北京大学

本科实验报告

课程名称：机器感知实验

姓　　名：金镇雄

学　　院：元培

　　系：智能科学系

专　　业：智能科学与技术

年　　级：19

学　　号：1900094619

指导教师：曲天书

职　　称：副教授

2022 年　　4 月　　12 日

实验三、语音增强

一、实验目的

- 使用 MATLAB 进行声音文件读入和播放
- 利用基本谱减法从含噪语音中提取目标语音信息

二、实验要求

- 界面清晰美观
- 可播放原始声音，含噪声音及增强后声音
- 可选择噪音种类
- 可输入信噪比
- 实验结果分析
- 实验讨论

三、实验原理

谱减法是语音增强算法中比较成熟、常用的算法之一。谱减法具有算法简单、运算量小的特点，便于实现快速处理，往往能够获得较高的输出信噪比，所以被广泛采用。该算法假设噪声是加性、统计平稳的，利用加性噪声与语音不相关的特点，用带噪信号的频谱减去噪声信号的频谱，得到纯净语音。得到纯净信号的幅度谱后，结合带噪语音相位，近似带替纯净语音相位，从而得到近似的纯净语音。

设$s(m),n(m),y(\mathrm{m})$分别表示语音信号、噪声信号、带噪信号。假设噪声是与语音不相关的加性噪声，则得到的带噪信号的加性模型为：$\mathrm{y(m)=s(m)+n(m)}$。利用傅里叶变换将上式转换到频域，有：$Y(\omega)=S(\omega)+N(\omega)$。

对应的功率谱为：$|\mathrm{Y}(\omega)|^2=|\mathrm{S}(\omega)|^2+|\mathrm{N}(\omega)|^2+|\mathrm{S}(\omega)||\mathrm{N}(\omega)|^*+|\mathrm{S}(\omega)|^*|\mathrm{N}(\omega)|$

由 S 和 N 的独立性假设，得

$$\left|\hat{\mathrm{S}}(\omega)\right|^2=|Y(\omega)|^2-E[|N(\omega)|^2]$$

由于带噪信号的频谱与估计噪声频谱相减后可能出现负值，所以在基本谱减法中将负值全部设置为 0，这样得到的结果作为输出降噪语音的频谱。其公式如下：

$$\mathrm{D}(\omega)=|Y(\omega)|^2-E[|N(\omega)|^2]$$

$$\left|\hat{\mathrm{S}}(\omega)\right|^2=\begin{cases}\mathrm{D}(\omega), & \text{if } \mathrm{D}(\omega)>0\\ 0\end{cases}$$

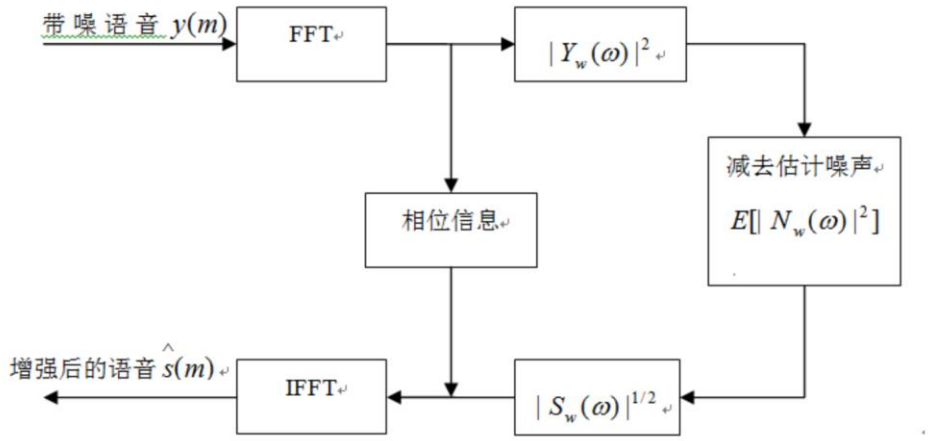
其缺点是谱相减后会有残余的噪声存在，产生“音乐噪声”。为了改善这种情况，可以使用 Berouti 谱减法。该算法多两个参数 α 和 β，分别表示过减因子和频谱下限阈值参数。其公式如下：

$$D(\omega) = |Y(\omega)|^2 - \alpha\, E[|N(\omega)|^2]$$

$$|\hat{S}(\omega)|^2 = \begin{cases} D(\omega), & \text{if } D(\omega) > (\alpha+\beta)\, E[|N(\omega)|^2] \\ \beta\, E[|N(\omega)|^2], & \text{else} \end{cases}$$

