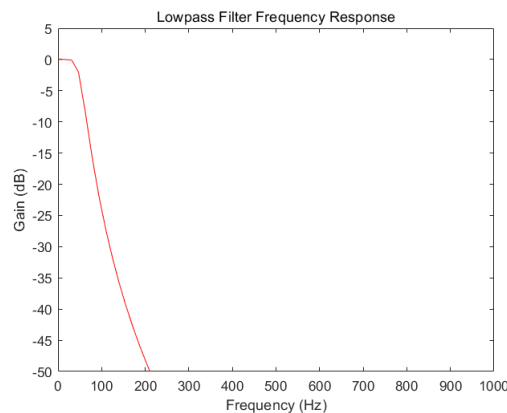


Figure 2: The certification of low-pass filer

Besides, I also check the bandpass filter to see if it is worked as expected, here is one example, which substantiate the correctness of the used filter.

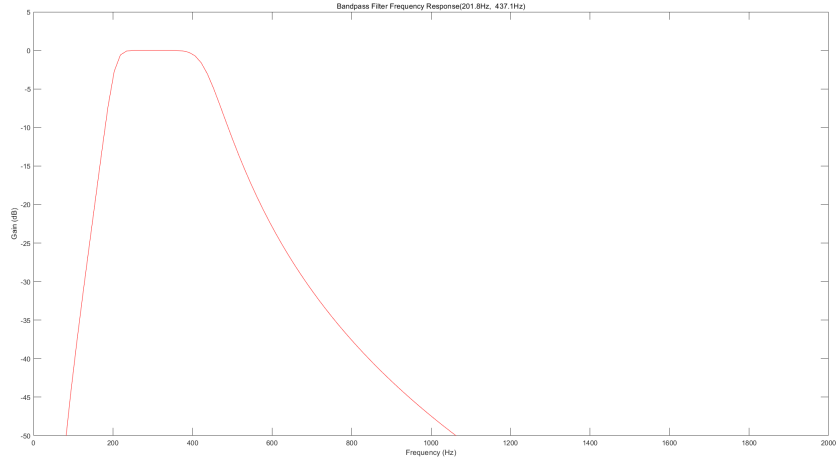


Figure 3: The certification of low-pass filer

I separately process the provided two signals, here is the original signal of the first signal.

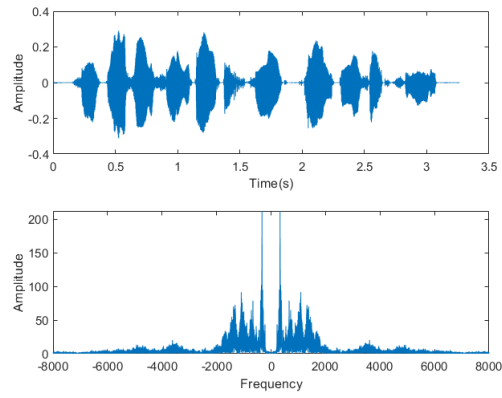


Figure 4: The original signal of the first signal

By set the number of bands $N = 1, 2, 4, 8, 16$, we get the following result, where the first 8 subplots are the waves of processed signals and next 8 subplots are their corresponding Fourier Transform.

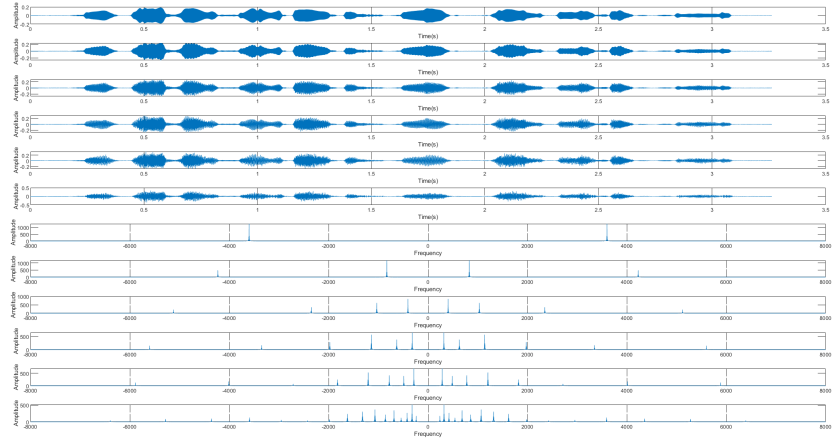


Figure 5: The processed signal of the first signal

Similarly, I processed the second signal using the same wave, here is the original signal and the processed signal of the second signal respectively.

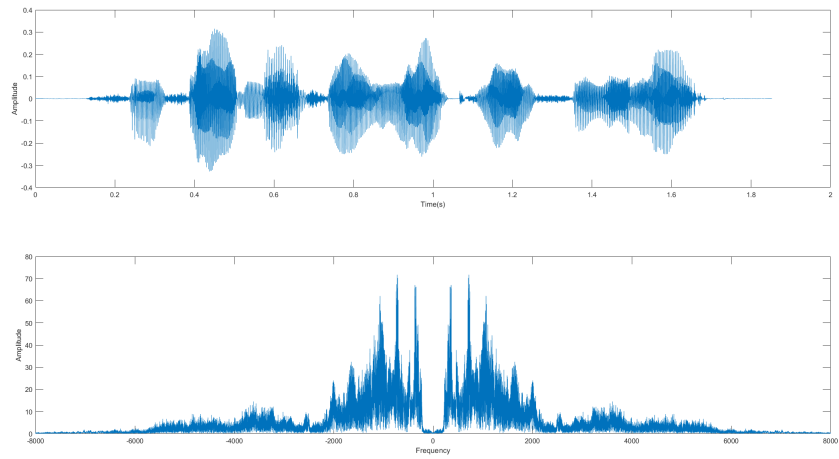


Figure 6: The original signal of the second signal

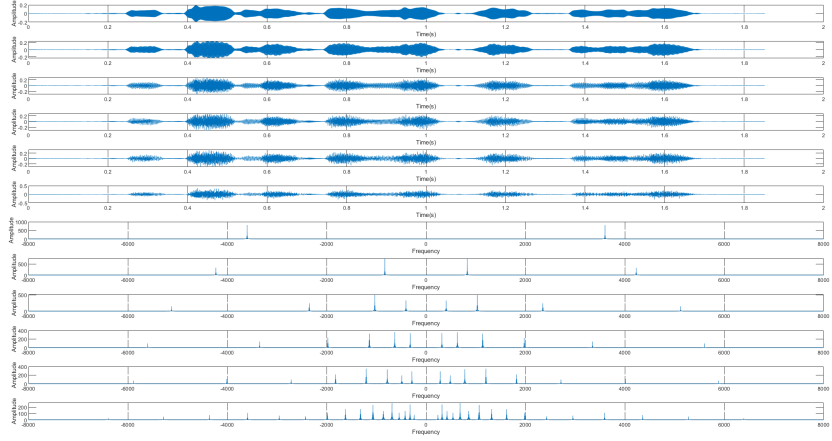


Figure 7: The processed signal of the second signal

Finally, I also draw the power spectrum of the two signal. First I will show the power spectrum of the original signal of first signal.

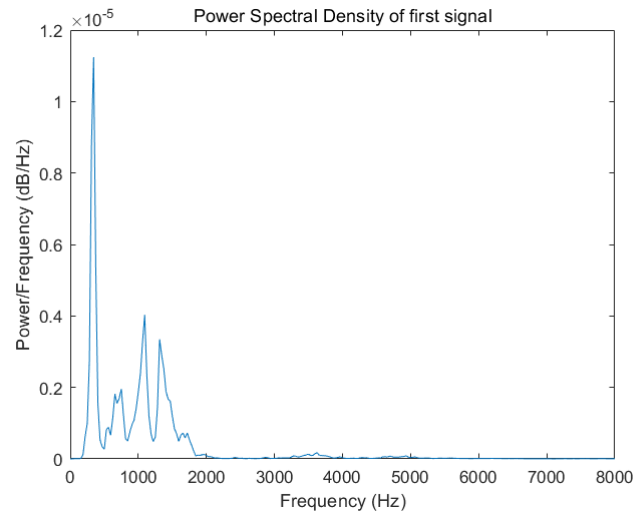


Figure 8: The power spectrum of the original signal of the first signal

Then I set the number of bands $N = 1, 2, 4, 8, 16$, the result are shown in the following picture respectively.

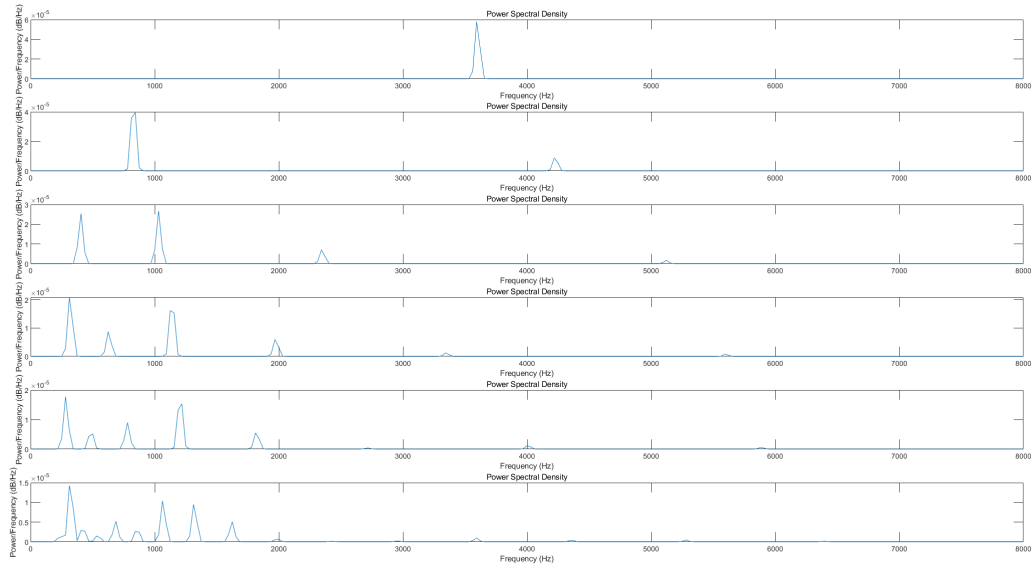


Figure 9: The power spectrum of the processed signal of the first signal

Similarly, the following two pictures are the result of the original and processed signal respectively.

