

DSP Lab 4  
104061171 紀伯翰

**1a. Specification**

Detect the R wave from your recorded ECG signals.

**1b. Implement**

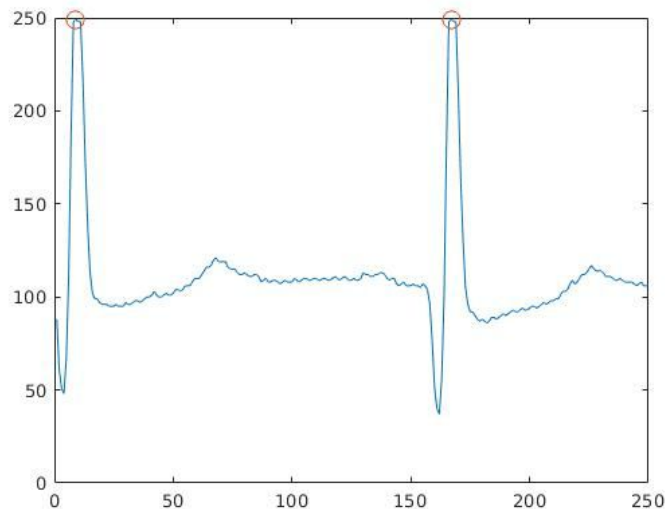


Figure 1.a

**1c. Discussion**

作法如下，先通過moving average filter將60hz的雜訊濾掉，再通過high pass filter  $[1, -1]$ ，可以粗略知道峰值就在通過 $[1, -1]$ 濾波器之後的波形，值由負轉正的那個點，以老師上課的作法是希望直接平方，但我認為平方後，負端也會被翻成正值，對於找尋真正的峰值實在是有所不易，所以我先把訊號內所有負的值變成0，做歸一化，方容易我之後取門檻值，再做平方，如此一來，在通過設定的threshold，方可以找到我要的peak點，對於訊號處理上就變得容易。實作過程中，我使用conv的函式，所以我有處理group delay的問題，根據conv的形式去shift訊號，讓訊號對到我所希望找到的峰值。

**2a. Specification**

Find the R-peaks in MIT-BIH database.

