语音信号处理实验

班级：

学号：

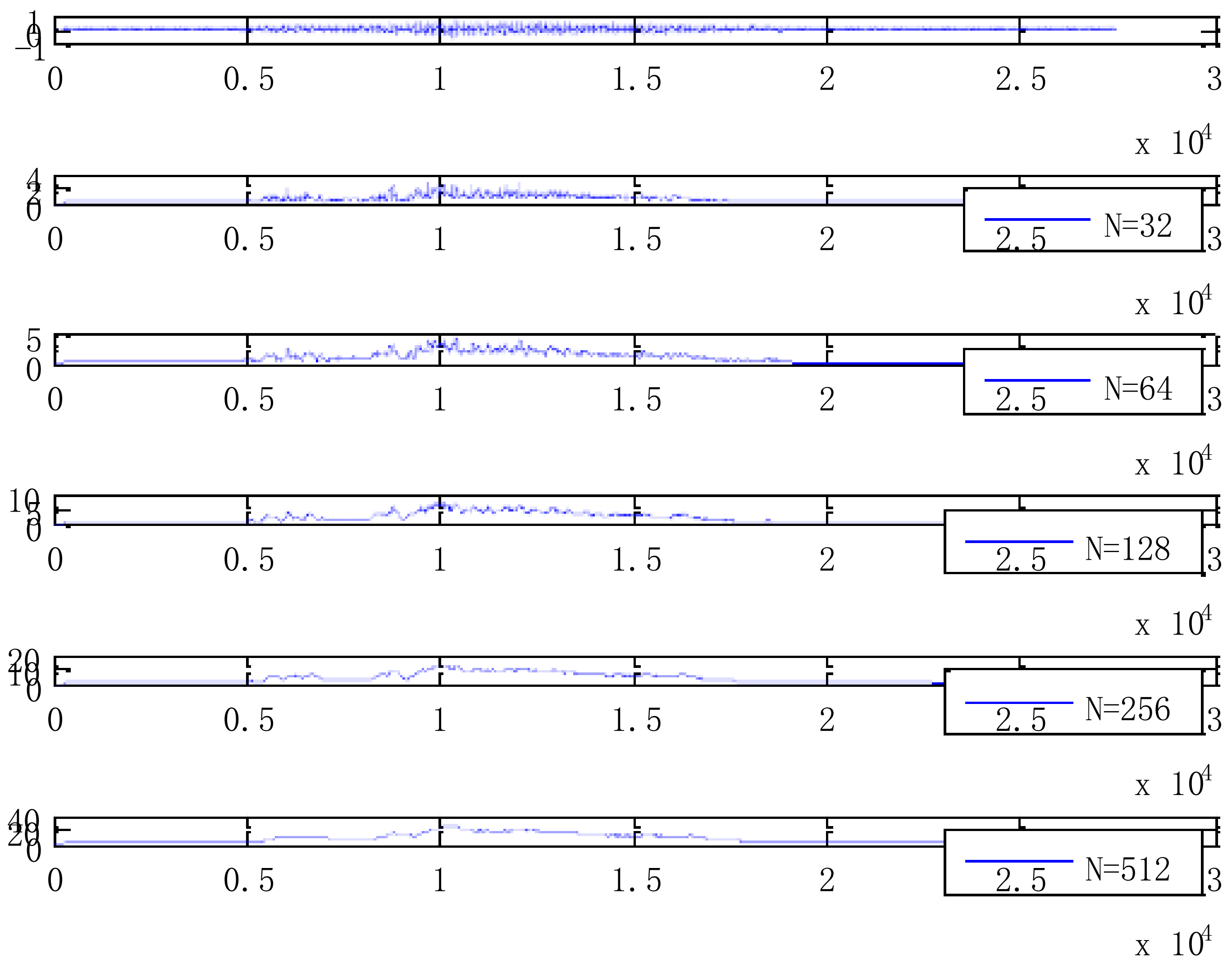
某某：

实验一基于 MATLAB 的语音信号时域特征分析〔2 学时〕

1〕短时能量

〔1〕加矩形窗

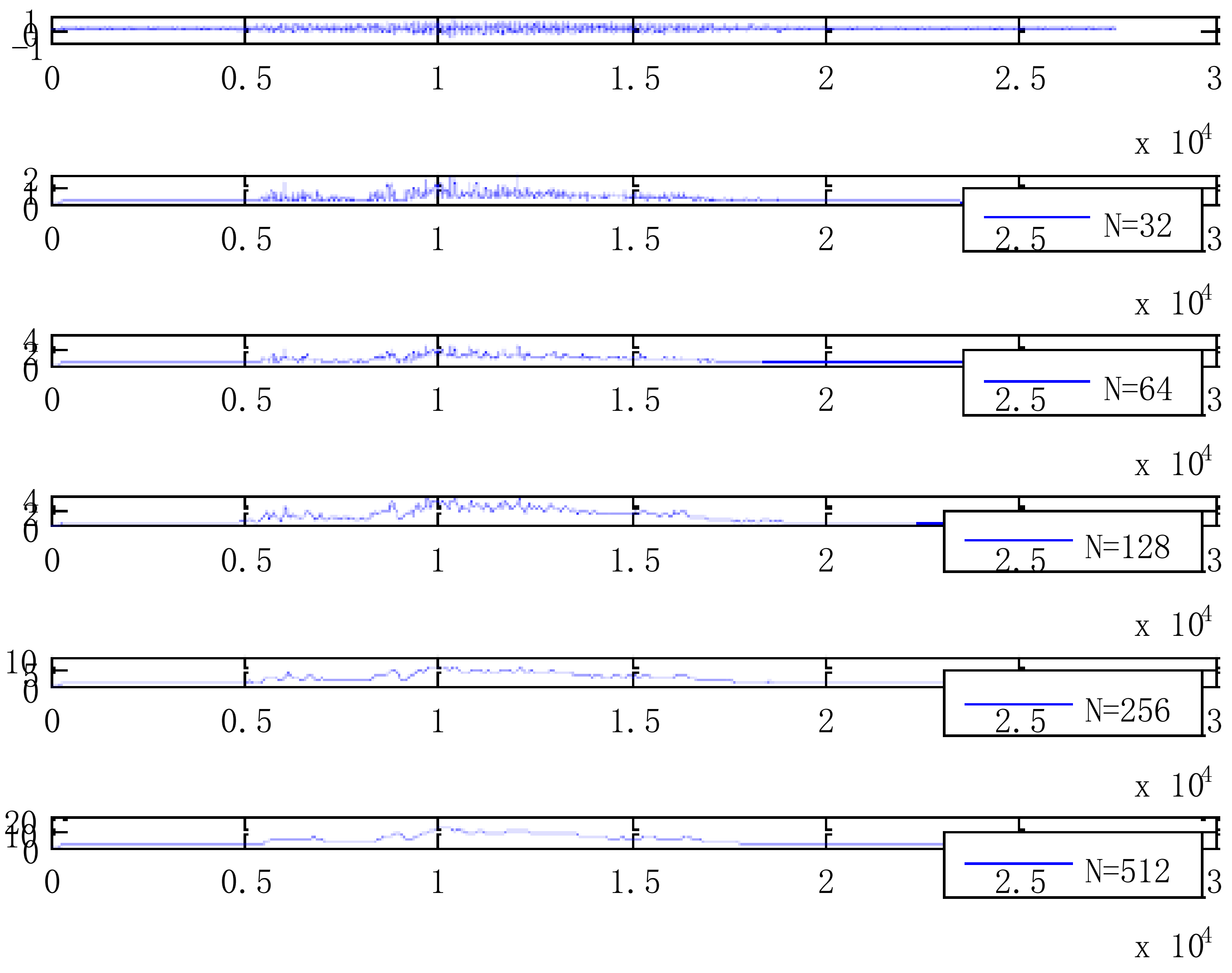
```
a=wavread('mike.wav');
  a=a(:,1);
  subplot(6,1,1),plot(a);
  N=32;
  for i=2:6
    h=linspace(1,1,2.^(i-2)*N);%形成一个矩形窗，长度为 2.^(i-2)*N
    En=conv(h,a.*a);%求短时能量函数En
     subplot(6,1,i),plot(En);
  if(i==2) ,legend('N=32');
  elseif(i==3), legend('N=64');
  elseif(i==4) ,legend('N=128');
  elseif(i==5) ,legend('N=256');
  elseif(i==6) ,legend('N=512');
  end
  end
```



〔2〕加汉明窗

```
a=wavread('mike.wav');
  a=a(:,1);
  subplot(6,1,1),plot(a);
  N=32;
```

```
for i=2:6
  h=hanning(2.^(i-2)*N);%形成一个汉明窗，长度为 2.^(i-2)*N
  En=conv(h,a.*a);%求短时能量函数En
   subplot(6,1,i),plot(En);
if(i==2), legend('N=32');
elseif(i==3), legend('N=64');
elseif(i==4) ,legend('N=128');
elseif(i==5) ,legend('N=256');
elseif(i==6) ,legend('N=512');
end
end
```