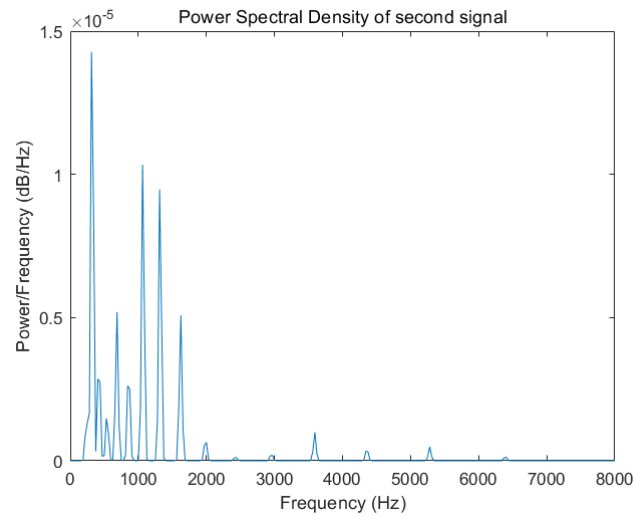


Figure 10: The power spectrum of the original signal of the second signal

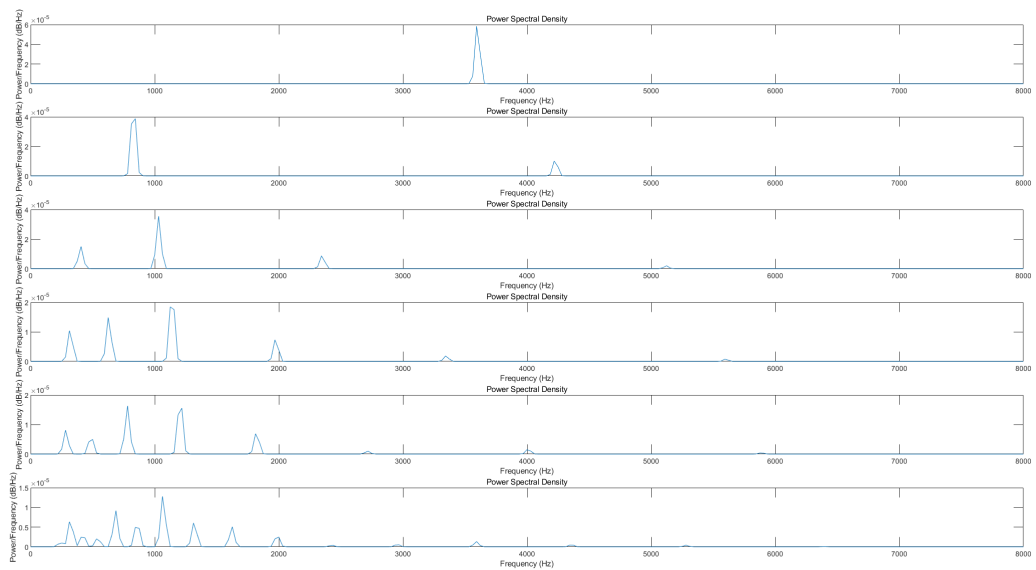


Figure 11: The power spectrum of the processed signal of the second signal

2.3 Analysis

As we can see, when the LPF cut-off frequency is fixed, the more bands the processed signal have, the more information the signal is held, which we can see from above figures, where the bands is more, the Fourier transform and the power spectrum is more close to the original signal's. The audio produced can also prove this point, when the bands is smaller than 4, we cannot understand any words in the audio, as the bands number is bigger than eight, we can roughly hear what words the audio contains, while more bands means clearer audio(When the band number is 64 or 128, it perform little difference compared to the original signal).

3 Task 2 (冯柏钧)

3.1 Task

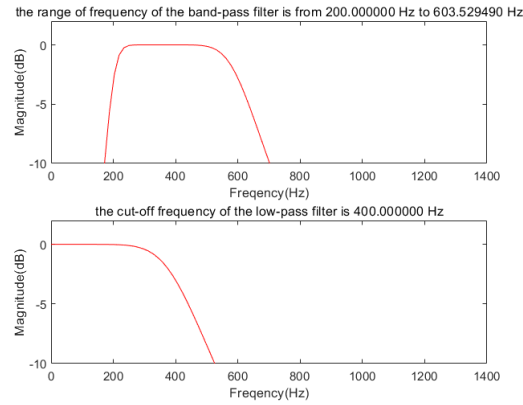
Project tasks -2

- **Task 2**
 - Set the number of bands $N=4$.
 - Implement tone-vocoder by changing the LPF cut-off frequency to 20 Hz, 50 Hz, 100 Hz, and 400 Hz.
 - Describe how the LPF cut-off frequency affects the intelligibility of synthesized sentence.

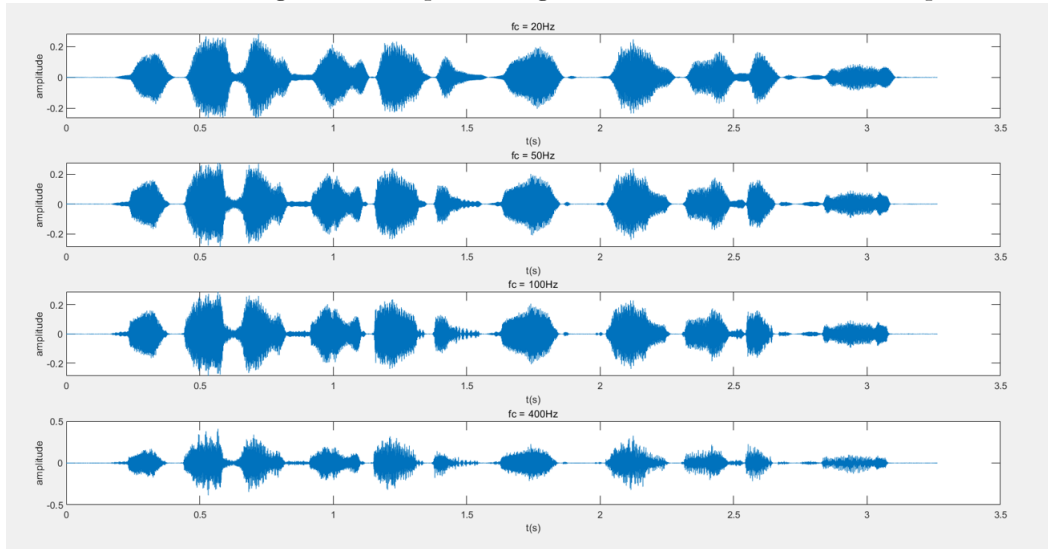
3.2 Result

Before we use the band-pass filter and the low-pass filter to get the envelopes, we must confirm the properties of the filter first. Since my task pays no attention to the exact order of the low-pass filter, I select the 4 as the number of order of all low-pass filters used in my task. In my task, the band number is always 4, so I plot the certification of one band-pass filter and one low-pass filter as an example. Some other configuration of the filters are listed in the figure.

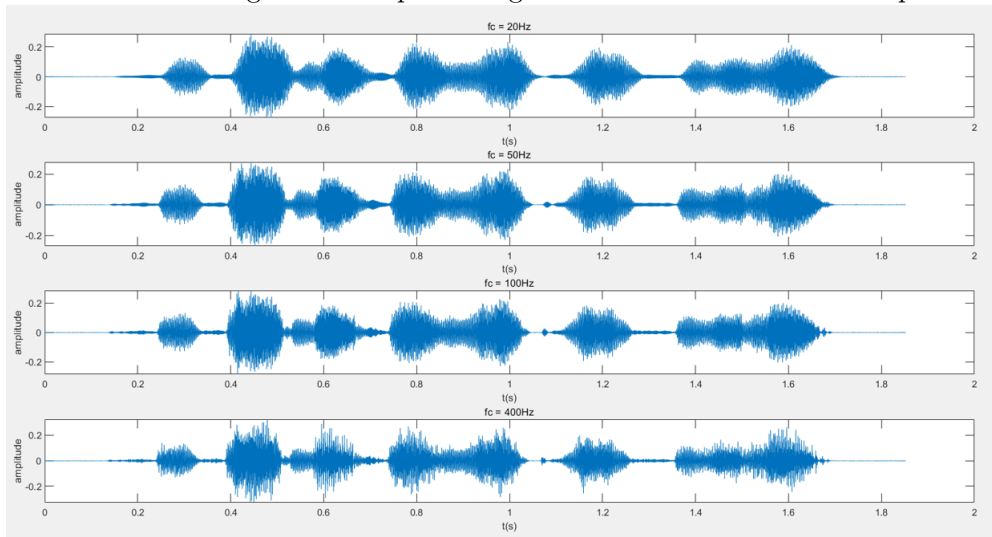
The certification of one band-pass filter and one low-pass filter



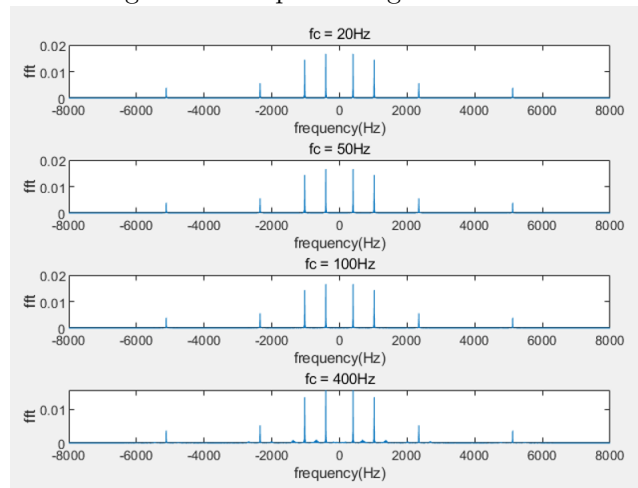
The waveform of signal 1 after processing with different LPF cut-off frequencies



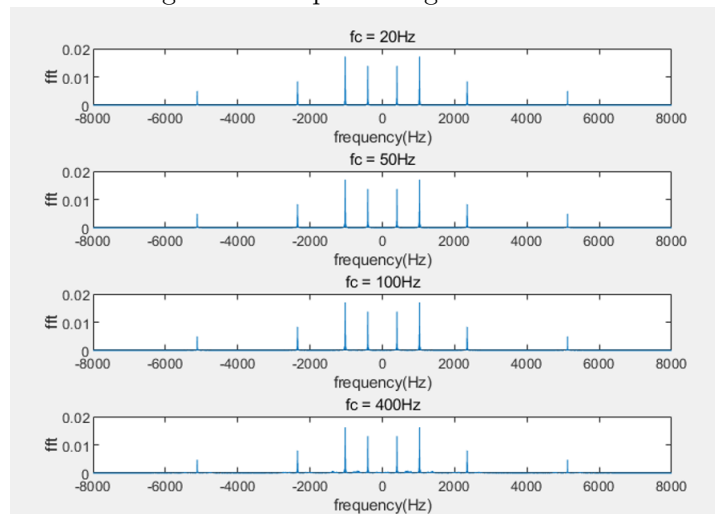
The waveform of signal 2 after processing with different LPF cut-off frequencies



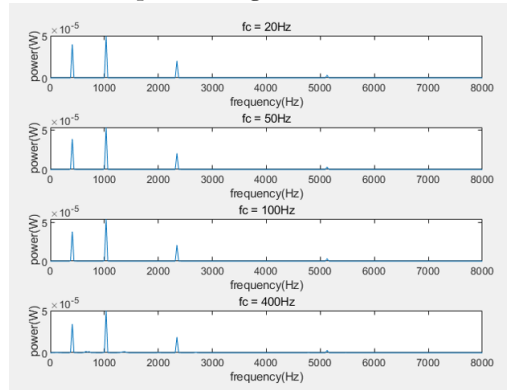
The Fourier transform of signal 1 after processing with different LPF cut-off frequencies



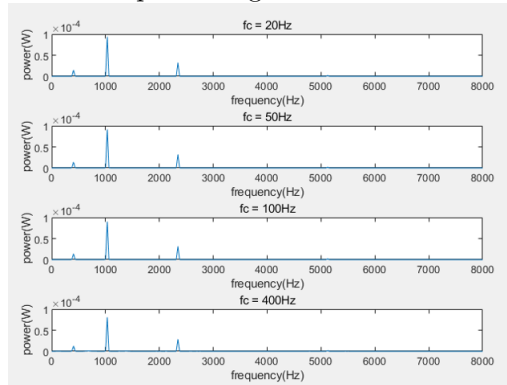
The Fourier transform of signal 2 after processing with different LPF cut-off frequencies



The power of signal 1 after processing with different LPF cut-off frequencies



The power of signal 1 after processing with different LPF cut-off frequencies



3.3 Analysis

Within the given range, as the LPF cut-off frequency increases, the voice of people becomes clearer.

I test the LPF cut-off frequency to be 4000Hz, it is clearer than the previous work, however, there still exists significant difference with Original file.

I design my program with the thinking of functional programming. Encapsulate my codes to make my work more clear and readable.

