Lab8 Algorithms for Estimating Speech Parameters

张旭东 12011923

1.Introduction

The lab is mainly divided into two parts. The first part is to test and master the processing of the median smoothing on speech parameter estimation. The second part is to test the commonly-used algorithms on pitch estimation, including autocorrelation method and ceptrum method. As we all know, the algorithms are often a combination of fundamental knowledge of the speech signal, digital signal processing theory, statistics and heuristics.

2.lab result and analysis

2.1 Probelm1

The purpose of this MATLAB exercise is to compare linear, median, and combination smoothers on short-time zero-crossing estimates. Compute the short-time zero-crossing rate(per 10 `msec` of speech) using a frame length of 10 `msec` and a frame shift of 5 `msec`. The function to calculate short-time zero-crossing rate is as below:

```
function [E,M,Z]=EMZ(s,L,R,window)
        N=length(s);
        N_E=floor(N/R);

        E=zeros(1,N_E);
        for n=1:floor((N-L)/R)
            for m=1:L
                E(n)=E(n)+s(m+(n-1)*R)^2*window(m);
            end
        end
```

```
        M=zeros(1,N_E);
        for n=1:floor((N-L)/R)
            for m=1:L
                M(n)=M(n)+abs(s(m+(n-1)*R))*window(m);
            end
        end

        sign=zeros(1,N);
        for i=1:N
            if s(i)>=0
                sign(i)=1;
            else
                sign(i)=-1;
            end
        end

        Z=zeros(1,N_E);
        for n=1:floor((N-L)/R)
            for m=2:L
                Z(n)=Z(n)+(abs(sign(m+R*(n-1))-sign(m-1+R*(n-
1))))*window(m)/(2*L);
            end
        end
end
```

The window function to calculate short-time zero-crossing rate is `Hamming` window.

Then design a lowpass filter which can preserve the low frequency band until about $f = 0.1F_s$ and remove the band from $f = 0.2 * F_s$ to $f = 0.5 * F_s$ to smooth the short-time zero-crossing rate contour.

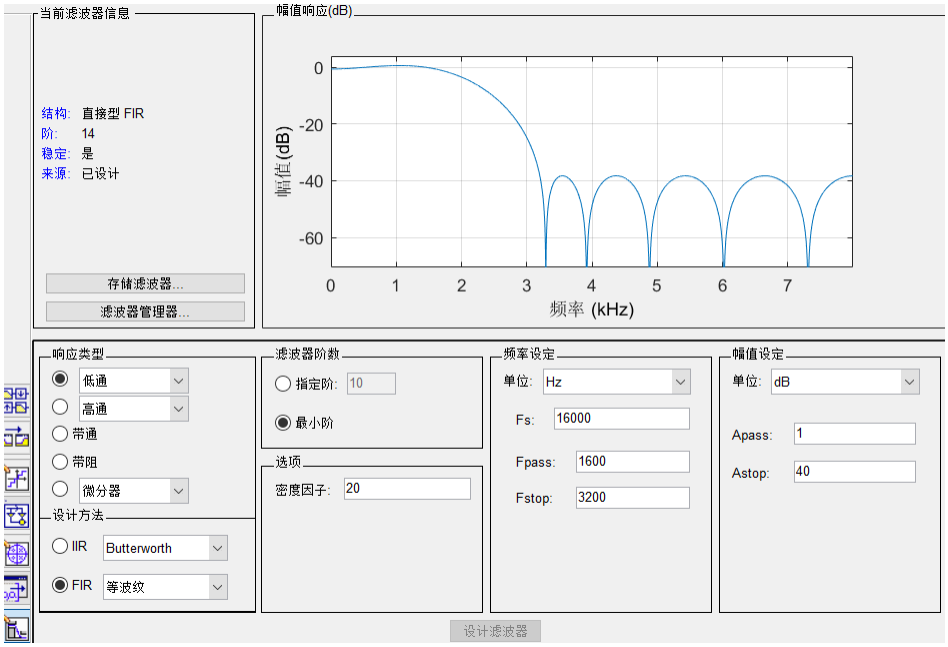


Fig.1 design of lowpass filter

After that, median-smooth the original zero-crossing rate contour using a combination of a running median of seven samples followed by a running median of five samples. The formula of median smoother with L points is:

$$y[n] = M_L[x[n]] = med_{m=0}^{L-1} x[n-m] \tag{1}$$

The function of median smoother is as below:

```
function y=MedianSmoother(x,n)
    y=x;
    for i =(n-1)/2+1:length(x)-(n-1)/2
        sub=x(i-(n-1)/2:i+(n-1)/2);
        y(i)=median(sub);
    end
end
```