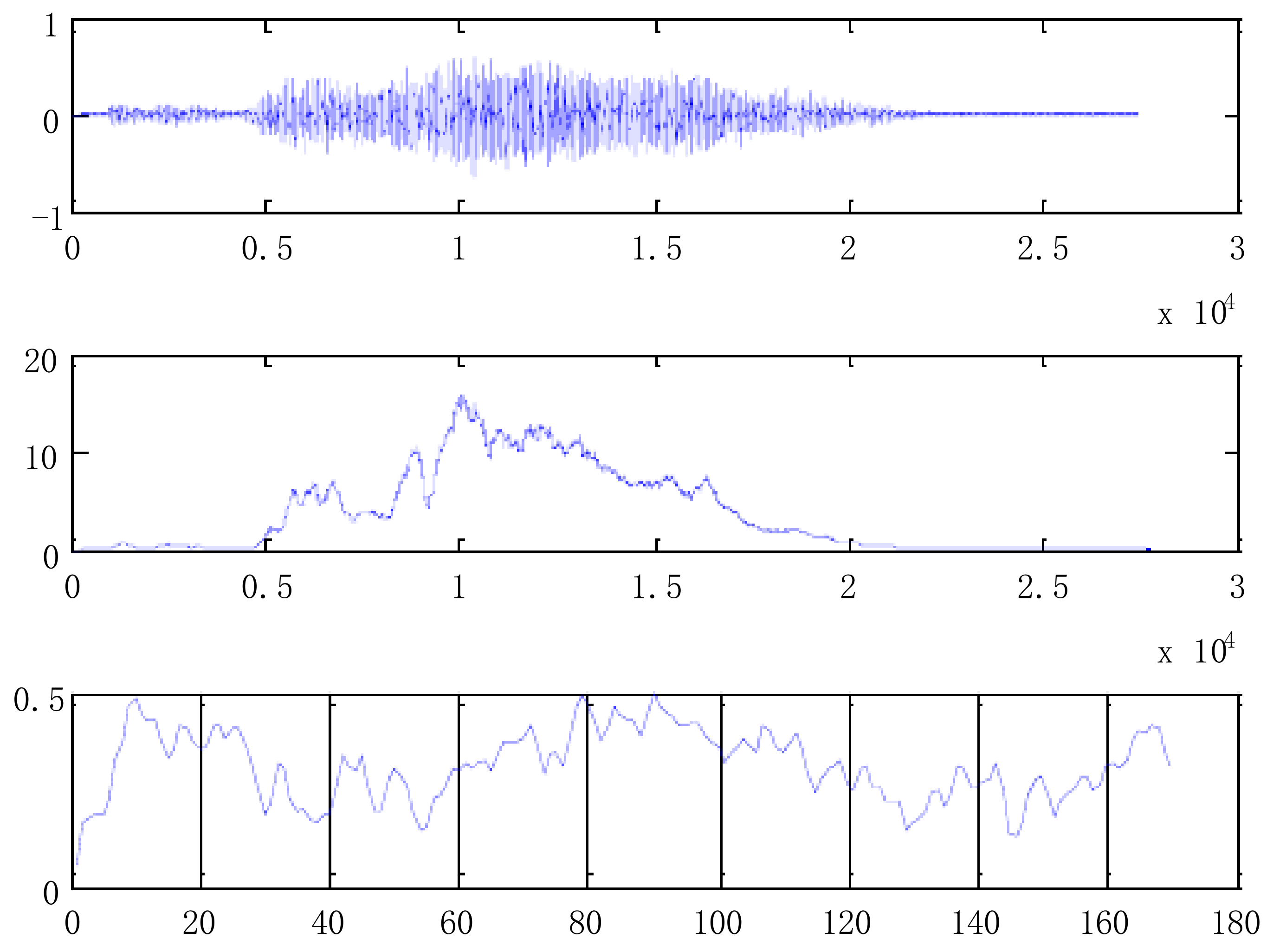
2）短时平均过零率

```
a=wavread('mike.wav');
a=a(:,1);
n=length(a);
N=320;
subplot(3,1,1),plot(a);
h=linspace(1,1,N);
En=conv(h,a.*a);%求卷积得其短时能量函数 En
subplot(3,1,2),plot(En);

for i=1:n-1
if  a(i)>=0
         b(i)= 1;
```