4 Task 3

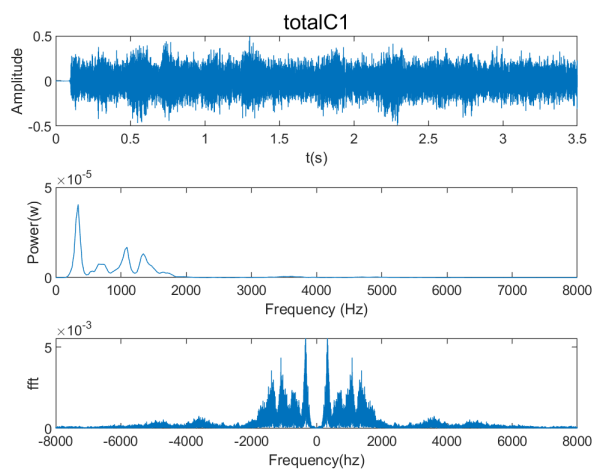
4.1 Task

Project tasks -3

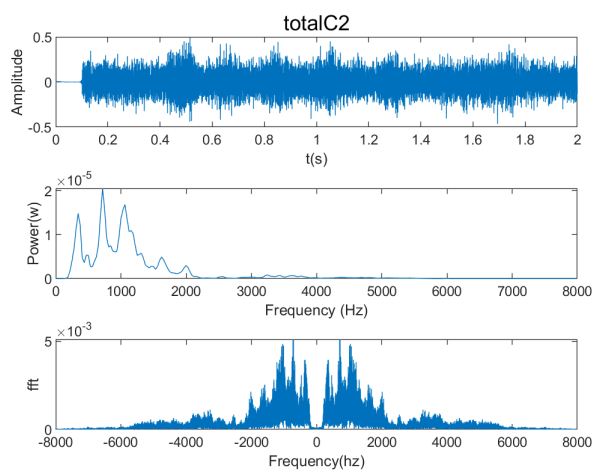
- **Task 3**
 - Generate a noisy signal (summing clean sentence and SSN) at SNR -5 dB.
 - Set LPF cut-off frequency to 50 Hz.
 - Implement tone-vocoder by changing the number of bands to N=2, N=4, N=6, N=8, and N=16.
 - Describe how the number of bands affects the intelligibility of synthesized sentence, and compare findings with those obtained in task 1.