**2b. implement**

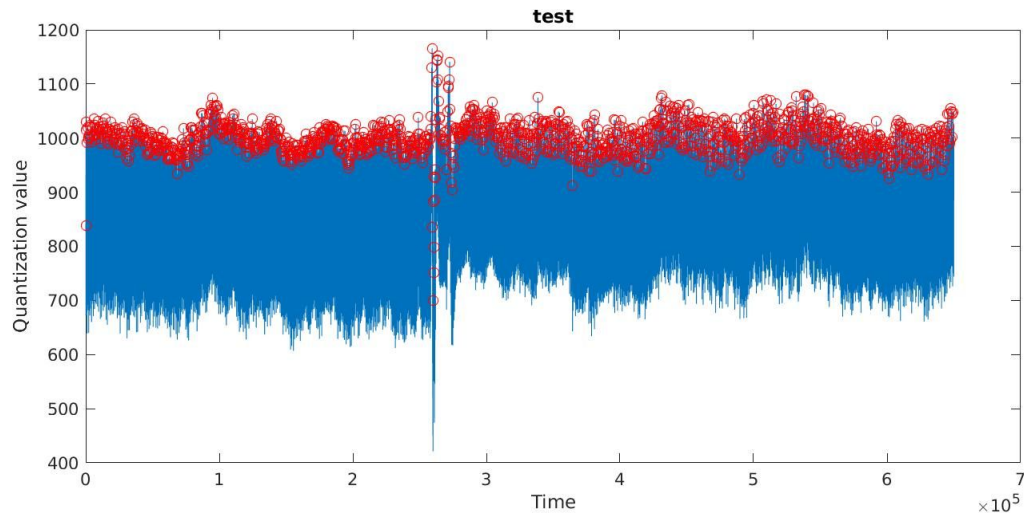


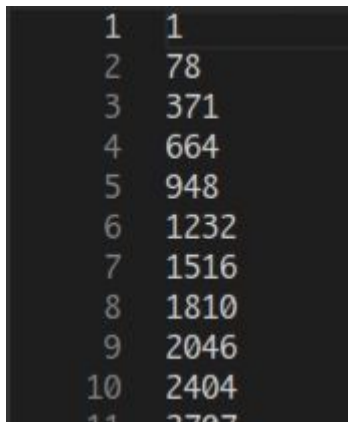
Fig 117m.mat

		TP	FN	FP	Precision
1					
2	100.m_first_row:	2271	1	1	99.96%
3	103.m_first_row:	2079	5	0	100.0%
4	112.m_first_row:	2539	0	0	100.0%
5	117.m_first_row:	1508	26	27	98.24%
6	122.m_first_row:	2476	0	0	100.0%
7					
8	107.m_first_row:	2080	57	47	97.79%
9	114.m_first_row:	935	943	514	64.53%
10	119.m_first_row:	1986	1	1	99.95%
11	205.m_first_row:	2583	73	63	97.62%
12	219.m_first_row:	2095	59	0	100.0%
13					
14	108.m_first_row:	960	802	565	62.95%
15	203.m_first_row:	1092	1877	273	80.0%
16	210.m_first_row:	2344	305	33	98.61%
17	222.m_first_row:	2252	231	3	99.87%
18	230.m_first_row:	2255	1	3	99.87%

## 2c. Discussion

Precision, recall的概念在專題時就已經碰過，比較好奇的是recall的值竟然不用紀錄？！，比較好奇這一點，precision指的是你選的資料內有多少是正確的，recall是所有正確的資料內，你選中多少比例，而TP,FP個別指的是你選的資料內正確的與錯誤的資料數，而FN指的是正確的資料內你有多少沒選到，首先,我會先記住每一個rpeak的index，並且在之後輸出成txt檔案，再透過python處理去比對資料。在比對完資料後，再將那些資料紀錄在我的result.txt內，做成表格輸出如上。而下圖分別為matlab輸出資料code碼與輸出的資料格式

```
fileID = fopen('Desktop/4up/DSP_LAB/hw4/predict/117_1.txt','w');
fprintf(fileID,'%d\n',rpeak);
```



1	1
2	78
3	371
4	664
5	948
6	1232
7	1516
8	1810
9	2046
10	2404
11	2707

再者，我使用自己寫的python code去做資料比對，首先會先把每一筆資料的時間load進變數內，誤差範圍我選擇的是 $\pm 10$ 點，大概為0.05秒左右，然後透過index先把index轉換成時間，在誤差範圍之內的差距，那個點就視為選中，然後是以一個範圍一個範圍做資料比對的。

### 3a.Specification

Implement the pre-processing of the ECG signals (Lab 3) and R-peak detection in real time ,and display the processed ECG signals and the R-peaks in real time.

### 3b.Implement

附壓縮檔內有

### 3c Discussion

後來real\_time的寫法，我將normalize寫成一個function，high\_pass也寫成一個function，moving\_average也寫成一個function，找尋peak也寫成一個function，模組化方便自己管理，每紀錄30個點，我就會去find peak尋找rpeak 點做標注index的動作，並透過index的vector，計算出heart\_beat\_rate。

### Extra Question

#### 1a.Specification

Please modularize your signal processing flow. That is, please make each block as a function and then perform function calls.

In your report, please specify the block diagram of your signal processing, and the corresponding function of each block.

If you're implementing real-time R-peak detection, please also justify your approach. Note that you can implement your signal processing modules in PC or in Arduino. Note that you have to hand in your codes along with your report, and please provide the average of the elapsed time for 100 loops of your signal processing via MATLAB function "tic" and "toc" (see ShortIntro2MatlabProfile.pdf)

#### 1b.Implement:

Function listing		
Color highlight code according to <input type="text" value="time"/>		
time	calls	line
0.007	1	2 load('Desktop/4up/DSP_LAB/hw4/MIT_database/easy/117m.mat');
		3
		4
< 0.001	1	5 disbuff = val(1,:);
< 0.001	1	6 time=[1:1:1000];
		7
0.029	1	8 h_plot=plot(nan,nan);
0.006	1	9 hold on;
0.002	1	10 d_plot=plot(nan,nan,'ro');
		11
		12
		13
0.004	1	14 out = low_pass(disbuff);
0.003	1	15 output= high_pass(out);
0.005	1	16 norm = normalize(output);
< 0.001	1	17 norm2 = norm.^ 2;
0.077	1	18 rpeak = findrpeak(norm2,disbuff);
0.002	1	19 set(h_plot,'xdata',((1:length(disbuff))-1),'ydata',disbuff);
		20
0.009	1	21 title('test');
0.003	1	22 xlabel('Time');
0.003	1	23 ylabel('Quantization value');
< 0.001	1	24 if length(rpeak)>=1
< 0.001	1	25 set(d_plot,'xdata',(rpeak-1),'ydata',disbuff(rpeak));
		26 %drawnow;
< 0.001	1	27 end