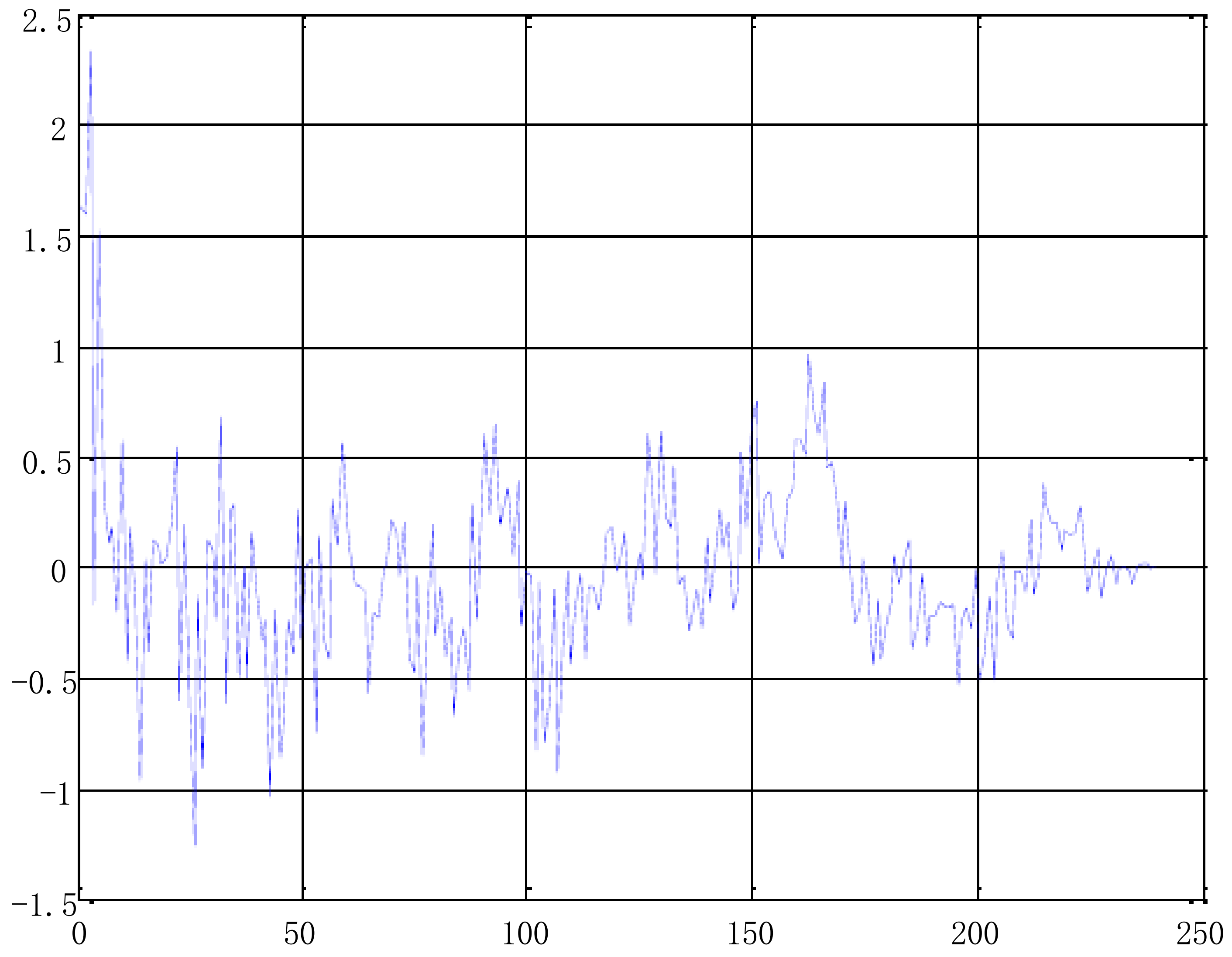
```
else
        b(i) = -1;
end
ifa(i+1)>=0
        b(i+1)=1;
else
         b(i+1)= -1;
end
       w(i)=abs(b(i+1)-b(i));%求出每相邻两点符号的差值的绝对值

end
k=1;
j=0;
while (k+N-1)<n
   Zm(k)=0;
for i=0:N-1;
    Zm(k)=Zm(k)+w(k+i);
end
   j=j+1;
   k=k+N/2; %每次移动半个窗
end
for w=1:j
      Q(w)=Zm(160*(w-1)+1)/(2*N); %短时平均过零率
end
 subplot(3,1,3),plot(Q),grid;
```



3）自相关函数

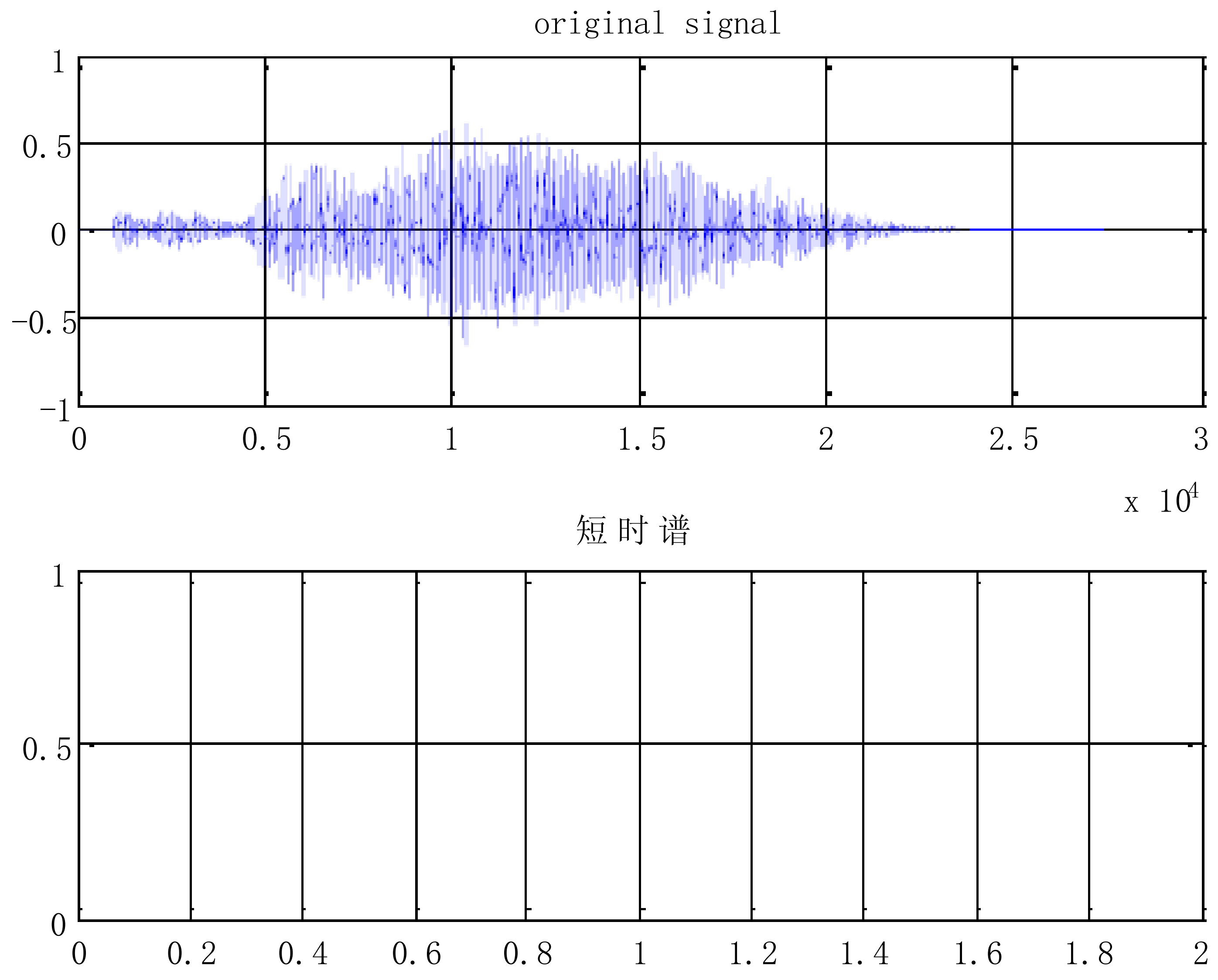
```
N=240
    y=wavread('mike.wav');
    y=y(:,1);
    x=y(13271:13510);
    x=x.*rectwin(240);
    R=zeros(1,240);
   for k=1:240
        for n=1:240-k
            R(k)=R(k)+x(n)*x(n+k);
        end
    end
j=1:240;
      plot(j,R);
      grid;
```