4.2 Result

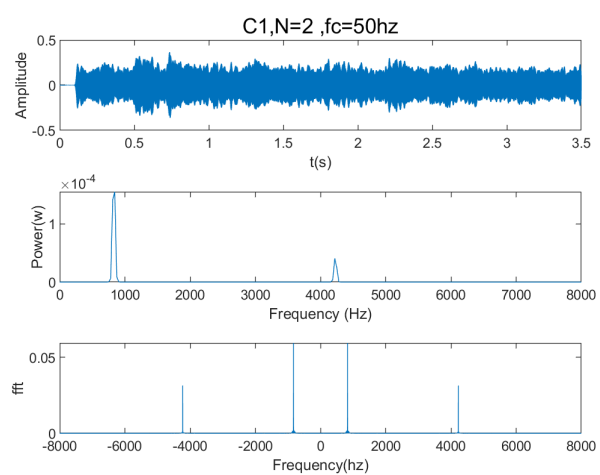
signal1 total

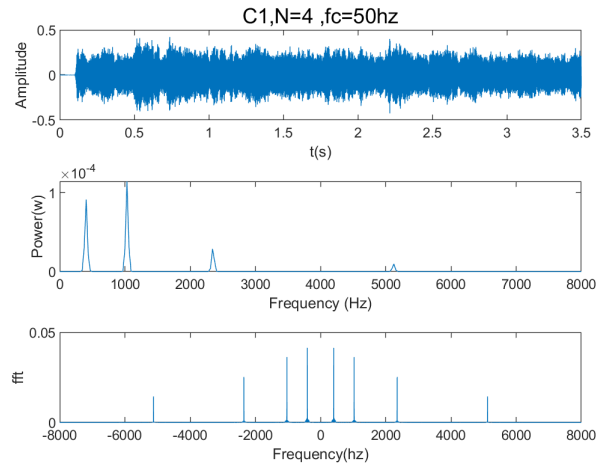


signal2 total

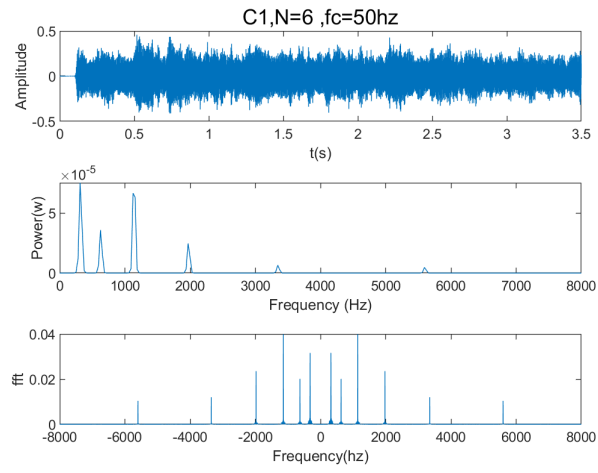


signal1 N=2

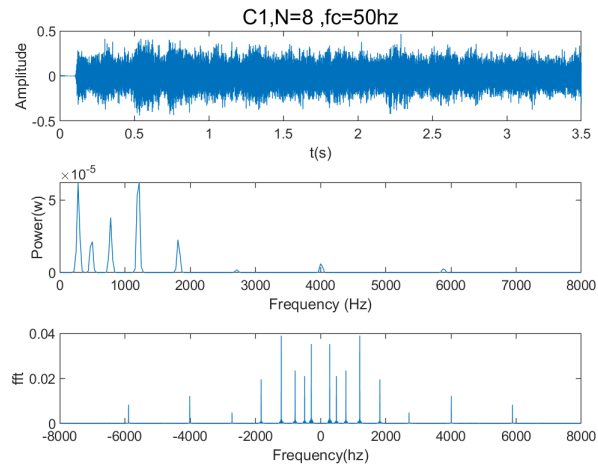




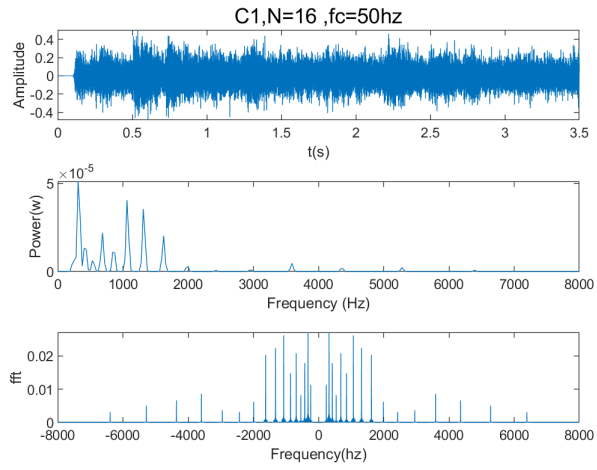
signal1 N=4



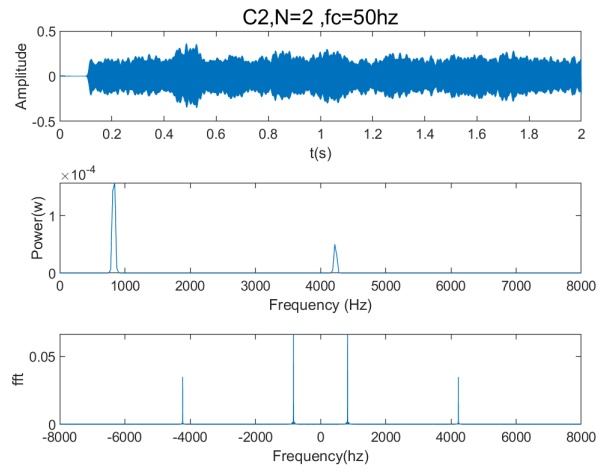
signal1 N=6



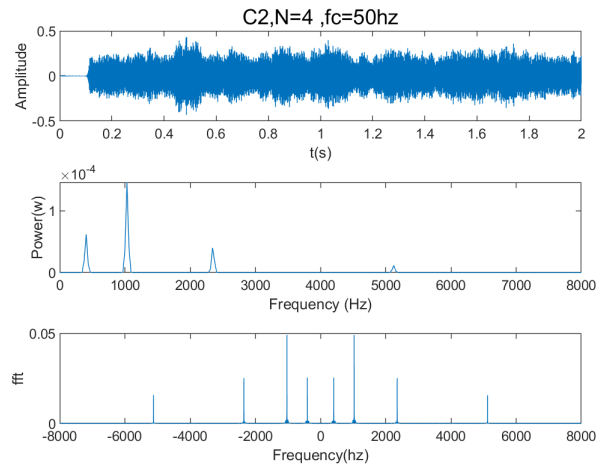
signal1 N=8



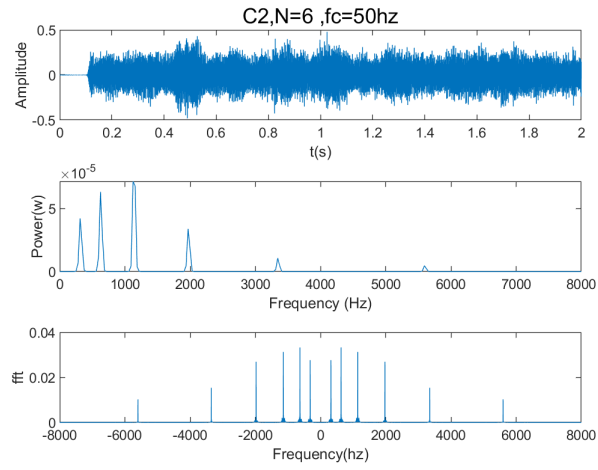
signal1 N=16



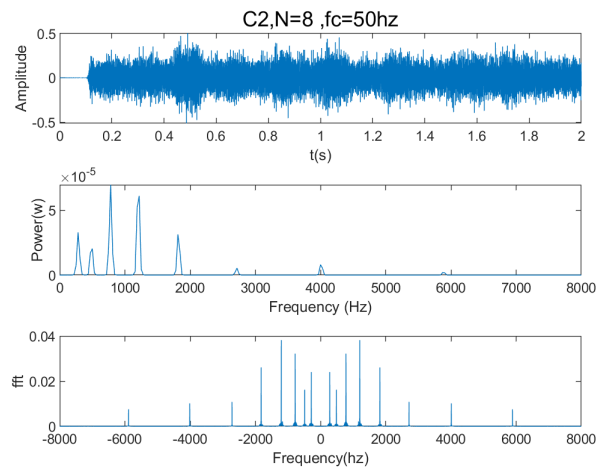
signal2 N=2



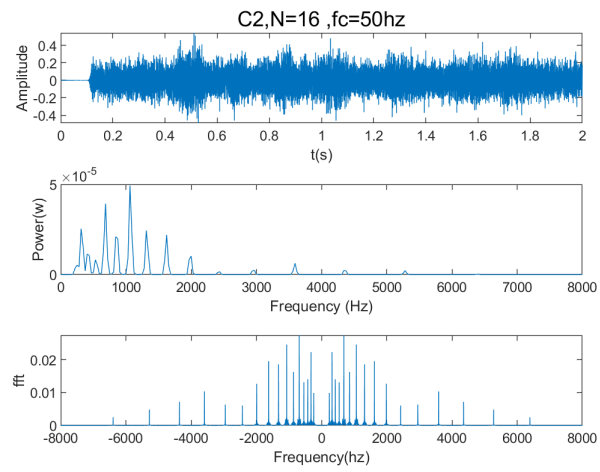
signal2 N=4



signal2 N=6

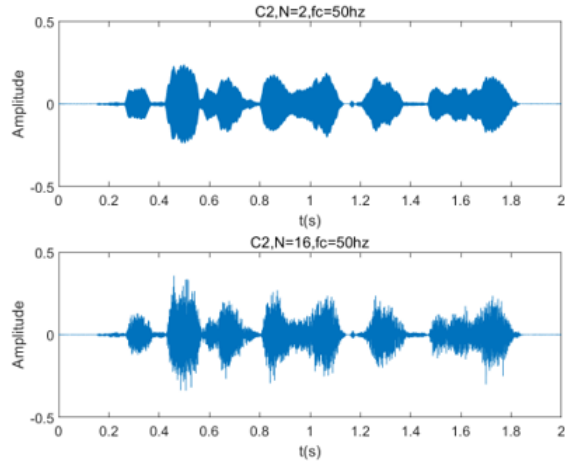


signal2 N=8

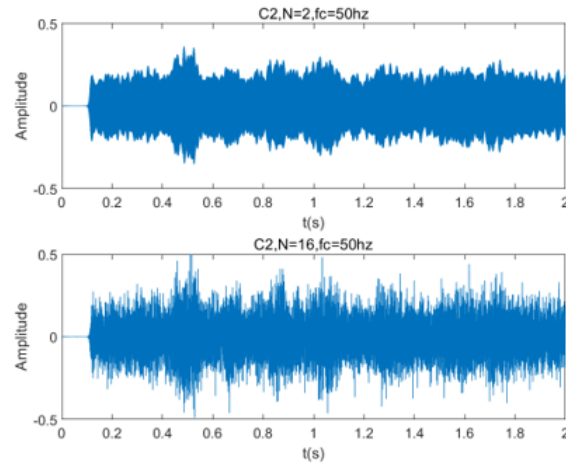


signal2 N=16

4.3 Analysis



Original signal

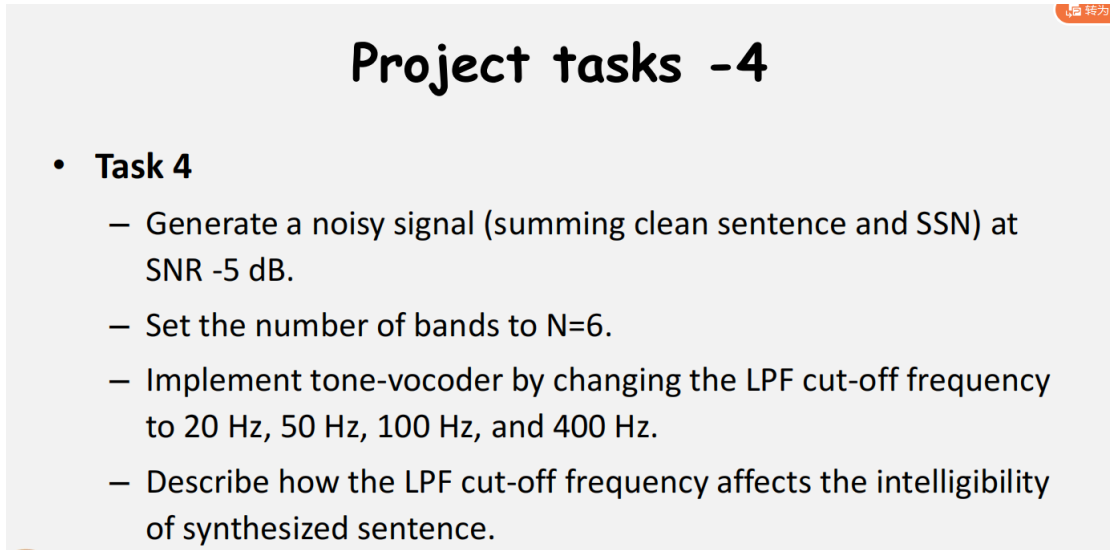


Synthetic signal

I compared N2 and N16 of the original signal with the signal with noise respectively. We can see from the figure that when $N=2$, the byte of the speech signal is not clearly distinguished from noise, and what we heard just now is almost noise. When $N=16$, there is a distinct distinction between speech signal and noise. Note: In a certain range, the larger N is, the smaller the masking effect of noise is.

5 Task 4(周安然)

5.1 Task

The image shows a presentation slide titled "Project tasks -4". It lists the requirements for Task 4: generating a noisy signal, setting the number of bands, implementing a tone-vocoder, and describing the effect of LPF cut-off frequency on intelligibility.

Project tasks -4

- **Task 4**
 - Generate a noisy signal (summing clean sentence and SSN) at SNR -5 dB.
 - Set the number of bands to N=6.
 - Implement tone-vocoder by changing the LPF cut-off frequency to 20 Hz, 50 Hz, 100 Hz, and 400 Hz.
 - Describe how the LPF cut-off frequency affects the intelligibility of synthesized sentence.

5.2 Result

1. Firstly, we make sure we have set SNR equals to -5.

noise2	1x29636 double
pulseSNR1	-5.1305
pulseSNR2	-5.0897

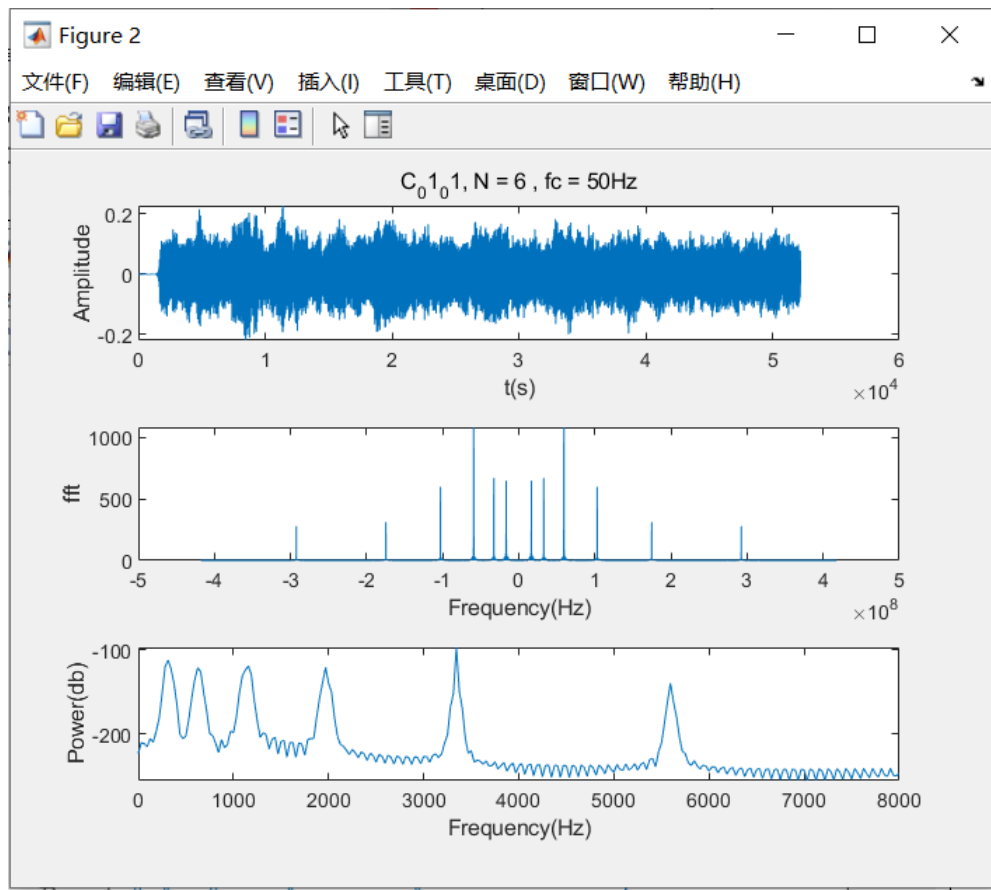
This is $C_01_01, SSN1$ and $y1 = C_01_01 + SSN1$.

This is $C_01_02, SSN2$ and $y2 = C_01_02 + SSN2$.

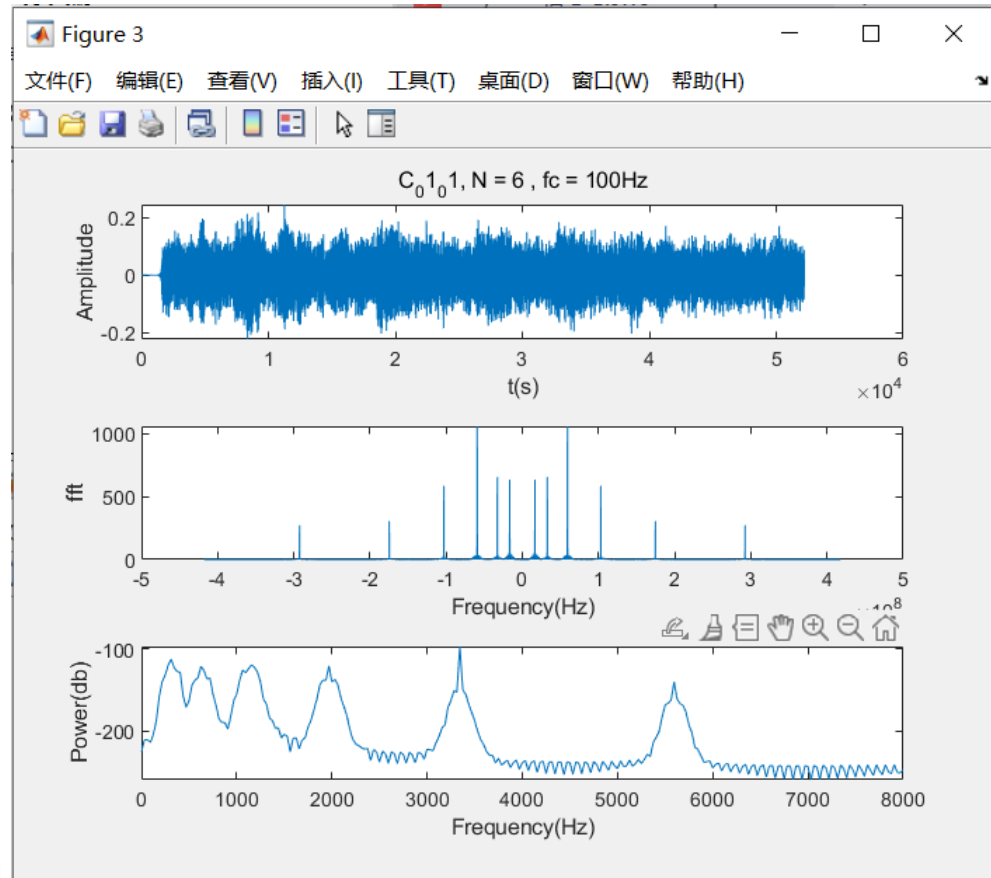
From the above pictures, we can see that We could find that the noise covers the most area of clean sentence at SNR = -5.

2. According to the request of task4, I set $N = 6$ and four different LPF cut-off frequency. Here are pictures which contains psd-Time, fft-Frequency and Power-Frequency of $y1$ and $y2$ with different LPF cutoff frequency.