若 α 大于 1，则可以保证相比于基本谱减法能够去除大部分的噪声，减少残余噪声。由于谱相减后差值还会可能为负值，所以设置一个语音的下限值 $\beta\, E[|N(\omega)|^2]$。将相减后的幅值小于此下限值的统一设置为这个固定值。与基本谱减法相比 Berouti 算法的残余峰值没有那么显著，从而减小了“音乐噪声”的影响。

基本谱减法的流程图如下：

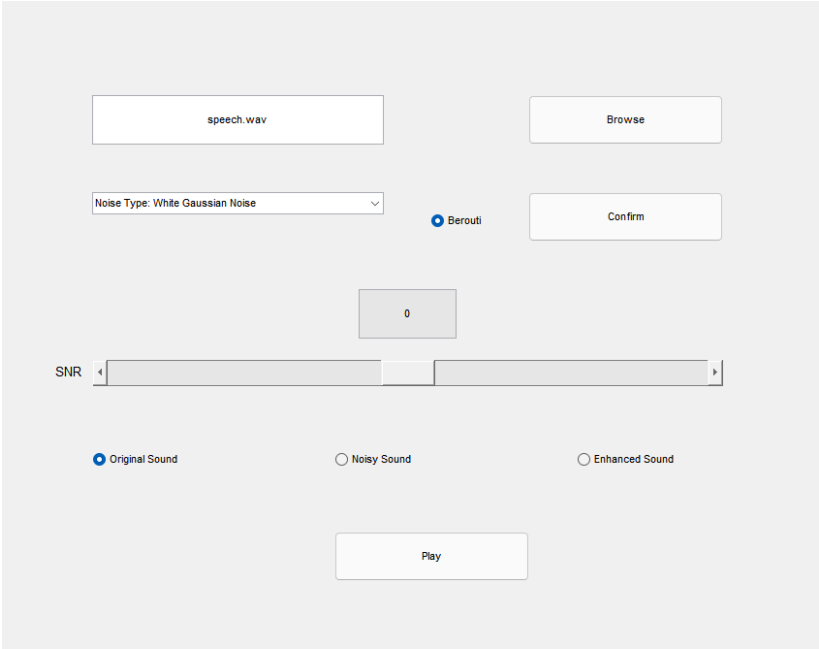


四、 实验仪器设备

PC 机、耳机、一个语音文件

五、 实验内容和步骤

1. GUI 界面设计



可以选择声音文件和噪声类型。界面中间有一个 berouti 按钮，其值为 0 时使用基本谱减法去噪，值为 1 时使用 Berouti 谱减法去噪。信噪比 SNR 可以由 editable_text 以及 slider 设定。

2. **读入声音文件**

3. **按信号种类和信噪比加入噪声**

本实验中噪声类型包括高斯白噪声、SSN 和粉红噪声。在原始语音信号中加高斯白噪声以 MATLAB 的 awgn()函数实现，其余两个噪声的加噪处理是由(.wav)形式的噪声文件和 add_noise()函数（参见代码）实现的。

本实验中信噪比可选范围为-5~+5（dB）。

4. **分帧**

本实验中帧长和重叠部分长度分别设置为 200 和 120，窗函数为汉明窗。注意帧合并时也需要重叠。

5. **利用静音段估计噪声能量**

本实验假设带噪信号的静音段是前 2s。由于提供的语音信号前静音段过小，加噪前在原始信号的前端补零。已知静音段时长为 2s，对应的帧数为 nSilence，可以求出该静音段的平均能量为：$E[|N(\omega)|^2]=\frac{1}{nSilence}\sum_{i-1}^{nSilence}|n(m)|^2$。

6. **基本谱减法——谱减**

对每一帧进行傅里叶变换，保留其相位信息，计算其功率谱。基于以下公式

$$D(\omega)=|Y(\omega)|^2-E[|N(\omega)|^2]$$

$$|\hat{S}(\omega)|^2=\begin{cases}D(\omega), & if\ D(\omega)>0\\ 0\end{cases}$$

在频域将带噪信号的功率谱减去估计噪声的功率谱进行谱减，得到语音的功率谱估计，开放后即可得到去噪信号的幅度估计。最终将幅度信息与以前保留的相位信息相结合，做逆傅里叶变换恢复时域信号。由于人耳对相位的感觉不灵敏，相位恢复时可使用带噪信号的相位信息。