实验二基于 MATLAB 分析语音信号频域特征

1）短时谱

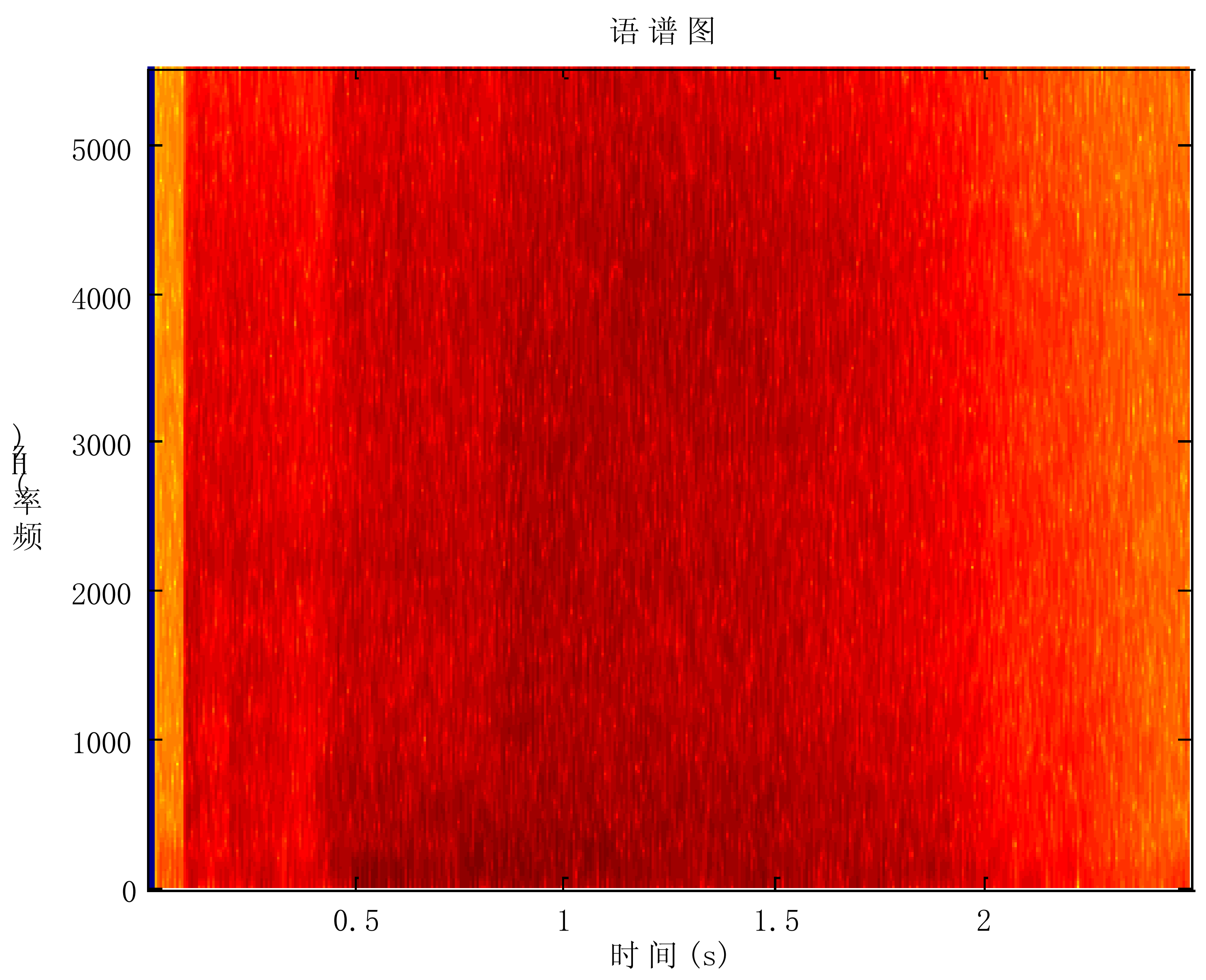
```
clear
   a=wavread('mike.wav');
   a=a(:,1);
   subplot(2,1,1),
    plot(a);title('original signal');
    grid
    N=256;
    h=hamming(N);
    for m=1:N
        b(m)=a(m)*h(m)
     end
    y=20*log(abs(fft(b)))
    subplot(2,1,2)
   plot(y);title('短时谱');
   grid
```