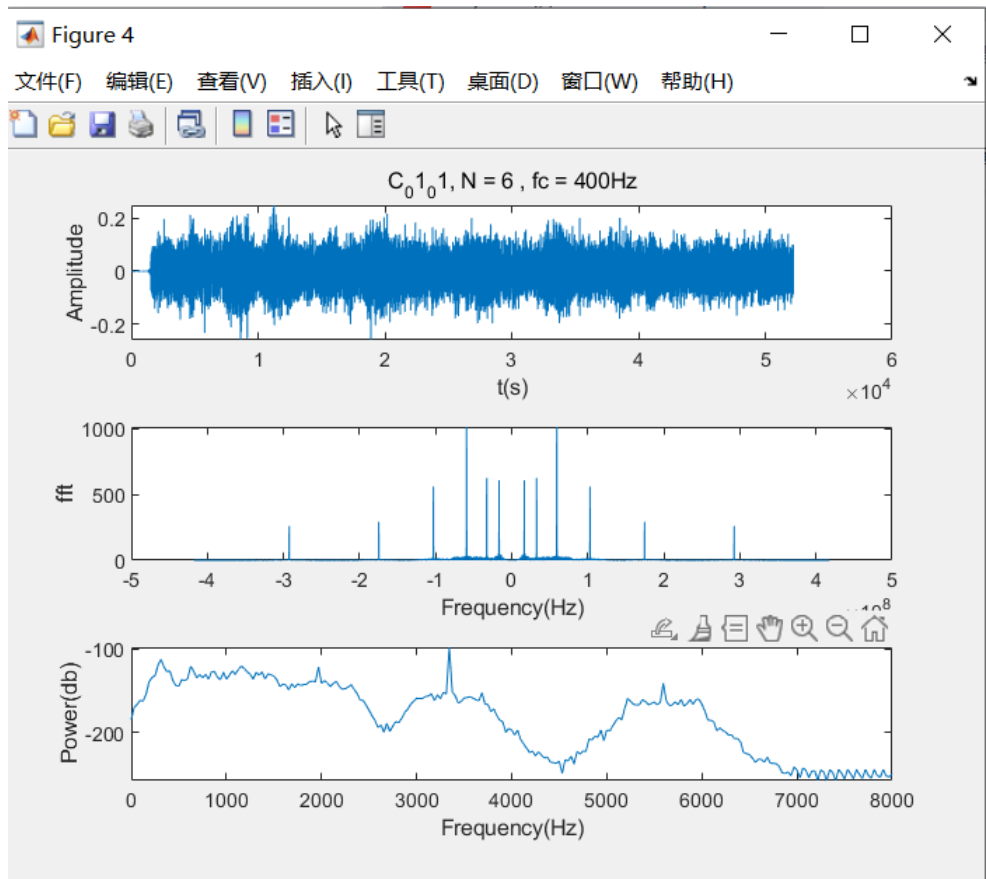
For $y1 = C_01_01 + SSN1$



Synthetic signal

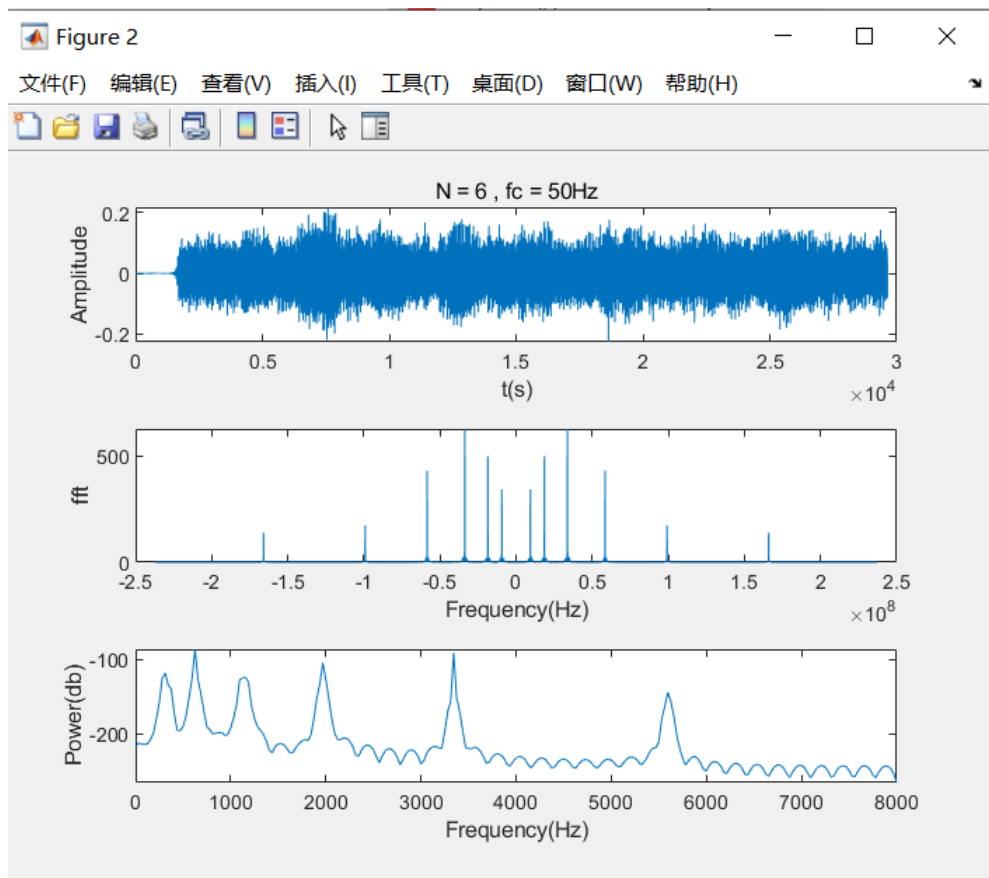


Synthetic signal

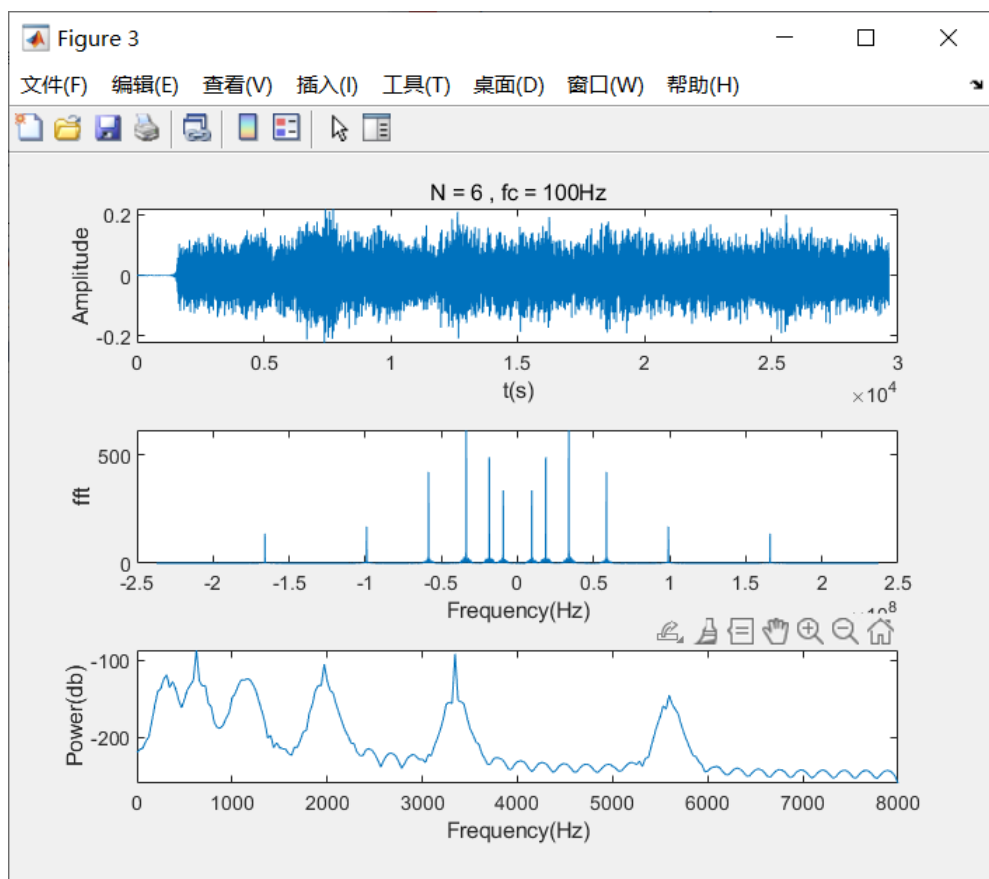


Synthetic signal

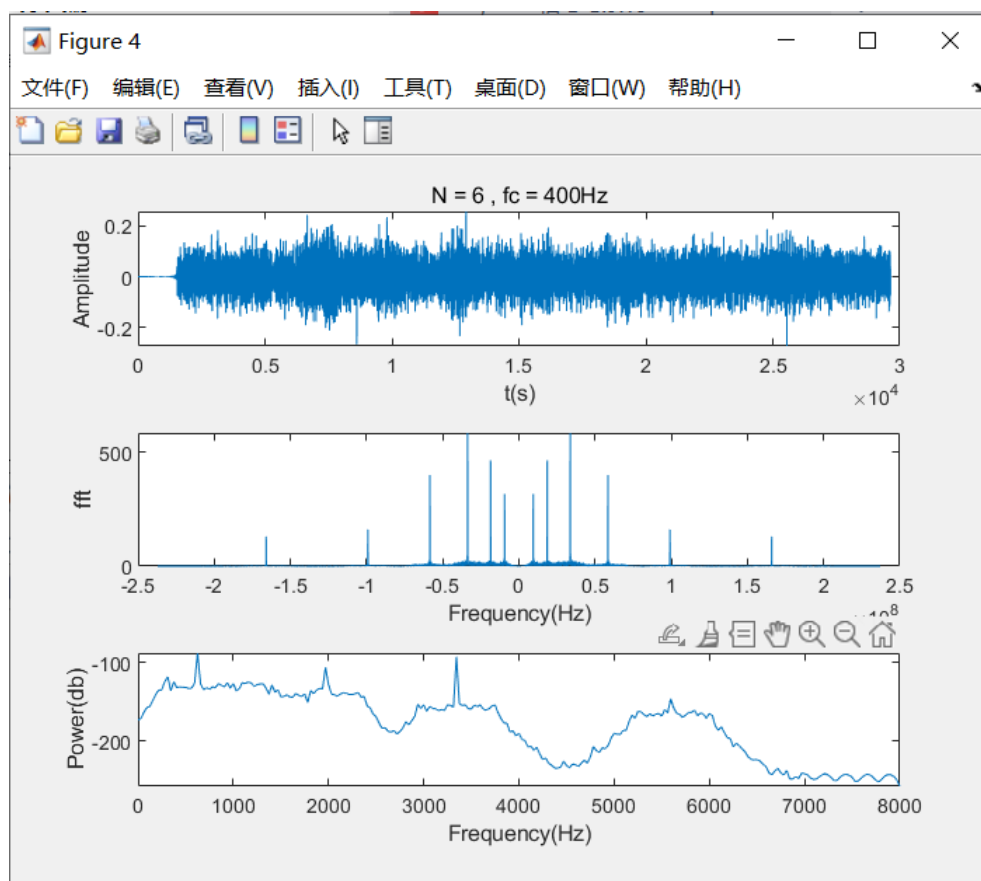
For $y_2 = C_01_02 + SSN_2$



Synthetic signal

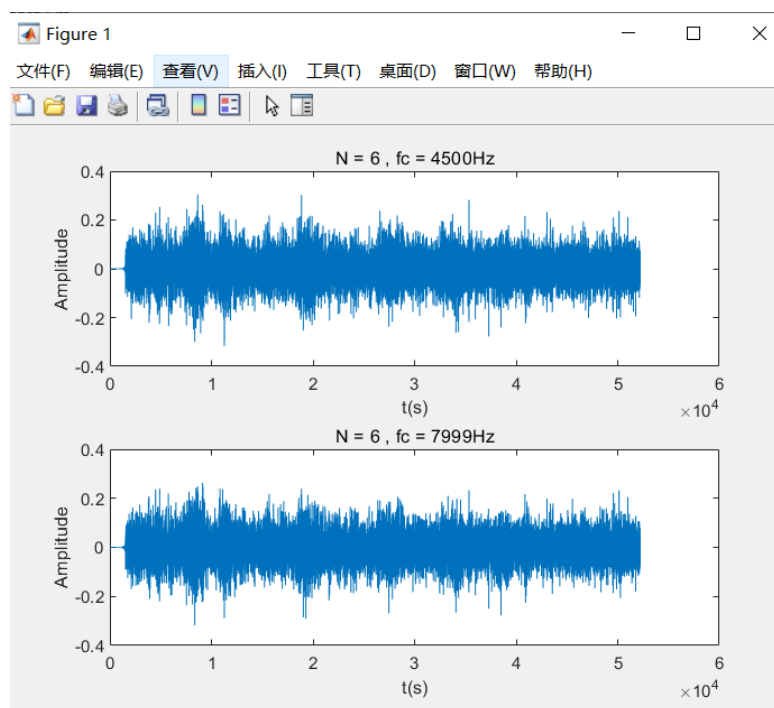


Synthetic signal

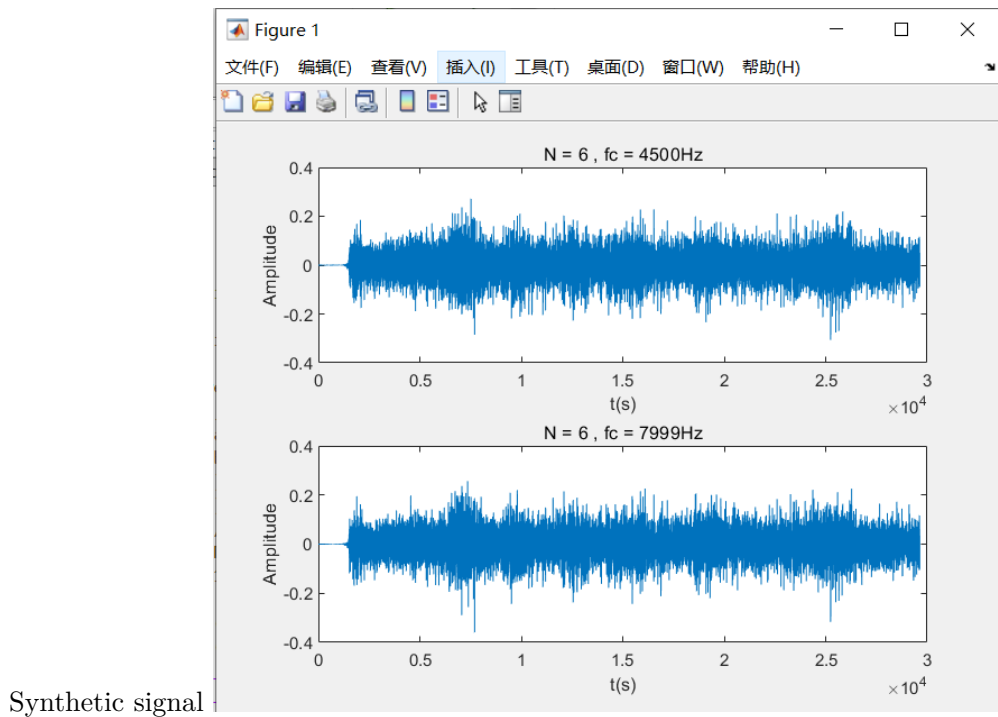


Synthetic signal

3. I change the LPF cut-off frequency to a higher level for each file in order to find how the LPF cut-off frequency affects the intelligibility of synthesized sentence.



Synthetic signal



5.3 Analysis

Compare the result pictures in 1, it is obvious that Speech-shaped-noise greatly interferes the intelligibility of synthesized sentence. In the case where $N(N = 6)$ is same, we can roughly distinguish speech when the LPF cut-off frequency covers from 100Hz to 400Hz. Therefore, we can analyze that as the LPF cut-off frequency increases in a certain range, the signal's identification will increase. I test the LPF cut-off frequency to be 4500Hz and 7999Hz, it is clearer than the previous work. However, because of interference of SSN, it is still very hard to hear the voice clear.

6 Experience

In this project, we applied what we have learned in this course comprehensively, which help us learn more about how to use the knowledge of signal and system to solve some real-world problems. In task1, I discover the effect of the bands not only by hear the generated audio, but also through the wave curve, Fourier Transform in frequency domain and the energy spectrum. My task4 is similar to task2. Both tasks are analyse how the LPF cut-off frequency affects the intelligibility of synthesized sentence. But task4 add SSN in the voice file. I use a function program of task2 to make my work more clear and readable.

7 Code

7.1 Task 1

```

1
2 [y1, fs1] = audioread('C_01_01.wav');
3 [b_low, a_low] = butter(4, 50/(fs1/2), 'low');
```

```

4
5 [h, f]= freqz(b_low,a_low,512,fs1);
6
7 nfft = 512;
8 Winodws = hann(nfft);
9 noverlap = nfft/2;
10
11 figure(5);
12 plot(f, 20 * log10(abs(h)), 'r' );
13 axis([0 1000 -50 5]);
14 title('Lowpass Filter Frequency Response');
15 xlabel('Frequency (Hz)');
16 ylabel('Gain (dB)');
17
18 [Pxx1, w1] = pwelch(y1, Winodws, noverlap, nfft, fs1);
19 figure(6)
20 plot(w1, Pxx1);
21 title('Power Spectral Density of first signal');
22 xlabel('Frequency (Hz)');
23 ylabel('Power/Frequency (dB/Hz)');
24
25
26 figure(1)
27 N = [1, 2, 4, 6, 8, 16];
28 for i = 1:length(N)
29     result_wave = zeros(length(y1),1);
30     n = N(i);
31     for j = 1:n
32         d_low = (27.2784-5.77322) / n * (j-1) + 5.77322;
33         d_high = (27.2784-5.77322) / n * (j) + 5.77322;
34         low_frequency = 165.4 * (10^(0.06*d_low) - 1);
35         high_frequency = 165.4 * (10^(0.06*d_high) - 1);
36         wave = get_band(y1, fs1, low_frequency, high_frequency, b_low,
37             a_low);
38         result_wave = result_wave + wave;
39     end
40     result_wave = result_wave * norm(y1)/ norm(result_wave);
41     fft_result_wave = fftshift(fft(result_wave));
42     t = linspace(0, length(result_wave)/fs1, length(result_wave));
43     line = linspace(0, fs1, length(fft_result_wave)) - fs1/2;
44     subplot(6,1,i);
45     [Pxx1, w1] = pwelch(result_wave, Winodws, noverlap, nfft, fs1);
46     plot(w1, Pxx1);

```

```

46     title('Power Spectral Density');
47     xlabel('Frequency (Hz)');
48     ylabel('Power/Frequency (dB/Hz)');
49
50 end
51
52 fft_y1 = fftshift(fft(y1));
53 t = linspace(0, length(y1)/fs1, length(y1));
