PTC3456 – Processamento de sinais biomédicos

Relatório do trabalho final – Detecção do complexo QRS

Marcelo Monari Baccaro – 8989262

1) Objetivo

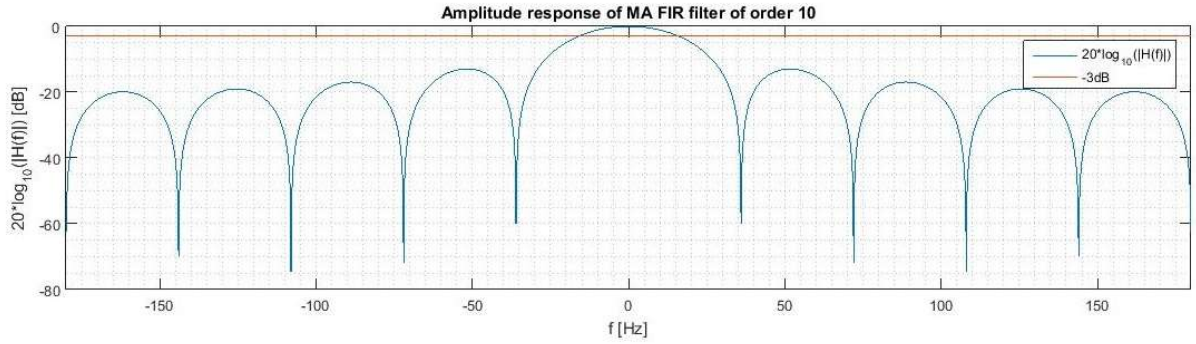
O objetivo deste trabalho é detectar os complexos QRS de um ECG (eletrocardiograma) explorando a técnica de extração de envelope de sinal fornecido pela transformada de Hilbert.

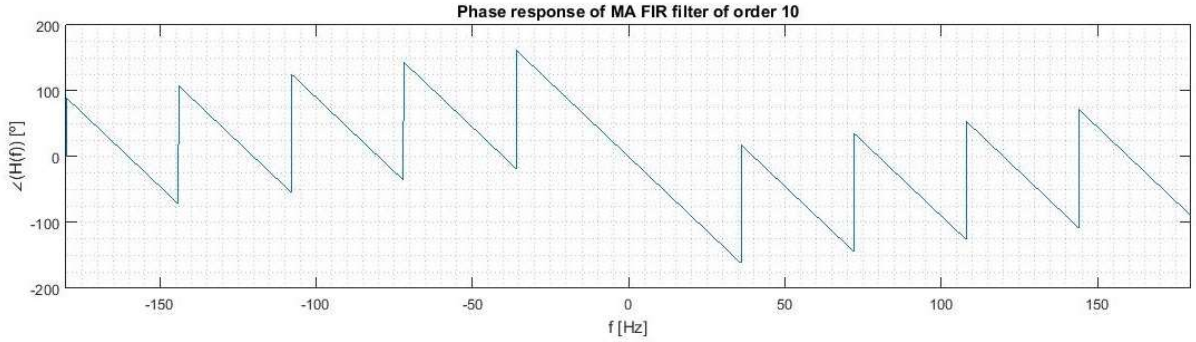
2) Metodologia

Para detectar o complexo QRS, eu executei os seguites passos:

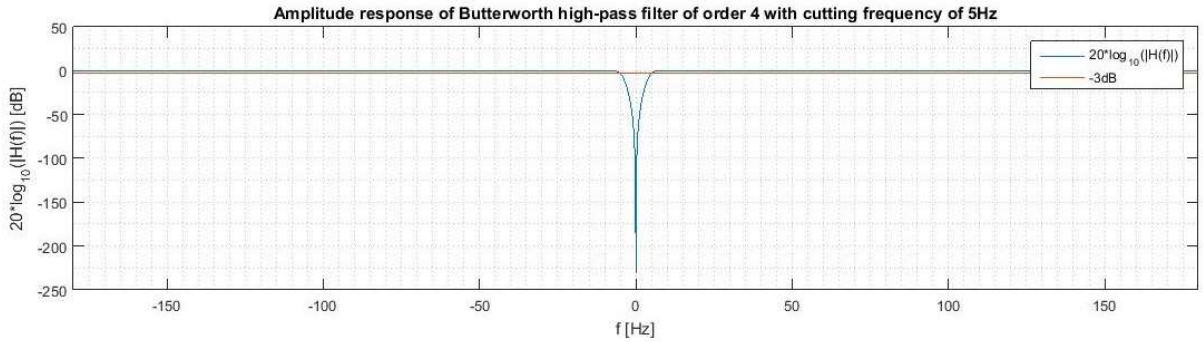
2.1) Gerei os sinais MLII atravez do arquivo '.mat' fornecido pelo PhysioNet.