7. **Berouti 谱减法——谱减**

Berouti 谱减法的算法实现与基本谱减法基本一致，只是在谱减处多两个参数 alpha 和 beta，可按照如下公式进行谱减：

$$D(\omega)=|Y(\omega)|^2-\alpha\,E[|N(\omega)|^2]$$

$$|\hat{S}(\omega)|^2=\begin{cases}D(\omega), & \text{if } D(\omega)>(\alpha+\beta)\,E[|N(\omega)|^2]\\ \beta\,E[|N(\omega)|^2], & \text{else}\end{cases}$$

Alpha 和 beta 的值与信噪比有关，可以根据以下式求值：

$$\alpha=4-\frac{3}{20}SNR\ (-5\le \mathrm{SNR}\le 20),$$

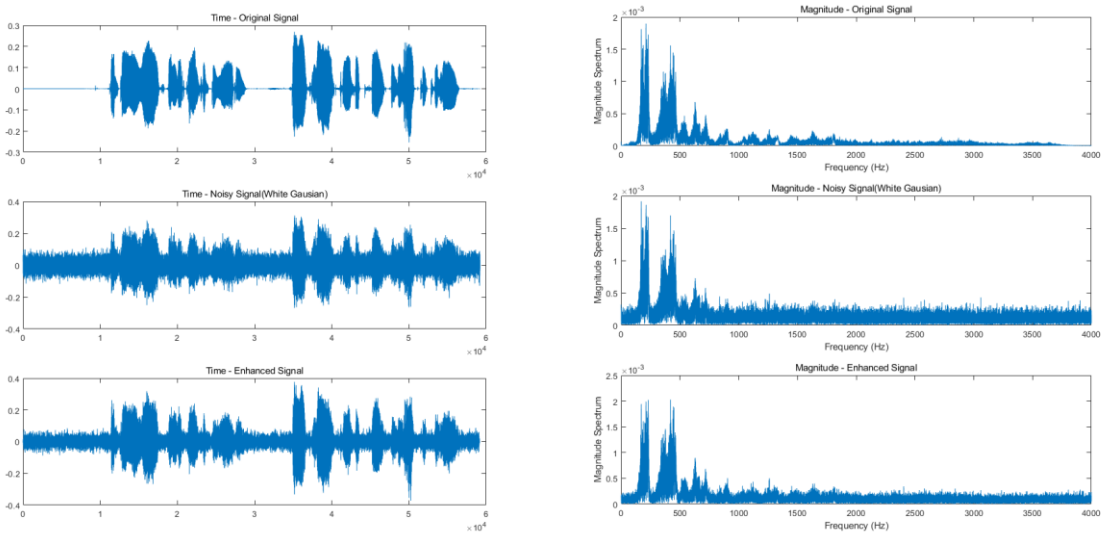
$0.02\le\beta\le 0.06\ (High\ Noise\ Level),\ 0.005\le\beta\le 0.02\ (High\ Noise\ Level)$。

8. **输出增强后声音**

六、实验结果和分析

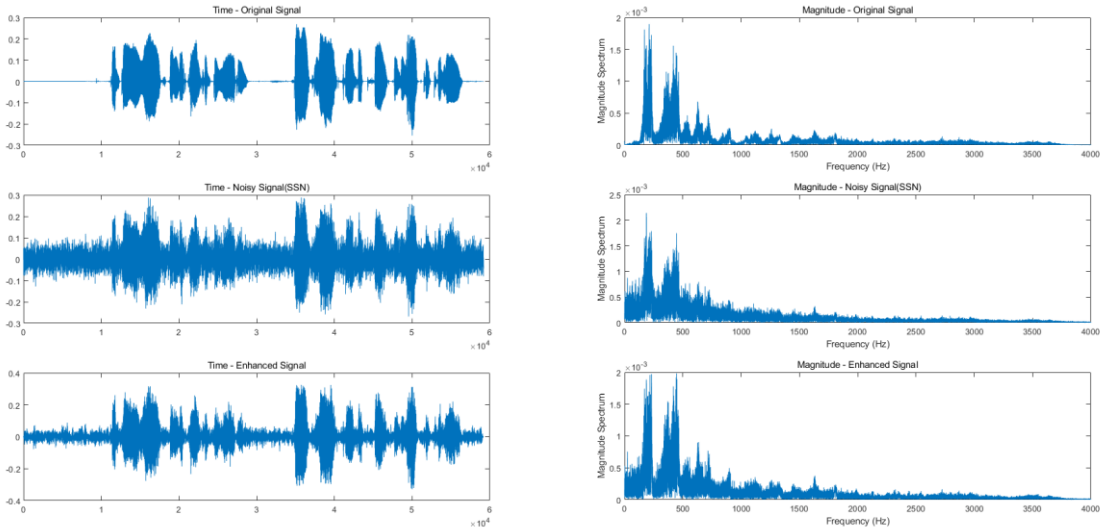
1. 基本谱减法——高斯白噪声（SNR 为 3dB）

以下两个图分别为噪声是信噪比为 3dB 的高斯白噪声时原始信号、带噪信号、增强信号的时域图和频谱图。语音增强时使用的算法为基本谱减法。可见去噪以后噪声亦然存在，没有得到与原始信号一致的信号。但需要注意，此噪声不是高斯白噪声而是“音乐信号”，虽然存在部分噪声，但与带噪语音相比有明显语音增强效果。