Finally, use a combination smoother filter to smoother the original zero-crossing rate contour. The composition of a combination smoothing filter is as below:

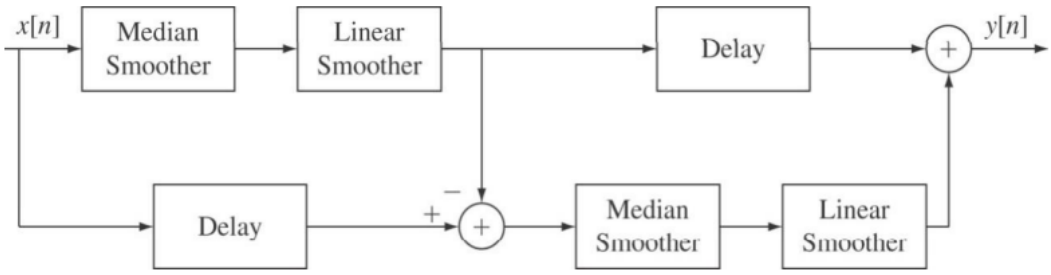


Fig.2 composition of a combination smoothing filter

The total code is as below.

```
[s,fs]=audioread('test_16k.wav');
N=length(s);
L=0.01*fs;
R=0.005*fs;
window=hamming(L);

%original zero-crossing rate contour
[E,M,Z]=EMZ(s,L,R,window');

%lowpass filter
Hd=lowpass_filter_problem1;
Z_lowpass=filter(Hd,Z);

%median filter
Z_median_7=MedianSmoother(Z,7);
Z_median_5=MedianSmoother(Z_median_7,5);

%combined
%delay=(length(Z)-1)/2;
delay=0;
Z1=filter(Hd,Z_median_5);

Z2=zeros(1,delay+length(Z));
Z2(delay+1:delay+length(Z))=Z;

Z11=zeros(1,delay+length(Z));
```

```
Z11(1:length(Z))=Z1;
Z3=Z2-Z11;

Z4=zeros(1,delay+length(Z));
Z4(delay+1:delay+length(Z))=Z1;

Z3_median_7=MedianSmoother(Z3,7);
Z3_median_5=MedianSmoother(Z3_median_7,5);
Z5=filter(Hd,Z3_median_5);

y=Z4+Z5;

figure
subplot(4,1,1);
plot(0:1/fs*R:(N/R-1)/fs*R,Z);
title("original zero-crossing rate contour");
xlabel('time(s)');
ylabel('zero-crossing rate');

subplot(4,1,2);
plot(0:1/fs*R:(N/R-1)/fs*R,Z_lowpass);
title("zero-crossing rate contour smoothed by lowpass filter");
xlabel('time(s)');
ylabel('zero-crossing rate');

subplot(4,1,3);
plot(0:1/fs*R:(N/R-1)/fs*R,Z_median_5);
title("zero-crossing rate contour smoothed by median filter");
xlabel('time(s)');
ylabel('zero-crossing rate');

subplot(4,1,4);
plot(0:N/fs/(length(y)-1):N/fs,y);
title("zero-crossing rate contour smoothed by combined filter");
xlabel('time(s)');
ylabel('zero-crossing rate');
axis([0 1.4 -0.2 0.6]);
```

```
figure
plot(0:1/fs*R:(N/R-1)/fs*R,Z); hold on;
plot(0:1/fs*R:(N/R-1)/fs*R,Z_lowpass);hold on;
plot(0:1/fs*R:(N/R-1)/fs*R,Z_median_5);hold on;
plot(0:N/fs/(length(y)-1):N/fs,y);
legend('original','lowpass filter','median filter','combined filter');
xlabel('time(s)');
ylabel('zero-crossing rate');
```

The result is as below.