54 line = linspace(0, fs1, length(fft_y1)) - fs1/2;
55 figure(2)
56 subplot(2,1,1);
57 plot(t, y1);
58 xlabel('Time(s)');
59 ylabel('Amplitude');
60 subplot(2,1,2);
61 plot(line, abs(fft_y1));
62 xlabel('Frequency');
63 ylabel('Amplitude');
64
65
66 [y2, fs2] = audioread('C_01_02.wav');
67
68 [b_low,a_low] = butter(4, 50/(fs2/2), 'low');
69
70 [Pxx2, w2] = pwelch(y2, Winodws, noverlap, nfft, fs2);
71 figure(7)
72 plot(w1, Pxx1);
73 title('Power Spectral Density of second signal');
74 xlabel('Frequency (Hz)');
75 ylabel('Power/Frequency (dB/Hz)');
76
77
78
79 figure(3)
80 N = [1, 2, 4, 6, 8, 16];
81 %N = [1];
82 for i = 1:length(N)
83     result_wave = zeros(length(y2),1);
84     n = N(i);
85     for j = 1:n
86         d_low = (27.2784-5.77322) / n * (j-1) + 5.77322;
87         d_high = (27.2784-5.77322) / n * (j) + 5.77322;
88         low_frequency = 165.4 * (10^(0.06*d_low) - 1)

```

```

89         high_frequency = 165.4 * (10^(0.06*d_high) - 1)
90         wave = get_band(y2, fs2, low_frequency, high_frequency, b_low,
91             a_low);
92         result_wave = result_wave + wave;
93     end
94     result_wave = result_wave * norm(y2) / norm(result_wave);
95     fft_result_wave = fftshift(fft(result_wave));
96     t = linspace(0, length(result_wave)/fs2, length(result_wave));
97     line = linspace(0, fs2, length(fft_result_wave)) - fs2/2;
98     subplot(6,1,i);
99     [Pxx2, w2] = pwelch(result_wave, Winodws, noverlap, nfft, fs2);
100    plot(w2, Pxx2);
101    title('Power Spectral Density');
102    xlabel('Frequency (Hz)');
103    ylabel('Power/Frequency (dB/Hz)');
104    audiowrite(['C_01_02_', num2str(n), '.wav'], result_wave, fs1);
105 end
106 %figure(4);
107 fft_y2 = fftshift(fft(y2));
108 t = linspace(0, length(y2)/fs2, length(y2));
109 line = linspace(0, fs2, length(fft_y2)) - fs2/2;
110
111
112
113 function wave = get_band(s, fs, low_frequency, high_frequency, b_low,
114     a_low)
115     len = length(s)/fs;
116     t = (0:1/fs:len - 1/fs);
117     [b, a] = butter(4, [low_frequency, high_frequency] / (fs / 2));
118     [h, f] = freqz(b, a, 512, fs);
119     %figure(6)
120     %plot(f, 20 * log10(abs(h)), 'r');
121     %axis([0 2000 -50 5]);
122     %title('Bandpass Filter Frequency Response(201.8Hz, 437.1Hz)');
123     %xlabel('Frequency (Hz)');
124     %ylabel('Gain (dB)');
125
126     y = filter(b, a, s);
127     y = abs(y);
128     mid_frequency = (low_frequency + high_frequency) / 2;
129     sin_wave = sin(2 * pi * mid_frequency * t);
130     sin_wave = sin_wave.';

```



```

130     envelope = filter(b_low, a_low, y);
131     wave = envelope .* sin_wave;
132     % wave = envelope;
133 end

```

7.2 Task 2

```

1  clc
2  clear
3  %the number of bands
4  N = 4;
5  %get the origin signal
6  [x1, fs1] = audioread('C_01_01.wav');
7  [x2, fs2] = audioread('C_01_02.wav');
8  %the lower bound of the frequency
9  fl = 200;
10 %the higher bound of the frequency
11 fh = 7000;
12 %enum cut-off frequencies of the LPF
13 fc1 = 20;
14 fc2 = 50;
15 fc3 = 100;
16 fc4 = 400;
17 x1 = x1';
18 x2 = x2';
19 %get the new signals
20 y11 = get_wave(x1, fs1, N, fl, fh, fc1);
21 y12 = get_wave(x1, fs1, N, fl, fh, fc2);
22 y13 = get_wave(x1, fs1, N, fl, fh, fc3);
23 y14 = get_wave(x1, fs1, N, fl, fh, fc4);
24
25 y21 = get_wave(x2, fs2, N, fl, fh, fc1);
26 y22 = get_wave(x2, fs2, N, fl, fh, fc2);
27 y23 = get_wave(x2, fs2, N, fl, fh, fc3);
28 y24 = get_wave(x2, fs2, N, fl, fh, fc4);
29
30 %time scale for the new signals
31 t1 = (0:1/fs1:length(x1)/fs1 - 1/fs1);
32 t2 = (0:1/fs2:length(x2)/fs2 - 1/fs2);
33
34 %plot new signals in time filed
35 figure(2)