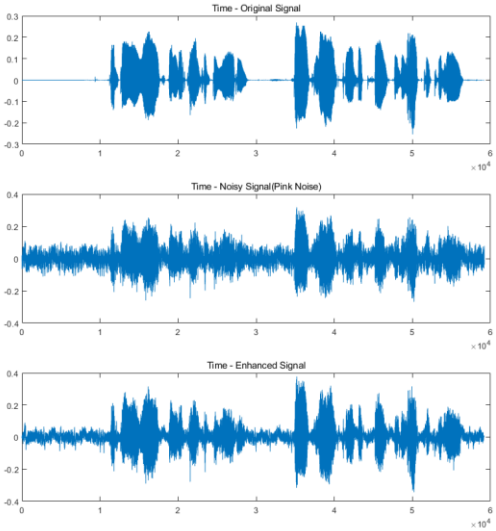
2. SSN 噪声（SNR 为 3dB）

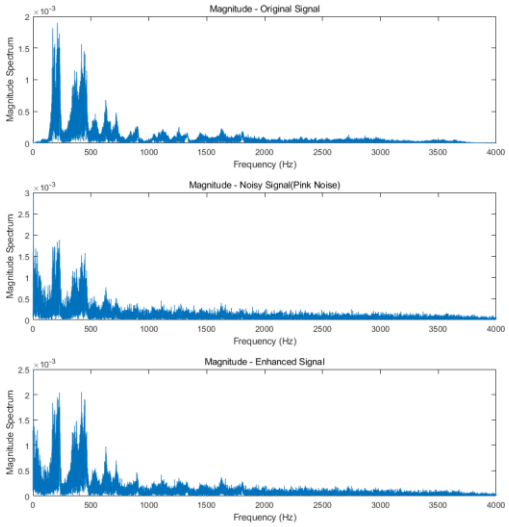
以下两个图分别为噪声是信噪比为 3dB 的 SSN 噪声时原始信号、带噪信号、增强信号的时域图和频谱图。与高斯白噪声为噪声时相比其语音增强效果更明显，但可以看到它没有完全去除频率为 0~1000Hz 的噪声。



3. 粉红噪声（SNR 为 3dB）

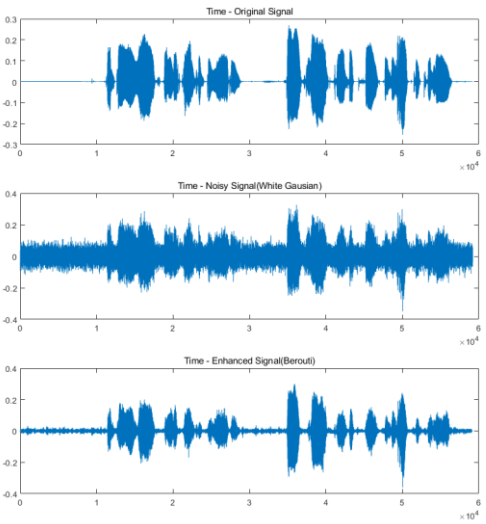
以下两个图分别为噪声是信噪比为 3dB 的粉红噪声时原始信号、带噪信号、增强信号的时域图和频谱图。