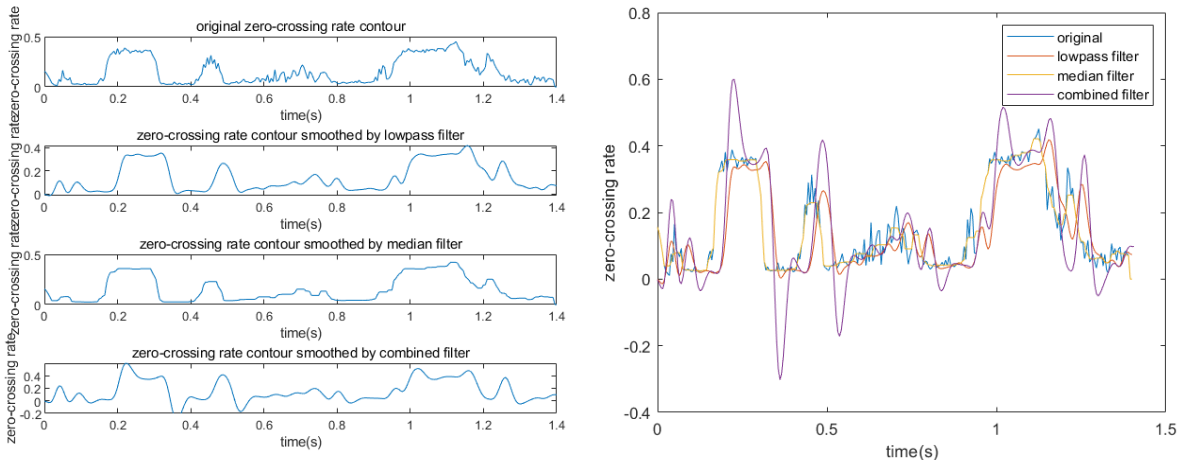


Fig.3 zero-crossing rate of problem1

From the above picture, what can be known is that the original zero-crossing rate is the sharpest with many burrs. Compared to the original zero-crossing rate, the zero-crossing rate smoothed by the lowpass filter has a time delay and well remove this excessive details, which looks more smoother. The median filter makes the change of zero-crossing rate less obvious and removes many unnecessary details. However, at the beginning and ending of the signal, the median filter is characterized by boundary, which leads to the retention of high-frequency components. The combination smoothing filter amplifies the difference between peaks and troughs and has a time delay.

In a conclusion, median filter is better than linear filter in skip edge processing. The reason is that median filter retains the feature of zero-crossing rate and eliminate some unreasonable fluctuations in linear filter.

2.2 Problem2

The requirement of problem2 is to write an autocorrelation-based pitch detector using the modified autocorrelation function and compare the results using both the original speech file and those obtained from a bandpass filtered version of the speech file.

Firstly, convert the sampling rate to standard value of $f_{sout} = 10000$ Hz for this exercise using function `resample`.

Then, a bandpass filter is designed to eliminate DC offset, 60 Hz hum, and high frequency(above 1000 Hz) signal using the design parameters: stopband from 0 to 80 Hz, transition band from 80 to 150 Hz, passband from 150 to 900 Hz, transition band from 900 to 970 Hz, stopband from 970 to 5000 Hz and filter length of $n = 301$ samples.