Figure 1.a

Function listing		
Color highlight code according to <input type="text" value="time"/>		
time	calls	line
		57 function rpeak = findrpeak(data,disbuff)
< 0.001	1	58 temp=[1];
< 0.001	1	59 i=1;
< 0.001	1	60 while i<(length(disbuff)-90)
0.022	511759	61 if data(i)>0.1
0.006	1535	62 [value ,index]=max(disbuff(i:i+90));
0.003	1535	63 temp=[temp index+i-1];
< 0.001	1535	64 i=i+90;
< 0.001	1535	65 end
0.021	511759	66 i = i+1;
0.023	511759	67 end
< 0.001	1	68 rpeak=temp;
< 0.001	1	69 end

Figure 1.b

My Module function:

```
function output=low_pass(data)

n=8;
one_filter = ones(n,1);
temp=conv(data,one_filter/n,'same');
output=temp(n-1:(length(temp)-(n-1)));

end
```

Figure 1.c

```
function output=high_pass(data)
    h1 = [1 -1];
    output =conv(data,h1);
    output = output(2:(length(output)-1));
end
```

Figure 1.d

```
function norm = normalize(data)
    data(data<0)=0;
    norm =data(:) ./ max(data);
end
```

Figure 1.e

```
function rpeak = findrpeak(data,disbuff)
    temp=[1];
    i=1;
    while i<(length(disbuff)-90)
        if data(i)>0.1
            [value ,index]=max(disbuff(i:i+90));
            temp=[temp index+i-1];
            i=i+90;
        end
        i = i+1;
    end
    rpeak=temp;
end
```

Figure 1.f

### 1c Discussion

我這邊以mit database為例，我發現我花最多時間的是在findrpeak這個我自己定義的function內，而我調查這個function發現其中是因為我的while loop要跑得點太多而產生的bottle neck，不過這也在所難免，因為我將threshold放寬的原因，才導致他要跑得點變多了。

### 2a.Specification

Please draw a table in your report. The first column is the name of the data set, the 2nd column is TP, the 3rd column is FN, and the 4th column is FP. Please justify how you estimate your TP, FN, and FP and the precision when matching your results with the ground truth.

### 2b Implement

		TP	FN	FP	Precision
1					
2	100.m_first_row:	2271	1	1	99.96%
3	103.m_first_row:	2079	5	0	100.0%
4	112.m_first_row:	2539	0	0	100.0%
5	117.m_first_row:	1508	26	27	98.24%
6	122.m_first_row:	2476	0	0	100.0%
7					
8	107.m_first_row:	2080	57	47	97.79%
9	114.m_first_row:	935	943	514	64.53%
10	119.m_first_row:	1986	1	1	99.95%
11	205.m_first_row:	2583	73	63	97.62%
12	219.m_first_row:	2095	59	0	100.0%
13					
14	108.m_first_row:	960	802	565	62.95%
15	203.m_first_row:	1092	1877	273	80.0%
16	210.m_first_row:	2344	305	33	98.61%
17	222.m_first_row:	2252	231	3	99.87%
18	230.m_first_row:	2255	1	3	99.87%

**FIG 2A**

As shown above

## 2c.Discussion

前面第二小題

## 3a.Specification

Note that you have to take care "the delay" (from linear phase FIR filter) introduced by your signal processing (e.g., FIR filtering) in order to obtain the same R-peak time as provided in the \*.txt file when you find R-peak

## 3b.implement

```
[
function output=low_pass(data)

    n=8;
    one_filter = ones(n,1);
    temp=conv(data,one_filter/n,'same');
    output=temp(n-1:(length(temp)-(n-1)));
end
```

```
function output=high_pass(data)
```

```
    h1 = [1 -1];
    output =conv(data,h1);
    output = output(2:(length(output)-1));
```



End

]

(以上為function code)(以下為main的一小部份)

```
if length(rpeak)>=1
    set(d_plot,'xdata',(rpeak-1),'ydata',disbuff(rpeak));
    %drawnow;
end
```

### 3c.Discussion

由上可以看到，因為有delay的關係，所以必須微調將點的位置微調回來

### 4a Specification

Real time filtering can be simply implemented according to the "structures" of the discrete-time FIR or IIR system (see at 5:53).