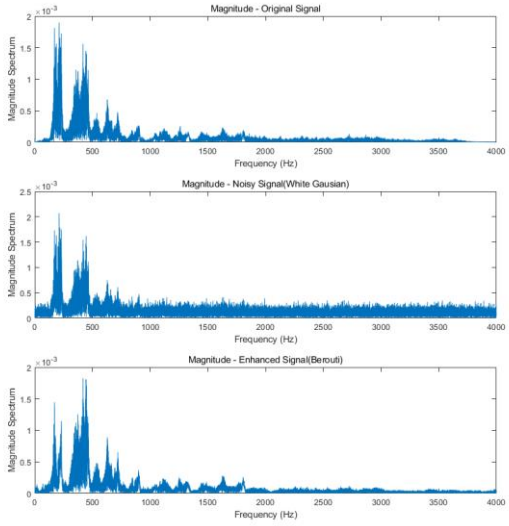




4. Berouti 谱减法——高斯白噪声（SNR 为 3dB）

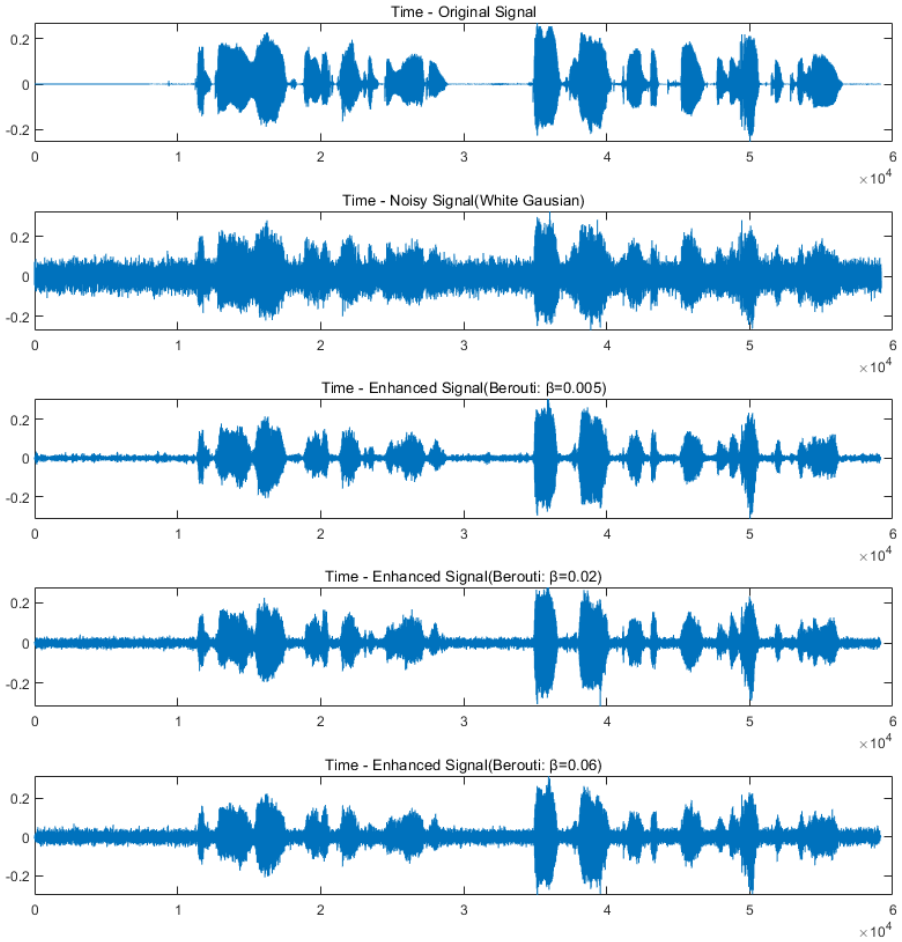
以下两个图分别为噪声是信噪比为 3dB 的高斯白噪声，采用 Berouti 谱减法去噪时，原始信号、带噪信号、增强信号的时域图和频谱图。Berouti 谱减法的两个参数 alpha 和 beta 分别取了 3.55 和 0.003。实验结果明显由于基本谱减法，基本上能够去除声音噪声。