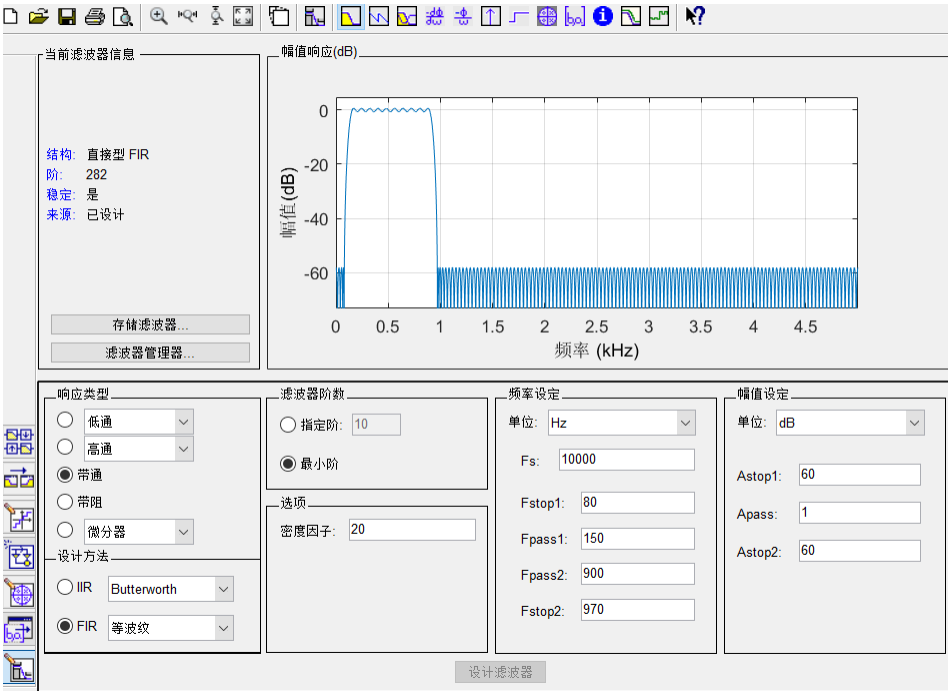


Fig.3 design of bandpass filter

After that, block the signal into frame of length $L = 400$ samples with frame shift of $R = 100$ samples and compute the frame-by-frame modified correlation between the frames of signal. The frames of signal specified as:

$$\begin{aligned} s_1[n] &= [s[n], s[n+1], \ldots, s[n+L-1]] \\ s_2[n] &= [s[n], s[n+1], \ldots, s[n+L+pdhigh-1]] \end{aligned} \tag{2}$$

where n is the starting sample of the current frame, and $pdhigh$ is the longest anticipated pitch period and is specified as:

$$pdhigh = \begin{cases} f_{sout}/75 & males \\ f_{sout}/150 & females \end{cases} \tag{3}$$

The $pdlow$ is specified as:

$$pdlow = \begin{cases} f_{sout}/200 & males \\ f_{sout}/300 & females \end{cases} \tag{4}$$

And according to the gender, search from $pdlow$ to $pdhigh$ to find the maximum of the modified autocorrelation function along with the value of the modified autocorrelation at the maximum. The former is the putative pitch period estimate for the current frame and the latter is the confidence score.

Then convert the confidence score to a log confidence, set a threshold at 0.75 of the maximum value of the log confidence score and set the pitch period to zero for all frames whose log confidence scores fell below the threshold.

Finally, use a 5-point median smoother to smooth the pitch period scores as well as the confidence score.

The code of function `PitchDetector_Autocorrelation` is as below.

```
function
[pitch_period,pitch_period_medianfilter,score_log,score_log_medianfilter]=Pitch
hDetector_Autocorrelation(s,fsout,gender)
    if (gender=="male")
        pdhigh=floor(fsout/75);%向下取整
        pdlow=ceil(fsout/200);%向上取整
    else
        pdhigh=floor(fsout/150);
        pdlow=ceil(fsout/300);
    end
    N=length(s);
    L=400;
    R=100;
    h=1:floor((N-L)/R);
    au=zeros(length(h),(L+pdhigh)*2-1);
```

```
    pitch_period=zeros(1,length(h));
    score=zeros(1,length(h));
    for n=1:floor((N-L)/R)
        s1=s((n-1)*R+1:(n-1)*R+L);
        s2=s((n-1)*R+1:(n-1)*R+L+pdhigh);
        au(n,:)=xcorr(s1,s2);
        temp=au(n,:);
        med=temp(round((length(temp)-1)/2):end);
        [score(n),maxvalue]=max(med(pdlow:pdhigh));
        pitch_period(n)=maxvalue+pdlow-1;
    end
    score_log=log10(score);
    threshold=max(score_log)*0.75;
    for n=1:floor((N-L)/R)
        if score_log(n)<threshold
            pitch_period(n)=0;
        end
    end
    pitch_period_medianfilter=MedianSmoother(pitch_period,5);
    score_log_medianfilter=MedianSmoother(score_log,5);



end
```