```

```

36 subplot(411)
37 plot(t1,y11),title('fc = 20Hz'),xlabel('t(s)'),ylabel('amplitude')
38 subplot(412)
39 plot(t1,y12),title('fc = 50Hz'),xlabel('t(s)'),ylabel('amplitude')
40 subplot(413)
41 plot(t1,y13),title('fc = 100Hz'),xlabel('t(s)'),ylabel('amplitude')
42 subplot(414)
43 plot(t1,y14),title('fc = 400Hz'),xlabel('t(s)'),ylabel('amplitude')
44
45 figure(3)
46 subplot(411)
47 plot(t2,y21),title('fc = 20Hz'),xlabel('t(s)'),ylabel('amplitude')
48 subplot(412)
49 plot(t2,y22),title('fc = 50Hz'),xlabel('t(s)'),ylabel('amplitude')
50 subplot(413)
51 plot(t2,y23),title('fc = 100Hz'),xlabel('t(s)'),ylabel('amplitude')
52 subplot(414)
53 plot(t2,y24),title('fc = 400Hz'),xlabel('t(s)'),ylabel('amplitude')
54
55 %define the x-axis fot the fourier transform of the new signals
56 freq1 = linspace(-fs1/2,fs1/2-1,length(x1));
57 freq2 = linspace(-fs2/2,fs2/2-1,length(x2));
58
59 %plot the fourier transform
60 figure(4)
61 subplot(411)
62 plot(freq1,fftshift(abs(fft(y11./(length(y11)))))),title('fc = 20Hz'),
        xlabel('frequency(Hz)'),ylabel('fft')
63 subplot(412)
64 plot(freq1,fftshift(abs(fft(y12./(length(y12)))))),title('fc = 50Hz'),
        xlabel('frequency(Hz)'),ylabel('fft')
65 subplot(413)
66 plot(freq1,fftshift(abs(fft(y13./(length(y13)))))),title('fc = 100Hz'),
        xlabel('frequency(Hz)'),ylabel('fft')
67 subplot(414)
68 plot(freq1,fftshift(abs(fft(y14./(length(y14)))))),title('fc = 400Hz'),
        xlabel('frequency(Hz)'),ylabel('fft')
69
70 figure(5)
71 subplot(411)
72 plot(freq2,fftshift(abs(fft(y21./(length(y21)))))),title('fc = 20Hz'),
        xlabel('frequency(Hz)'),ylabel('fft')
73 subplot(412)

```

```

74 plot(freq2, fftshift(abs(fft(y22./length(y22))))), title('fc = 50Hz'),
    xlabel('frequency(Hz)'), ylabel('fft')
75 subplot(413)
76 plot(freq2, fftshift(abs(fft(y23./length(y23))))), title('fc = 100Hz'),
    xlabel('frequency(Hz)'), ylabel('fft')
77 subplot(414)
78 plot(freq2, fftshift(abs(fft(y24./length(y24))))), title('fc = 400Hz'),
    xlabel('frequency(Hz)'), ylabel('fft')
79
80 %calculate the power of the new signals
81 [p11,w11] = pwelch(y11,[],[],512,fs1);
82 [p12,w12] = pwelch(y12,[],[],512,fs1);
83 [p13,w13] = pwelch(y13,[],[],512,fs1);
84 [p14,w14] = pwelch(y14,[],[],512,fs1);
85 [p21,w21] = pwelch(y21,[],[],512,fs2);
86 [p22,w22] = pwelch(y22,[],[],512,fs2);
87 [p23,w23] = pwelch(y23,[],[],512,fs2);
88 [p24,w24] = pwelch(y24,[],[],512,fs2);
89
90 %plot new signals in power field
91 figure(6)
92 subplot(411)
93 plot(w11,p11), title('fc = 20Hz'), xlabel('frequency(Hz)'), ylabel('power(
    W)')
94 subplot(412)
95 plot(w12,p12), title('fc = 50Hz'), xlabel('frequency(Hz)'), ylabel('power(
    W)')
96 subplot(413)
97 plot(w13,p13), title('fc = 100Hz'), xlabel('frequency(Hz)'), ylabel('power
    (W)')
98 subplot(414)
99 plot(w14,p14), title('fc = 400Hz'), xlabel('frequency(Hz)'), ylabel('power
    (W)')
100
101 figure(7)
102 subplot(411)
103 plot(w21,p21), title('fc = 20Hz'), xlabel('frequency(Hz)'), ylabel('power(
    W)')
104 subplot(412)
105 plot(w22,p22), title('fc = 50Hz'), xlabel('frequency(Hz)'), ylabel('power(
    W)')
106 subplot(413)
107 plot(w23,p23), title('fc = 100Hz'), xlabel('frequency(Hz)'), ylabel('power

```

```

(W)')
108 subplot(414)
109 plot(w24,p24),title('fc = 400Hz'),xlabel('frequency(Hz)'),ylabel('power
(W)')
110
111 %record the new signals
112 audiowrite('P1_1_fc=_20.wav',y11,fs1);
113 audiowrite('P1_1_fc=_50.wav',y12,fs1);
114 audiowrite('P1_1_fc=_100.wav',y13,fs1);
115 audiowrite('P1_1_fc=_400.wav',y14,fs1);
116
117 audiowrite('P1_2_fc=_20.wav',y21,fs2);
118 audiowrite('P1_2_fc=_50.wav',y22,fs2);
119 audiowrite('P1_2_fc=_100.wav',y23,fs2);
120 audiowrite('P1_2_fc=_400.wav',y24,fs2);
121
122 %encapsulate the process with a function
123 function res = get_wave(in, fs, N, fl, fh, fc)
124     %predefine the size of the result
125     res = zeros(1,length(in));
126     %calculate the physical distance of human ear corresponding to given frequencies
127     dl = log10(fl/165.4 + 1)/0.06;
128     dh = log10(fh/165.4 + 1)/0.06;
129     d = (dh - dl)/N;
130     len = length(in)/fs;
131     %define the time scale for the sine signals
132     t = (0:1/fs:len - 1/fs);
133     for n = 1:N
134         %calculate the lower bound and the higher bound of the band-pass
135         %filter
136         l = 165.4*(10^(0.06*(dl + d*(n-1)))-1);
137         h = 165.4*(10^(0.06*(dl + d*n))-1);
138         [b, a] = butter(4, [l, h]/(fs/2));
139         wave = filter(b, a, in);
140         %full-wave rectification
141         wave = abs(wave);
142         [x, y] = butter(4, fc/(fs/2));
143         %plot the certification for the filter
144         if (n == 1)
145             [h1, f1] = freqz(b,a,512,fs);
146             [h2, f2] = freqz(x,y,512,fs);
147             figure(1)
148             subplot(211)

```

```

149     plot(f1,20*log10(abs(h1)), 'r'), axis([0 1400 -10 2]), title(sprintf
    ("the range of frequency of the band-pass filter is from %f Hz
    to xlabel('Frequency(Hz)'),ylabel('Magnitude(dB)')
150 subplot(212)
151 plot(f2,20*log10(abs(h2)), 'r'), axis([0 1400 -10 2]), title(sprintf
    ("the cut-off frequency of the low-pass filter is %f
    Hz",fc));
152 xlabel('Frequency(Hz)'), ylabel('Magnitude(dB)')
153 end
154 env = filter(x, y, wave);
155 sin_sig = sin(2 * pi * (1 + (h - 1)/2)* t);
156 %sum up
157 res = sin_sig.* env + res;
158 end
159 %normalization
160 res = res*norm(in)/norm(res);
161 end
162
163
164
165 %Q2

```

7.3 Task 3

```

1 clear;
2 clc;
3 [y1, fs1] = audioread('C_01_01.wav');
4 [y2, fs2] = audioread('C_01_02.wav');
5 sig1 = repmat(y1, 1, 10);
6 sig2 = repmat(y2, 1, 10);
7 nfft = 512;
8 Winodws = hann(nfft);
9 noverlap = nfft/2;
10 [Pxx1, w1] = pwelch(sig1, Winodws, noverlap, nfft, fs1);
11 [Pxx2, w2] = pwelch(sig2, Winodws, noverlap, nfft, fs2);
12 b1 = fir2(3000, w1/(fs1/2), sqrt(Pxx1/max(Pxx1)));
13 b2 = fir2(3000, w2/(fs2/2), sqrt(Pxx2/max(Pxx2)));
14 [h1, wh1] = freqz(b1, 1, 128);
15 [h2, wh2] = freqz(b2, 1, 128);
16 noise_1 = 1-2*rand(1,52215);
17 noise_2 = 1-2*rand(1,29636);
18 ssn1 = filter(b1, 1, noise_1)./1.065;
19 ssn2 = filter(b2, 1, noise_2)./1.53;

```

```

20 SNR1=20*log10(norm(y1)/norm(ssn1'))
21 SNR2=20*log10(norm(y2)/norm(ssn2'))
22 k1=y1+ssn1';
23 k2=y2+ssn2';
24 [Pyy1,w1] =pwelch(ssn1',Winodws, noverlap, nfft, fs2);
25 [Pyy2,w2] =pwelch(k2,Winodws, noverlap, nfft, fs2);
26 [b_low,a_low] = butter(4, 50/(fs1/2), 'low');
27 N = [1, 2, 4, 6, 8, 16, 32, 64, 128, 256];
28 for i = 1:length(N)
29     result_wave1 = zeros(length(y1),1);
30     result_wave2 = zeros(length(y1),1);
31     n = N(i);
32     for j = 1:n
33         d_low = (27.2784-5.77322) / n * (j-1) + 5.77322;
34         d_high = (27.2784-5.77322) / n * (j) + 5.77322;
35         low_frequency = 165.4 * (10^(0.06*d_low) - 1);
36         high_frequency = 165.4 * (10^(0.06*d_high) - 1);
37         wave1 = get_band(ssn1', fs1, low_frequency, high_frequency,
38             b_low, a_low);
39         wave2 = get_band(k1, fs1, low_frequency, high_frequency, b_low,
40             a_low);
41         result_wave1 = result_wave1 + wave1;
42         result_wave2 = result_wave2 + wave2;
43     end
44     result_wave1 = result_wave1 * norm(ssn1')/ norm(result_wave1);
45     result_wave2 = result_wave2 * norm(k1)/ norm(result_wave2);
46     audiowrite(['(n)C_01_01_', num2str(n), '.wav'], result_wave1, fs1);
47     audiowrite(['(k)C_01_01_', num2str(n), '.wav'], result_wave2, fs1);
48 end
49 [b_low,a_low] = butter(4, 50/(fs2/2), 'low');
50 N = [1, 2, 4, 6, 8, 16, 32, 64, 128, 256];
51 for i = 1:length(N)
52     result_wave1 = zeros(length(y2),1);
53     result_wave2 = zeros(length(y2),1);
54     n = N(i);
55     for j = 1:n
56         d_low = (27.2784-5.77322) / n * (j-1) + 5.77322;
57         d_high = (27.2784-5.77322) / n * (j) + 5.77322;
58         low_frequency = 165.4 * (10^(0.06*d_low) - 1);
59         high_frequency = 165.4 * (10^(0.06*d_high) - 1);
60         wave1 = get_band(ssn2', fs2, low_frequency, high_frequency,
61             b_low, a_low);
62         wave2 = get_band(k2, fs2, low_frequency, high_frequency, b_low,

```

```

        a_low);
60     result_wave1 = result_wave1 + wave1;
61     result_wave2 = result_wave2 + wave2;
62     end
63     result_wave1 = result_wave1 * norm(ssn2') / norm(result_wave1);
64     result_wave2 = result_wave2 * norm(k2) / norm(result_wave2);
65     audiowrite(['(n)C_01_02_', num2str(n), '.wav'], result_wave1, fs2);
66     audiowrite(['(k)C_01_02_', num2str(n), '.wav'], result_wave2, fs2);
67 end
68 audiowrite(['C_01_01_ssn1.wav'], ssn1, fs1);
69 audiowrite(['C_01_02_ssn2.wav'], ssn2, fs2);
70 audiowrite(['C_01_01_total1.wav'], k1, fs1);
71 audiowrite(['C_01_02_total2.wav'], k2, fs2);
72 [yx2, fsx2] = audioread('(k)C_01_02_2.wav');
73 player1 = audioplayer(yx2, fsx2);
74 play(player1)
75 [Py2, w2] = pwelch(yx2, Winodws, noverlap, nfft, fsx2);
76 subplot(3,1,1)
77 t2=linspace(0,2,29636);
78 plot(t2, yx2);
79 xlabel('t(s)')
80 ylabel('Amplitude');
81 title("C2,N=2 ,fc=50hz",'fontsize', 13)
82 subplot(3,1,2)
83 plot(w2, Py2);
84 xlabel('Frequency (Hz)')
85 ylabel('Power(w)');
86 subplot(3,1,3)
87 xx2=fft(yx2) ./ length(yx2);
88 t2=linspace(-8000,8000,29636);
89 plot(t2, abs(fftshift(xx2)));
90 xlabel('Frequency(hz)')
91 ylabel('fft');
92 [yx1, fsx1] = audioread('(k)C_01_01_2.wav');
93 [Py1, w1] = pwelch(yx1, Winodws, noverlap, nfft, fsx1);
94 subplot(3,1,1)
95 t1=linspace(0,3.5,52215);
96 plot(t1, yx1);
97 xlabel('t(s)')
98 ylabel('Amplitude');
99 title("C1,N=2 ,fc=50hz",'fontsize', 13)
100 subplot(3,1,2)
101 plot(w1, Py1);