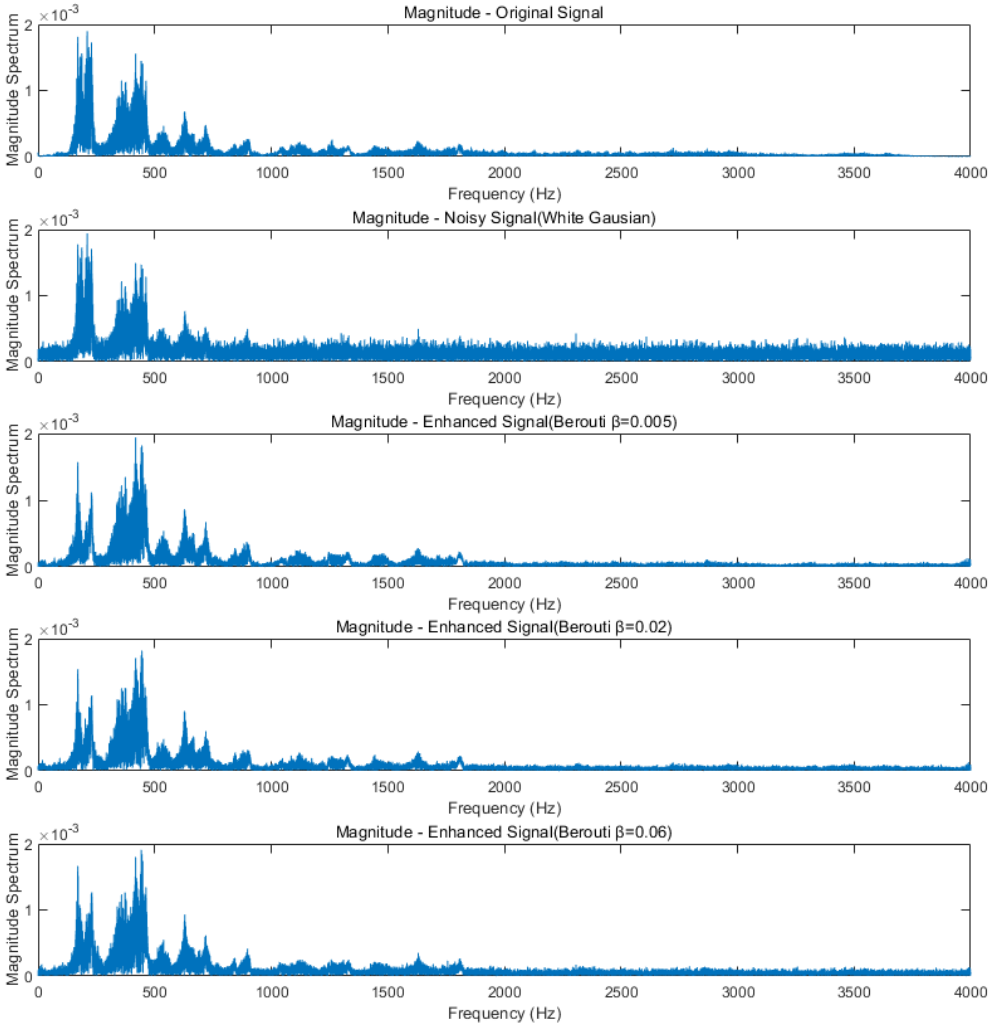




5. Berouti 谱减法与 β 的取值

Berouti 谱减法的效果受到 α 和 β 取值的影响。以下图中第一个和第二个图分别是原始语音信号和带噪语音信号的时域图，接下来三个图分别是 beta 取 0.005、0.02、0.06 时去噪处理后得到的增强信号的时域图。显然，beta 取 0.005 时语音增强效果最佳，beta 取 0.02 和 0.06 时与基本谱减法类似，亦然存在很多音乐噪声。





七、讨论

1. 谱减法存在的问题

A. 音乐噪声

从实验结果中可以发现采用基本谱减法去噪后时域波形中依然存在一些噪声。在谱减法算法中，噪音的估计是静音段的噪音强度取平均值得到的，因此有的地方的噪音强度可能大于平均值，谱减后有残留的噪音存在。此外，带噪语音的功率谱与估计出来的噪声的功率谱相减会出现负值，说明对噪声出现了过估计问题，在基本谱减法中将其负值设为 0，以保证非负的幅度谱。但是对负值的这种处理，会导致信号帧频谱的随机位置上出现小的，独立的峰值，称为音乐噪声。由谱减所产生的音乐噪音与语音信号不相关，是由具有随即频率和幅度的窄带信号所组成。而 Berouti 谱减法正解决了此问题，能去除大部分音乐噪音。该算法的

参数 alpha 为过减因子，主要影响语音谱的失真程度；Beta 是谱下限参数，可以控制残留噪声的多少以及音乐噪声的大小。

B. **相位信息的误差**

相位信息使用的是带噪语音的相位。语音信号经过傅里叶变换后由实数域转换为复数域，降噪时只考虑了幅度值，忽略了相位信息，在不同的信噪比情况下，误差表现不一样：高信噪比时，误差小，低信噪比时，误差大。

2. **改进的谱减算法**

A. **非线性谱减法**

用一个频率相关的减法因子来处理不同类型的噪声。

B. **多带谱减法**

该算法中将语音频谱划分为 N 个互不重叠的子带，谱减法在每个子带独立运行。多带谱减与非线性谱减的主要区别在于对过减因子的估计。多带算法针对频带估计减法因子，而非线性谱减算法针对每一个频点，导致频点上的信噪比可能有很大变化。这种剧烈变化是谱减法中所遇到的语音失真（音乐噪声）的原因之一。相反，子带信噪比变化则不会特别剧烈。