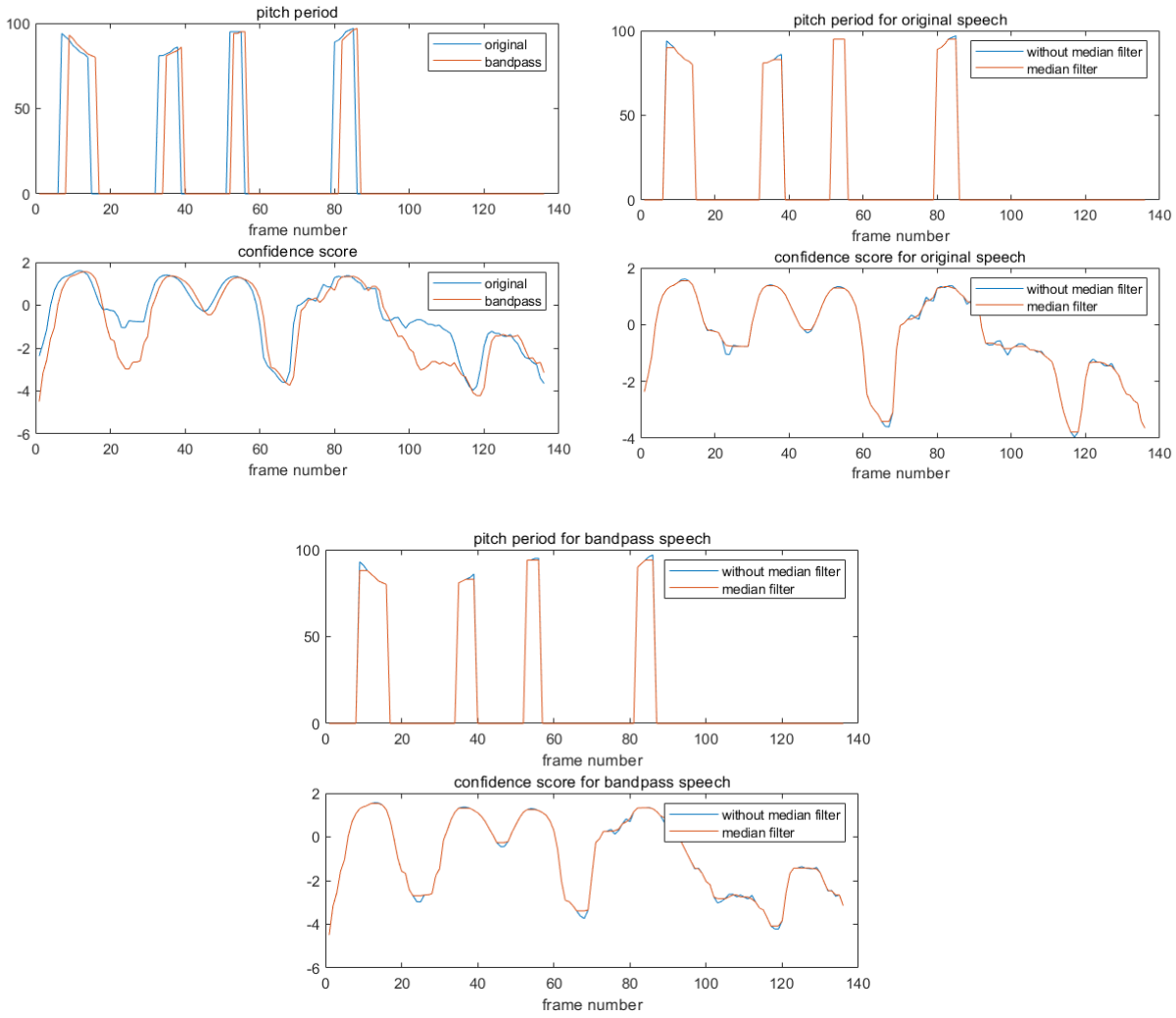
The total code is as below.

```
[s,fs]=audioread('test_16k.wav');
fsout=10000;

%resample
s_resample=resample(s,fsout,fs);

%bandpass
Hd=bandpassfilter_problem2;
s_resample_bandpass=filter(Hd,s_resample);
```

```
%save speech file
audiowrite('original resampled speech.wav',s_resample,fsout);
audiowrite('original resampled bandpassed
speech.wav',s_resample_bandpass,fsout);

%operation
[pitch_period_original,pitch_period_medianfilter_original,score_log_original,s
core_log_medianfilter_original]=PicthDetector_Autocorrelation(s_resample,fsout
,"male");
[pitch_period_bandpass,pitch_period_medianfilter_bandpass,score_log_bandpass,s
core_log_medianfilter_bandpass]=PicthDetector_Autocorrelation(s_resample_bandp
ass,fsout,"male");
save('pitch_period_medianfilter_original.mat','pitch_period_medianfilter_origi
nal');
save('score_log_medianfilter_original.mat','score_log_medianfilter_original');
save('pitch_period_medianfilter_bandpass.mat','pitch_period_medianfilter_bandp
ass');
save('score_log_medianfilter_bandpass.mat','score_log_medianfilter_bandpass');

figure
subplot(2,1,1);
plot(pitch_period_original);
hold on;
plot(pitch_period_bandpass);
title('pitch period');
xlabel('frame number');
legend('original','bandpass');
subplot(2,1,2);
plot(score_log_original);
hold on;
plot(score_log_bandpass);
title('confidence score');
xlabel('frame number');
legend('original','bandpass');

figure;
subplot(2,1,1);
```

```
plot(pitch_period_original);
hold on;
plot(pitch_period_medianfilter_original);
title('pitch period for original speech');
xlabel('frame number');
legend('without median filter','median filter');
subplot(2,1,2);
plot(score_log_original);
hold on;
plot(score_log_medianfilter_original);
title('confidence score for original speech');
xlabel('frame number');
legend('without median filter','median filter');

figure;
subplot(2,1,1);
plot(pitch_period_bandpass);
hold on;
plot(pitch_period_medianfilter_bandpass);
title('pitch period for bandpass speech');
xlabel('frame number');
legend('without median filter','median filter');
subplot(2,1,2);
plot(score_log_bandpass);
hold on;
plot(score_log_medianfilter_bandpass);
title('confidence score for bandpass speech');
xlabel('frame number');
legend('without median filter','median filter');
```