2）语谱图

```
[x,fs,nbits]=wavread('mike.wav')
x=x(:,1);
   specgram(x,512,fs,100);
   xlabel('时间(s)');
   ylabel('频率(Hz)');
```

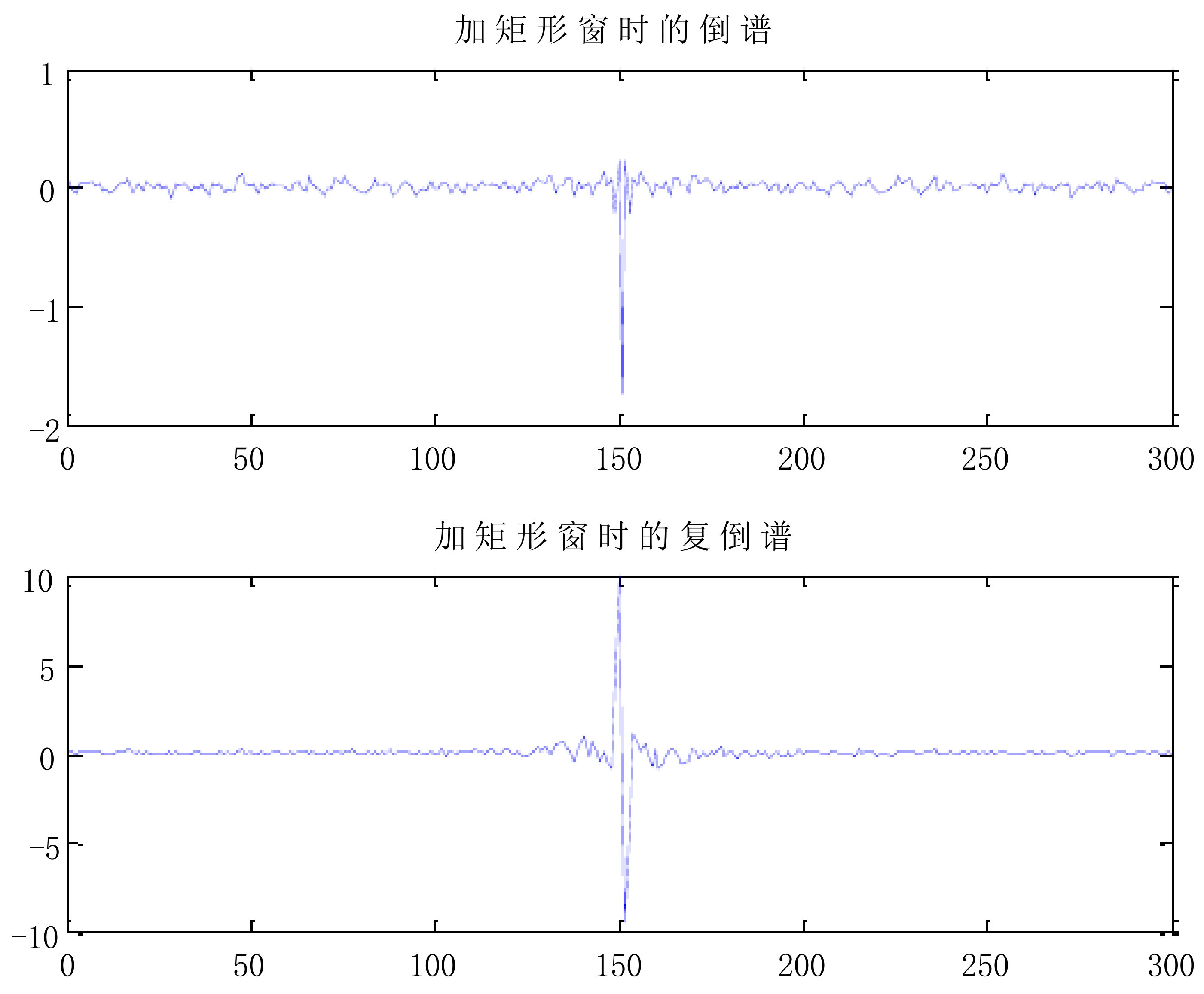
```
title('语谱图');
```



3）倒谱和复倒谱

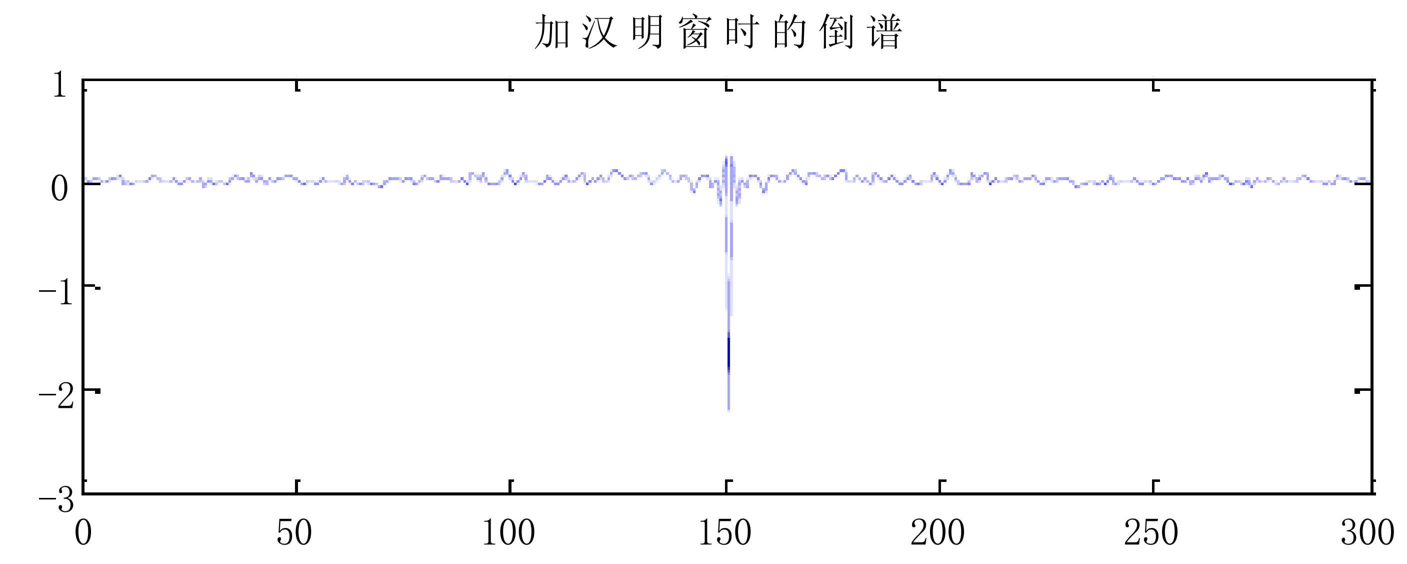
〔1〕加矩形窗时的倒谱和复倒谱

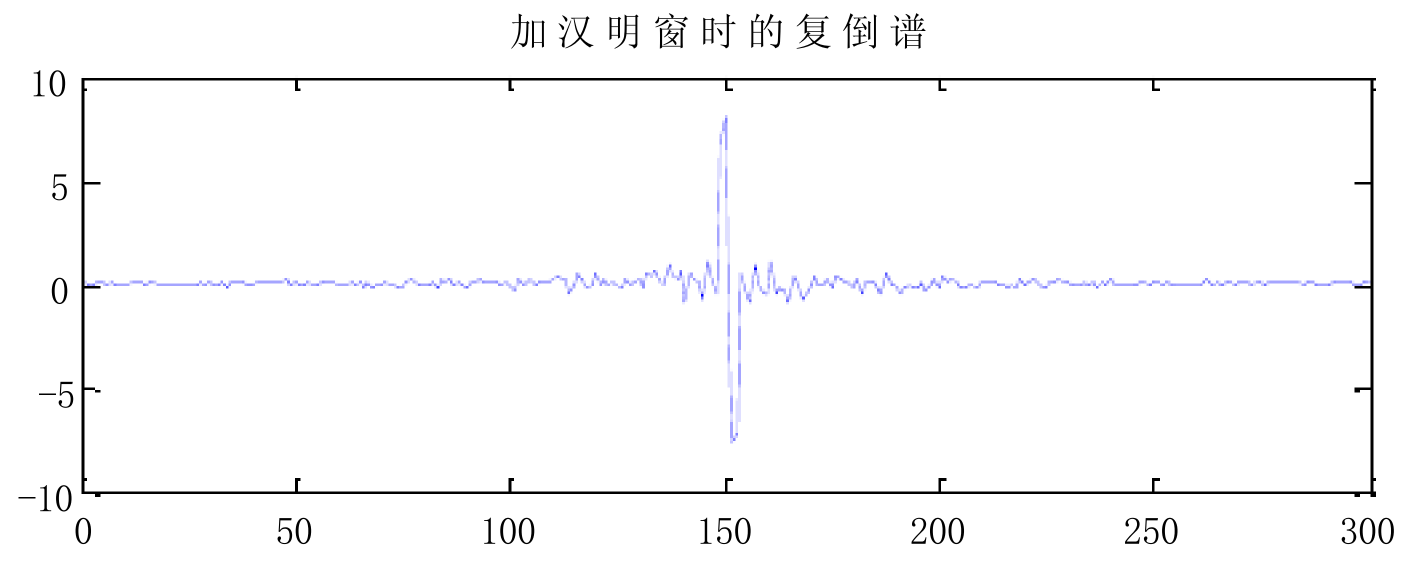
```
clear
a=wavread('mike.wav',[4000,4350]);
a=a(:,1);
N=300;
h=linspace(1,1,N);
for m=1:N
 b(m)=a(m)*h(m);
 end
 c=cceps(b);
 c=fftshift(c);
 d=rceps(b);
 d=fftshift(d);
 subplot(2,1,1)
plot(d);title('加矩形窗时的倒谱')
subplot(2,1,2)
plot(c);title('加矩形窗时的复倒谱')
```