C. **扩展谱减法**

基于自适应维纳滤波与谱减原理的结合。维纳滤波用于估计噪声谱，然后从带噪语音信号中减去该噪声谱。

八、代码

```
function varargout = lab3(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @lab3_OpeningFcn, ...
                   'gui_OutputFcn',  @lab3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before lab3 is made visible.
function lab3_OpeningFcn(hObject, eventdata, handles, varargin)

% Choose default command line output for lab3
```

```
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = lab3_OutputFcn(hObject, eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in browse_button.
function browse_button_Callback(hObject, eventdata, handles)
global audio;
global fs;
global file;
[file,~] = uigetfile('*.wav','????ëåÙþËì£°');
[audio,fs] = audioread(file);
set(handles.original_sound,'value',1);
set(handles.audio_file,'String',file);

function audio_file_Callback(hObject, eventdata, handles)
global file;
file = get(hObject,'String');

% --- Executes during object creation, after setting all properties.
function audio_file_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in noise_type.
function noise_type_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function noise_type_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,'value',1);

% --- Executes on slider movement.
function snr_slider_Callback(hObject, eventdata, handles)
global snr;
snr = get(hObject,'Value')-5;
set(handles.snr_text,'String',num2str(snr));

% --- Executes during object creation, after setting all properties.
function snr_slider_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
set(hObject,'Value',5)

function snr_text_Callback(hObject, eventdata, handles)
global snr;
```

```
input = str2double(get(hObject,'String'));
if input <= 5 && input >= -5
    snr = input;
    set(handles.snr_slider,'Value',snr+5);
else
    msgbox('Invalid Value: -5 <= SNR <= 5','warning','warn'); %error
message window
end

% --- Executes during object creation, after setting all properties.
function snr_text_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global snr;
snr = 0;
set(hObject,'string','0');

% --- Executes on button press in confirm_button.
function confirm_button_Callback(hObject, eventdata, handles)
global audio;
global noisy;
global enhanced;
global snr;
global fs;


st = 2;                                 % 静音段长度(s)
sil_frame = zeros(fs,1);
tmp_audio = [sil_frame;audio];          % 在音频信号前端加静音段

if get(handles.noise_type,'value') == 1
    noisy = awgn(tmp_audio,snr,'measured');
elseif get(handles.noise_type,'value') == 2
    noisy = add_noise(tmp_audio,fs,'SSN.wav',snr);
elseif get(handles.noise_type,'value') == 3
    noisy = add_noise(tmp_audio,fs,'pink_noise.wav',snr);
else
    noisy = awgn(tmp_audio,snr,'measured');
end



win_len = 200;
overlap = 120;
win = hamming(win_len);
inc = win_len - overlap;

nSilence = fix((st*fs-win_len)/inc +1);                 % ?ëåÓ«??
nframe = floor((length(noisy) - win_len) / inc) + 1;    % ãá???

% 计算噪声幅度
hfreq = win_len/2+1;
noise_energy = zeros(hfreq,1);
for k = 1 : nSilence
    tmp = noisy((1:win_len) + (k-1) * inc).*win;
    tmp_fft = fft(tmp);
    noise_energy = noise_energy + abs(tmp_fft(1:hfreq)).^2;
```

```
end
noise_energy_avg = noise_energy / nSilence;

% berouti 参数设定
alpha = 1;  % 过减因子
beta = 0;   % 频谱下限阈值参数
if get(handles.berouti_button,'value') == 1
    alpha = 4-snr*3/20
    if snr >= 0
       beta = 0.003
    else
       beta = 0.02
    end
end

% 增强语音
enhanced = zeros((nframe-1)*inc+win_len,1);
for k = 1 : nframe
    % 分帧
    tmp = noisy((1:win_len) + (k-1) * inc).*win;
    tmp_fft = fft(tmp);

    audio_phase = angle(tmp_fft(1:hfreq));      % 保留信号相位
    audio_energy = abs(tmp_fft(1:hfreq)).^2;    % 计算每帧能量

    % 谱减（减去估计噪声）
    sub_energy = zeros(hfreq,1);
    for m = 1:hfreq
       if  audio_energy(m) >= (alpha+beta) * noise_energy_avg(m)