The result is as below.



According to the above pictures, after the original speech file is filtered by the bandpass filter, the pitch period of the bandpass filtered speech has a small delay, compared to that of the original speech. What's more, the log confidence score of the bandpass filtered speech is significantly reduced in the places where the log confidence should be small, which indicates the accuracy of the result is improved by the bandpass filter. Besides, it is obvious that the fluctuations in some local part of the pitch period contour of the original speech are removed after being filtered by 5-point median filter. The log confidence score of the original speech filtered by 5-point median filter is more smoother. So as the bandpass filtered speech. Because of these reasons, using the bandpass filtered speech file processing works best.

2.3 problm3

The requirement of problem3 is to program a pitch detector based on the real cepstrum of a speech signal. The real cepstrum is defined as the inverse FFT of the log magnitude spectrum of the signal, and the pitch period for voiced speech sections is found as the location of the peak of the cepstrum over the range of pitch periods appropriate for the gender of the speaker.

The signal processing parameters define the size of the FFT used to measure the spectrum and the cepstrum as $nfft = 4000$ to minimize aliasing. Also, a threshold an the ratio of the primary cepstral peak to the secondary cepstral peak is specified as `pthr1=4` and is used to define a region of certainty about pitch period estimates. The pitch period range for searching the cepstrum for the pitch peak is specified as the range $nlow \leq n \leq nhigh$, where the low and high of the pitch period range is specified as:

$$mlow = \begin{cases} 40 & males \\ 28 & females \end{cases} \tag{5}$$

$$nhigh = \begin{cases} 167 & males \\ 67 & females \end{cases} \tag{6}$$

To find primary and secondary cepatral peaks, the first step is to locate the maximum of the cepatrum over the specified range(p_1), and record the quefrency at which the maximum occurs(pd_1). Then, zero the cepstrum over a range of ± 4 quefrencies around the maximum location found in the previous step, thereby eliminating the possibility that the secondary cepstral maximum will essentially be the same as the primary maximum. After that, locate the secondary maximum of the processed cepstrum,recording its quefrency(pd_2) and its value(p_2).

The next step is to define reliable regions of voices speech for those frames whose ratio of primary cepstral maximum to secondary cepstral maximum exceeds the threshold. Note that we should detect the adjacent frames whose primary or secondary pitch periods are within 10% of the pitch period at the boundary frames. The process is continued until all reliable regions have been extended to include neighboring pitch estimates.

The final step in the signal processing is median smoothing of the pitch period contour using 5-point median smoother.