In your report, please "elaborate" how you implement your real time filtering and provide your codes in your report and explain your codes.

(from this you should know the "structure" of a system (or a transfer function,  $H(z) = Y(z)/X(z)$ ,  $Y(z)$ : output,  $X(z)$ : input) can be used for real-time implementation of a digital signal processing system)

### 4b.Implement

```
%%%%%setup%%%%%%%%
```

```
clear all;
fclose('all');
serialobj=instrfind;
if ~isempty(serialobj)
    delete(serialobj)
end
clc;clear all;close all;
s1 = serial('/dev/ttyACM0'); %define serial port
s1.BaudRate=115200; %define baud rate
```

```
disbuff=nan(1,1000);
```

```
fopen(s1);
clear data;
N_point = 1000;
fs=250; %sample rate
```

```
time=[1:1:1000];
```

Figure

```
h_plot=plot(nan,nan);
```

```
hold on;
```

```
d_plot=plot(nan,nan,'ro');hold off;
```

```
tic
```

```
%%%%%%%%set up%%%%%%%%
```

```
i=1;
```

```
%%%%%%%%infinite loop%%%%%%%%
```

```
while 1
```

```
    data=fscanf(s1);%read sensor
```

```
    y(i) = str2double(data);
```

```
    if i<=1000
```

```
        disbuff(i)=y(i);
```

```
    else
```

```
        disbuff=[disbuff(2:end) y(i)];
```

```
    end
```

```
    if i>1 & mod(i,30) == 0    %update every 30 point
```

```
        out = low_pass(disbuff); %lowpass
```

```
        output= high_pass(out);    %high pass
```

```
        norm = normalize(output);    %normalize
```

```
        norm2 = norm.^ 2;            %square
```

```
        rpeak = findrpeak(norm2,disbuff); %rpeak index get
```

```
        heartbeatrte = hbr(rpeak);    %heart beat rate
```

```
        set(h_plot,'xdata',time,'ydata',disbuff) %draw
```

```
        title(['Heart Beat rate is ',num2str(heartbeatrte),'Hz']);
```

```
        xlabel('Time');
```

```
        ylabel('Quantization value');
```

```
        if length(rpeak)>=1
```

```
            set(d_plot,'xdata',rpeak,'ydata',disbuff(rpeak));
```

```
            drawnow;
```

```
        end
```

```
end
```



```

    i=i+1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%lowpass%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function output=low_pass(data)    %low pass

    n=8;
    one_filter = ones(n,1);
    temp=conv(data,one_filter/n,'same');
    output=temp(n-1:(length(temp)-(n-1)));

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%lowpass%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%highpass%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function output=high_pass(data)    %high pass

    h1 = [1 -1];
    output =conv(data,h1);
    output = output(2:(length(output)-1));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%highpass%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%normalize%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function norm = normalize(data)    %normalize

    data(data<0)=0;
    norm      =data(:) ./ max(data);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%normalize%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%rpeak%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function rpeak = findrpeak(data,disbuff)    %find peak

    temp=[];
    i=1;

    while i <(length(data)-25)

        if data(i)>0.5

```

```

        [value ,index]=max(disbuff(i:i+25));%period=25
        temp=[temp index+i-1];
        i=i+25;
    end

    i=i+1;

end

rpeak=temp;

End
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%peak%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% heart-beat calculate %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function output = hbr(rpeakvector) %heart beat rate

    interval = [-1 1];
    temp=[];

    for i = 2:(length(rpeakvector)-1)

        disp((sum((rpeakvector(i:i+1) .* interval))/250.0))
        temp(i-1)= (1/(sum((rpeakvector(i:i+1) .* interval))/250.0));

    end

    output = mean(temp);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%heart beat calculate%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

#### 4c.Discussion

code中，每三十個點，會找peak點一次，並畫出來，每一個function都有模組化管理寫清楚，在real time把圖秀出來

#### 4.Conclusion

這次作業除了算心跳圖，還有用python做資料的比對，以及學到了觀看程式效率的函式，precision,recall也算是幫我複習到不少觀念與想法。

#### 5.Reference

Wiki

Python

壓縮檔會附上python code做資料比對