           sub_energy(m) = audio_energy(m) - alpha * noise_energy_avg(m);
       else
           sub_energy(m) = beta * noise_energy_avg(m);
       end
    end

    % 取根号
    abs_freq = sqrt(sub_energy);

    % 加相位信息
    audio_freq = abs_freq .* exp(1j*audio_phase);

    % 上下翻转
    tmp_freq = [audio_freq ; conj(audio_freq(end-1:-1:2))];

    % ifft
    tpm_time = real(ifft(tmp_freq));

    % 帧合并
    s=(k-1)*inc+1;
    enhanced(s:s+win_len-1)=enhanced(s:s+win_len-1) + tpm_time;
end
```

```
% N = length(tmp_audio);
% N1=2^nextpow2(N);
% f1=fft(tmp_audio,N1);
% f2=fft(noisy,N1);
% f3=fft(enhanced,N1);
% f1=f1(1:N1/2);
% f2=f2(1:N1/2);
% f3=f3(1:N1/2);
% x1=abs(f1);
% x2=abs(f2);
% x3=abs(f3);
% p1=phase(f1);
% p2=phase(f2);
% p3=phase(f3);
% f=fs*(0:N1/2-1)/N1; %Frequency axis
%
% figure;
% subplot(311)
% plot(tmp_audio); title("Time - Original Signal")
% subplot(312)
% plot(noisy); title("Time - Noisy Signal(White Gausian)")
% subplot(313)
% plot(enhanced); title("Time - Enhanced Signal(Berouti)")
%
% figure;
% subplot(311)
% plot(f,(x1/N1));
% xlabel('Frequency (Hz)'); ylabel('Magnitude Spectrum');
% title("Magnitude - Original Signal")
% subplot(312)
% plot(f,(x2/N1));
% xlabel('Frequency (Hz)'); ylabel('Magnitude Spectrum');
% title("Magnitude - Noisy Signal(White Gausian)")
% subplot(313)
% plot(f,(x3/N1));
% xlabel('Frequency (Hz)'); ylabel('Magnitude Spectrum');
% title("Magnitude - Enhanced Signal(Berouti)")
%
% figure;
% subplot(311)
% plot(f,p1);
% xlabel('Frequency (Hz)'); ylabel('Phase Spectrum');
% title("Phase - Original Signal")
% subplot(312)
% plot(f,p2);
% xlabel('Frequency (Hz)'); ylabel('Phase Spectrum');
% title("Phase - Noisy Signal(White Gausian)")
% subplot(313)
% plot(f,p3);
% xlabel('Frequency (Hz)'); ylabel('Phase Spectrum');
% title("Phase - Enhanced Signal(Berouti)")

function y = add_noise(x,fs,type,SNR)
% add_noisem add determinated noise to a signal.
% X is signal, and its sample frequency is fs;
[n,fs1] = audioread(type);
if fs1~=fs
    tmp = resample(n,fs,fs1);
end
nx = size(x,1);
```

```
xlen = length(x);
tlen = length(tmp);
if xlen < tlen
    noise = tmp(1:nx);
elseif xlen == tlen
    noise = tmp;
else
    a = floor(xlen / tlen);
    m = mod(xlen,tlen);
    noise = [];
    for i = 1:a
        noise = [noise;tmp];
    end
    noise = [noise;tmp(1:m)];
end
noise = noise - mean(noise);
signal_power = 1/nx*sum(x.*x);
noise_variance = signal_power / ( 10^(SNR/10) );
noise=sqrt(noise_variance)/std(noise)*noise;
y = x + noise;

% --- Executes on button press in original_sound.
function original_sound_Callback(hObject, eventdata, handles)
set(handles.original_sound,'value',1);
set(handles.noisy_sound,'value',0);
set(handles.enhanced_sound,'value',0);

% --- Executes on button press in noisy_sound.
function noisy_sound_Callback(hObject, eventdata, handles)
set(handles.original_sound,'value',0);
set(handles.noisy_sound,'value',1);
set(handles.enhanced_sound,'value',0);

% --- Executes on button press in enhanced_sound.
function enhanced_sound_Callback(hObject, eventdata, handles)
set(handles.original_sound,'value',0);
set(handles.noisy_sound,'value',0);
set(handles.enhanced_sound,'value',1);

% --- Executes on button press in play_button.
function play_button_Callback(hObject, eventdata, handles)
global audio;
global noisy;
global enhanced;
global fs;
global player;

if get(handles.original_sound,'value')
    player = audioplayer(audio,fs);
elseif get(handles.noisy_sound,'value')
    player = audioplayer(noisy,fs);
elseif get(handles.enhanced_sound,'value')
    player = audioplayer(enhanced,fs);
end
player.play;


% --- Executes on button press in berouti_button.
function berouti_button_Callback(hObject, eventdata, handles)
```

```
set(handles.berouti_button,'value',1);
```