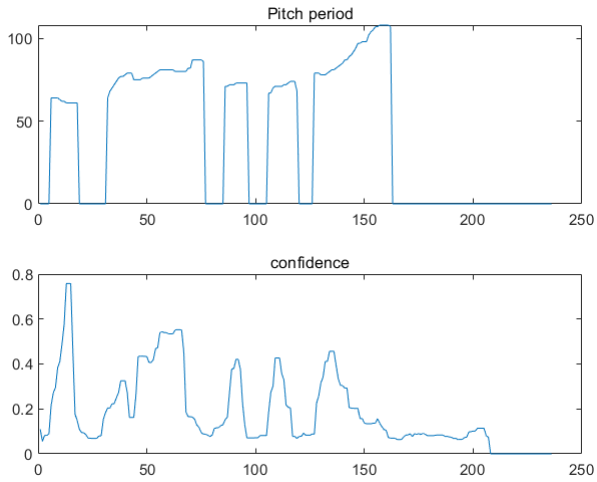
The code of function `PitchDetector_Cepstrum` is as below:

```
function [p1_median,pd1_median]=PictnDetector_Cepstrum(s,fs,gender)
    if (gender=="male")
        nlow=40;
```

```
        nhigh=167;
    else
        nlow=28;
        nhigh=67;
    end

    nfft=4000;
    N=length(s);
    L=400;R=100;

    pthr1=4;
    h=1:floor((N-L)/R);
    reliable=zeros(1,length(h));

    %real cepstrum
    rceps=zeros(length(h),nfft);
    p1=zeros(1,length(h));
    p2=zeros(1,length(h));
    pd1=zeros(1,length(h));
    pd2=zeros(1,length(h));

    for n=1:floor((N-L)/R)
        X=fft(s((n-1)*R+1:(n-1)*R+L).*hamming(L),nfft);
%         theta=unwrap(angle(X));
%         X_log=complex(log(abs(X)),theta);
%
%         rceps(n,:)=fftshift(ifft(X_log,nfft));
        rceps(n,:)=fftshift(ifft(log(abs(X)),nfft));
        temp=rceps(n,:);
        temp(isnan(temp))=0;
        temp=temp(nfft/2+1:end);
        [p1(n),pd1(n)]=max(temp(nlow:nhigh));
        pd1(n)=pd1(n)+nlow-1;
        temp(pd1(n)-4:pd1(n)+4)=0;
        [p2(n),pd2(n)]=max(temp(nlow:nhigh));
        pd2(n)=pd2(n)+nlow-1;
        if (p1(n)/p2(n)>=pthr1)
```

```
            reliable(n)=1;
        end
    end

    for n=1:floor((N-L)/R)
        if (reliable(n)==1)
            i=1;j=1;
            while((n+i<length(pd1))&&((abs(pd1(n+i)-pd1(n+i-1))<=0.1*pd1(n+i-
1))||(abs(pd2(n+i)-pd2(n+i-1))<=0.1*pd2(n+i-1))))
                reliable(n+i)=1;
                i=i+1;
            end
            while((n-j>0)&&((abs(pd1(n-j)-pd1(n-j+1))<=0.1*pd1(n-j+1))||
(abs(pd2(n-j)-pd2(n-j+1))<=0.1*pd2(n-j+1))))
                reliable(n-j)=1;
                j=j+1;
            end
        end
    end

    for n=1:floor((N-L)/R)
        if(reliable(n)==0)
            pd1(n)=0;
        end
    end

    p1_median=MedianSmoother(p1,5);
    pd1_median=MedianSmoother(pd1,5);
end
```

The result is as below.



According to the above figures, it is easily found that the confidence score is high at the places where the predicted pitch period is. However, it is obvious that blocking and incoherence appear after being smoothed by median filter.

3.conclusion

The main harvest of this experiment is to master the processing of median smoothing on speech parameters estimation and the commonly-used algorithms on pitch estimation based on autocorrelation and cepstrum. Median filter is better than linear filter in skip edge processing because median filter retains the feature of zero-crossing rate and eliminate some unreasonable fluctuations in linear filter. What's more, the effect using a bandpass filtered version of the speech file is better than that using the original speech file in terms of detecting the pitch period contour and confidence score based on autocorrelation.