```

```

102 xlabel('Frequency (Hz)')
103 ylabel('Power(w)');
104 subplot(3,1,3)
105 xx1=fft(yx1)./length(yx1);
106 t1=linspace(-8000,8000,52215);
107 plot(t1,abs(fftshift(xx1)));
108 xlabel('Frequency(hz)')
109 ylabel('fft');
110 [Pyy1,w1]=pwelch(k1,Winodws,noverlap,nfft,fsx1);
111 [Pyy2,w2]=pwelch(ssn2',Winodws,noverlap,nfft,fsx2);
112 subplot(3,1,1)
113 t1=linspace(0,3.5,52215);
114 plot(t1,k1);
115 xlabel('t(s)')
116 ylabel('Amplitude');
117 title("totalC1",'fontsize',13)
118 subplot(3,1,2)
119 plot(w1,Pyy1);
120 xlabel('Frequency (Hz)')
121 ylabel('Power(w)');
122 subplot(3,1,3)
123 xx1=fft(k1)./length(k1);
124 t1=linspace(-8000,8000,52215);
125 plot(t1,abs(fftshift(xx1)));
126 xlabel('Frequency(hz)')
127 ylabel('fft');
128 [yx1,fsx1]=audioread('C_01_02_2.wav');
129 [yx2,fsx2]=audioread('C_01_02_16.wav');
130 subplot(2,1,1)
131 t2=linspace(0,2,29636);
132 plot(t2,yx1);
133 axis([0,2,-0.5,0.5]);
134 xlabel('t(s)')
135 ylabel('Amplitude');
136 title("C2,N=2,fc=50hz")
137 subplot(2,1,2)
138 t1=linspace(0,2,29636);
139 plot(t2,yx2);
140 axis([0,2,-0.5,0.5]);
141 xlabel('t(s)')
142 ylabel('Amplitude');
143 title("C2,N=16,fc=50hz")

```


7.4 Task 4

```
1
2 clc;
3 clear;
4
5 %Set the number of bands to N = 6
6 N = 6;
7 %Changing LPF Cutoff
8 fl = 200;
9 fh = 7000;
10 fc1 = 20;
11 fc2 = 50;
12 fc3 = 100;
13 fc4 = 400;
14
15 %Generate a noisy signal at SNR -5dB
16
17 [s1,fs1] = audioread('C_01_01.wav');
18 sig1 = s1';
19 [pxx1,w1] = pwelch(sig1,[],[],512,fs1);
20 b1 = fir2(3000,w1/(fs1/2),sqrt(pxx1/max(pxx1)));
21 noise1 = 1-2*rand(1,length(s1));
22 SSN1 = filter(b1,1,noise1)./0.65;
23 E1 = norm(sig1);
24 E2 = norm(SSN1);
25 SSN1 = sqrt(E1/E2)*SSN1;
26 y1 = sig1 + SSN1;
27 y1 = y1/norm(y1)*norm(sig1);
28 %sound(y1,fs1);
29 y11 = get_wave(y1, fs1, 6, fl, fh, fc1);
30 y12 = get_wave(y1, fs1, 6, fl, fh, fc2);
31 y13 = get_wave(y1, fs1, 6, fl, fh, fc3);
32 y14 = get_wave(y1, fs1, 6, fl, fh, fc4);
33
34 audiowrite('P1_1_fc=_20.wav',y11,fs1);
35 audiowrite('P1_1_fc=_50.wav',y12,fs1);
36 audiowrite('P1_1_fc=_100.wav',y13,fs1);
37 audiowrite('P1_1_fc=_400.wav',y14,fs1);
38
39 figure(1)
40 subplot(3,1,1)
41 plot(1:length(y1),y11)
```

```

42 title('C_01_01, N = 6 , fc = 20Hz')
43 xlabel('t(s)'); ylabel('Amplitude')
44 subplot(3,1,2)
45 f = fs1*(-length(y1)/2:length(y1)/2-1);
46 plot(f, fftshift(abs(fft(y1))))
47 xlabel('Frequency(Hz)'); ylabel('fft')
48 subplot(3,1,3)
49 [pxx1,w1] = pwelch(y11,[],[],512,fs1);
50 plot(w1,20*log10(pxx1))
51 xlabel('Frequency(Hz)'); ylabel('Power(db)')
52
53 figure(2)
54 subplot(3,1,1)
55 plot(1:length(y1),y12)
56 title('C_01_01, N = 6 , fc = 50Hz')
57 xlabel('t(s)'); ylabel('Amplitude')
58 subplot(3,1,2)
59 f = fs1*(-length(y1)/2:length(y1)/2-1);
60 plot(f, fftshift(abs(fft(y12))))
61 xlabel('Frequency(Hz)'); ylabel('fft')
62 subplot(3,1,3)
63 [pxx1,w1] = pwelch(y12,[],[],512,fs1);
64 plot(w1,20*log10(pxx1))
65 xlabel('Frequency(Hz)'); ylabel('Power(db)')
66
67
68 figure(3)
69 subplot(3,1,1)
70 plot(1:length(y1),y13)
71 title('C_01_01, N = 6 , fc = 100Hz')
72 xlabel('t(s)'); ylabel('Amplitude')
73 subplot(3,1,2)
74 f = fs1*(-length(y1)/2:length(y1)/2-1);
75 plot(f, fftshift(abs(fft(y13))))
76 xlabel('Frequency(Hz)'); ylabel('fft')
77 subplot(3,1,3)
78 [pxx1,w1] = pwelch(y13,[],[],512,fs1);
79 plot(w1,20*log10(pxx1))
80 xlabel('Frequency(Hz)'); ylabel('Power(db)')
81
82 figure(4)
83 subplot(3,1,1)
84 plot(1:length(y1),y14)

```

```

85 title('C_01_01, N = 6 , fc = 400Hz')
86 xlabel('t(s)');ylabel('Amplitude')
87 subplot(3,1,2)
88 f = fs1*(-length(y1)/2:length(y1)/2-1);
89 plot(f,fftshift(abs(fft(y1))))
90 xlabel('Frequency(Hz)');ylabel('fft')
91 subplot(3,1,3)
92 [pxx1,w1] = pwelch(y1,[],[],512,fs1);
93 plot(w1,20*log10(pxx1))
94 xlabel('Frequency(Hz)');ylabel('Power(db)')
95
96
97 clc;
98 clear;
99
100 %Set the number of bands to N = 6
101 N = 6;
102 %Changing LPF Cutoff
103 fl = 200;
104 fh = 7000;
105 fc1 = 20;
106 fc2 = 50;
107 fc3 = 100;
108 fc4 = 400;
109 %Generate a noisy signal at SNR -5dB
110 [s2,fs2] = audioread('C_01_02.wav');
111 sig2 = s2';
112 [pxx2,w2] = pwelch(sig2,[],[],512,fs2);
113 b2 = fir2(3000,w2/(fs2/2),sqrt(pxx2/max(pxx2)));
114 noise2 = 1-2*rand(1,length(s2));
115 SSN2 = filter(b2,1,noise2)./0.7;
116 E3 = norm(sig2);
117 E4 = norm(SSN2);
118 SSN2 = sqrt(E3/E4)*SSN2;
119 y2 = sig2 + SSN2;
120 y2 = y2/norm(y2)*norm(sig2);
121 %sound(y2,fs2)
122 y21 = get_wave(y2, fs2, 6, fl, fh, fc1);
123 y22 = get_wave(y2, fs2, 6, fl, fh, fc2);
124 y23 = get_wave(y2, fs2, 6, fl, fh, fc3);
125 y24 = get_wave(y2, fs2, 6, fl, fh, fc4);
126
127 audiowrite('P1_2_fc=_20.wav',y21,fs2);

```

```

128 audiowrite('P1_2_fc=_50.wav',y22,fs2);
129 audiowrite('P1_2_fc=_100.wav',y23,fs2);
130 audiowrite('P1_2_fc=_400.wav',y24,fs2);
131
132 figure(1)
133 subplot(3,1,1)
134 plot(1:length(y2),y21)
135 title('N = 6 , fc = 20Hz')
136 xlabel('t(s)');ylabel('Amplitude')
137 subplot(3,1,2)
138 f = fs2*(-length(y2)/2:length(y2)/2-1);
139 plot(f,fftshift(abs(fft(y21))))
140 xlabel('Frequency(Hz)');ylabel('fft')
141 subplot(3,1,3)
142 [pxx2,wl] = pwelch(y21,[],[],512,fs2);
143 plot(wl,20*log10(pxx2))
144 xlabel('Frequency(Hz)');ylabel('Power(db)')
145
146 figure(2)
147 subplot(3,1,1)
148 plot(1:length(y2),y22)
149 title('N = 6 , fc = 50Hz')
150 xlabel('t(s)');ylabel('Amplitude')
151 subplot(3,1,2)
152 f = fs2*(-length(y2)/2:length(y2)/2-1);
153 plot(f,fftshift(abs(fft(y22))))
154 xlabel('Frequency(Hz)');ylabel('fft')
155 subplot(3,1,3)
156 [pxx2,wl] = pwelch(y22,[],[],512,fs2);
157 plot(wl,20*log10(pxx2))
158 xlabel('Frequency(Hz)');ylabel('Power(db)')
159
160 figure(3)
161 subplot(3,1,1)
162 plot(1:length(y2),y23)
163 title('N = 6 , fc = 100Hz')
164 xlabel('t(s)');ylabel('Amplitude')
165 subplot(3,1,2)
166 f = fs2*(-length(y2)/2:length(y2)/2-1);
167 plot(f,fftshift(abs(fft(y23))))
168 xlabel('Frequency(Hz)');ylabel('fft')
169 subplot(3,1,3)
170 [pxx2,wl] = pwelch(y23,[],[],512,fs2);

```

```

171 plot(w1,20*log10(pxx2))
172 xlabel('Frequency(Hz)');ylabel('Power(db)')
173
174 figure(4)
175 subplot(3,1,1)
176 plot(1:length(y2),y24)
177 title('N = 6 , fc = 400Hz')
178 xlabel('t(s)');ylabel('Amplitude')
179 subplot(3,1,2)
180 f = fs2*(-length(y2)/2:length(y2)/2-1);
181 plot(f,fftshift(abs(fft(y24))))
182 xlabel('Frequency(Hz)');ylabel('fft')
183 subplot(3,1,3)
184 [pxx2,w1] = pwelch(y24,[],[],512,fs2);
185 plot(w1,20*log10(pxx2))
186 xlabel('Frequency(Hz)');ylabel('Power(db)')
187
188 %Function need to be used
189 function res = get_wave(in, fs, N, fl, fh, fc)
190     res = zeros(1,length(in));
191     dl = log10(fl/165.4 + 1)/0.06;
192     dh = log10(fh/165.4 + 1)/0.06;
193     d = (dh - dl)/N;
194     len = length(in)/fs;
195     t = (0:1/fs:len - 1/fs);
196     for n = 1:N
197         l = 165.4*(10^(0.06*(dl + d*(n-1)))-1);
198         h = 165.4*(10^(0.06*(dl + d*n))-1);
199         [b, a] = butter(4, [l, h]/(fs/2));
200         wave = filter(b, a, in);
201         wave = abs(wave);
202         [x, y] = butter(4, fc/(fs/2));
203         env = filter(x, y, wave);
204         sin_sig = sin(2 * pi * (l + (h - l)/2)* t);
205         res = sin_sig.* env + res;
206     end
207     res = res*norm(in)/norm(res);
208 end

```