〔2〕加汉明窗时的倒谱和复倒谱

```
clear
a=wavread('mike.wav',[4000,4350]);
a=a(:,1);
N=300;
h=hamming(N);
for m=1:N
 b(m)=a(m)*h(m);
 end
 c=cceps(b);
 c=fftshift(c);
 d=rceps(b);
 d=fftshift(d);
 subplot(2,1,1)
plot(d);title('加汉明窗时的倒谱')
subplot(2,1,2)
plot(c);title('加汉明窗时的复倒谱')
```


加汉明窗时的倒谱
1
0
−1
−2
−3
0
50
100
150
200
250
300


加汉明窗时的复倒谱
10
5
0
−5
−10
0
50
100
150
200
250
300

实验三基于 MATLAB 的 LPC 分析

```
MusicSource = wavread('mike.wav');
MusicSource=MusicSource(:,1);
Music_source = MusicSource';
N = 256;% window length, N = 100 -- 1000;
Hamm = hamming(N); % create Hamming window
frame = input('请键入想要处理的帧位置 = ');
% origin is current frame
origin = Music_source(((frame - 1) * (N / 2) + 1):((frame - 1) * (N / 2) + N));
Frame = origin .* Hamm';

%
%Short Time Fourier Transform
%
[s1,f1,t1] = specgram(MusicSource,N,N/2,N);
[Xs1,Ys1] = size(s1);
for i = 1:Xs1
    FTframe1(i) = s1(i,frame);
end

N1 = input('请键入预测器阶数 = ');% N1 is predictor's order
[coef,gain] = lpc(Frame,N1);% LPC analysis using Levinson-Durbin recursion
est_Frame = filter([0 -coef(2:end)],1,Frame);% estimate frame(LP)
FFT_est = fft(est_Frame);
err = Frame - est_Frame;% error
% FFT_err = fft(err);
subplot(2,1,1),plot(1:N,Frame,1:N,est_Frame,'-r');grid;title('原始语音帧vs.预测后语音帧')
subplot(2,1,2),plot(err);grid;title('误差');
pause

%subplot(2,1,2),plot(f',20*log(abs(FTframe2)));title('短时谱')

%
% Gain solution using G^2 = Rn(0) - sum(ai*Rn(i)),i = 1,2,...,P
%
fLength(1 : 2 * N) = [origin,zeros(1,N)];
Xm = fft(fLength,2 * N);
X = Xm .* conj(Xm);
Y = fft(X , 2 * N);
Rk = Y(1 : N);
PART = sum(coef(2 : N1 + 1) .* Rk(1 : N1));
G = sqrt(sum(Frame.^2) - PART);
```

```
A = (FTframe1 - FFT_est(1 : length(f1'))) ./ FTframe1  %inverse;filter A(Z)
subplot(2,1,1),plot(f1',20*log(abs(FTframe1)),f1',(20*log(abs(1 ./ A))),'-r');grid;title('短时谱');
subplot(2,1,2),plot(f1',(20*log(abs(G ./ A))));grid;title('LPC谱');
pause

%plot(abs(ifft(FTframe1 ./ (G ./ A))));grid;title('excited')
%plot(f1',20*log(abs(FFT_est(1 : length(f1')) .* A / G )));grid;
%pause

%
% find_pitch
%
temp = FTframe1 - FFT_est(1 : length(f1'));

% not move higher frequnce
pitch1 = log(abs(temp));
pLength = length(pitch1);
result1 = ifft(pitch1,N);

% move higher frequnce
pitch1((pLength - 32) : pLength) = 0;
result2 = ifft(pitch1,N);

% direct do real cepstrum with err
pitch = fftshift(rceps(err));
origin_pitch = fftshift(rceps(Frame));
subplot(211),plot(origin_pitch);grid;title('原始语音帧倒谱(直接调用函数)');
subplot(212),plot(pitch);grid;title('预测误差倒谱(直接调用函数)');
pause

subplot(211),plot(1:length(result1),fftshift(real(result1)));grid;title('预测误差倒谱(根据定义编
写，没有去除高频分量)');
subplot(212),plot(1:length(result2),fftshift(real(result2)));grid;title('预测误差倒谱(根据定义编
写，去除高频分量)');
```