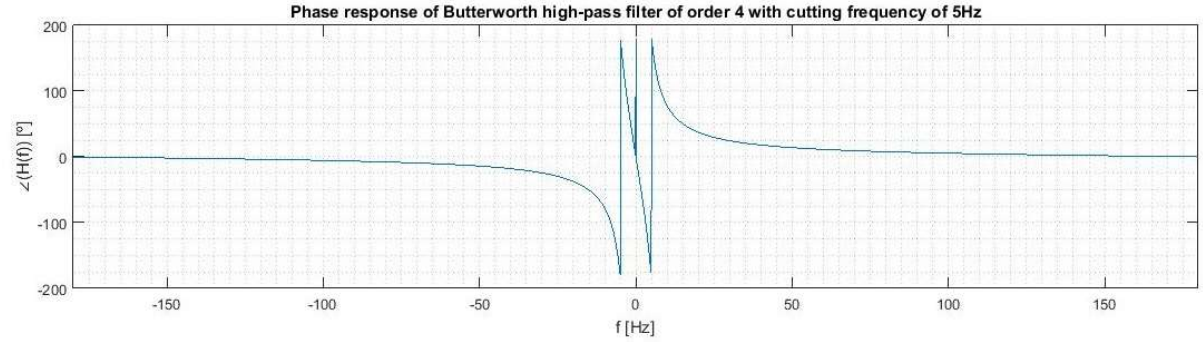
2.2) Usei um filtro passa-baixa do tipo MA (Moving Average) FIR (Finite Impulse Response) de ordem 10. Não usei o filtro de ordem 8, pois a frequência de corte deste é de 20Hz, enquanto que o filtro de ordem 10 tem frequência de corte de 16Hz, o que ajuda na suavização do envelope do sinal.





2.3) Usei um filtro filtro Butterworth passa-alta de ordem 4 e frequência de corte em 5Hz.





2.4) Obtive o envelope do sinal filtrado usando transformada de Hilbert. A transformada de Hilbert retorna o sinal analítico e o módulo deste forma o envelope. [2, Cap 12]

2.5) Desenvolvi um algoritmo simples para detectar picos, em que um valor é classificado como "pico" se for maior que o máximo de "n" valores anteriores e posteriores e também for maior que um threshold.

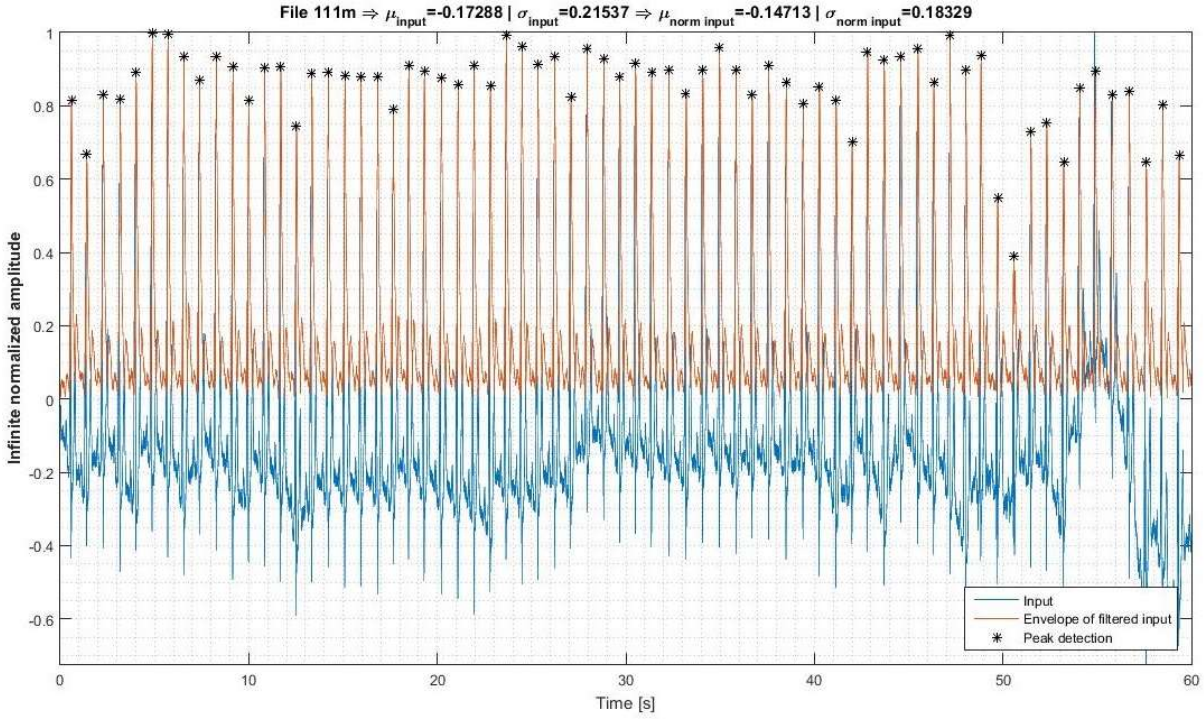
3) Resultados

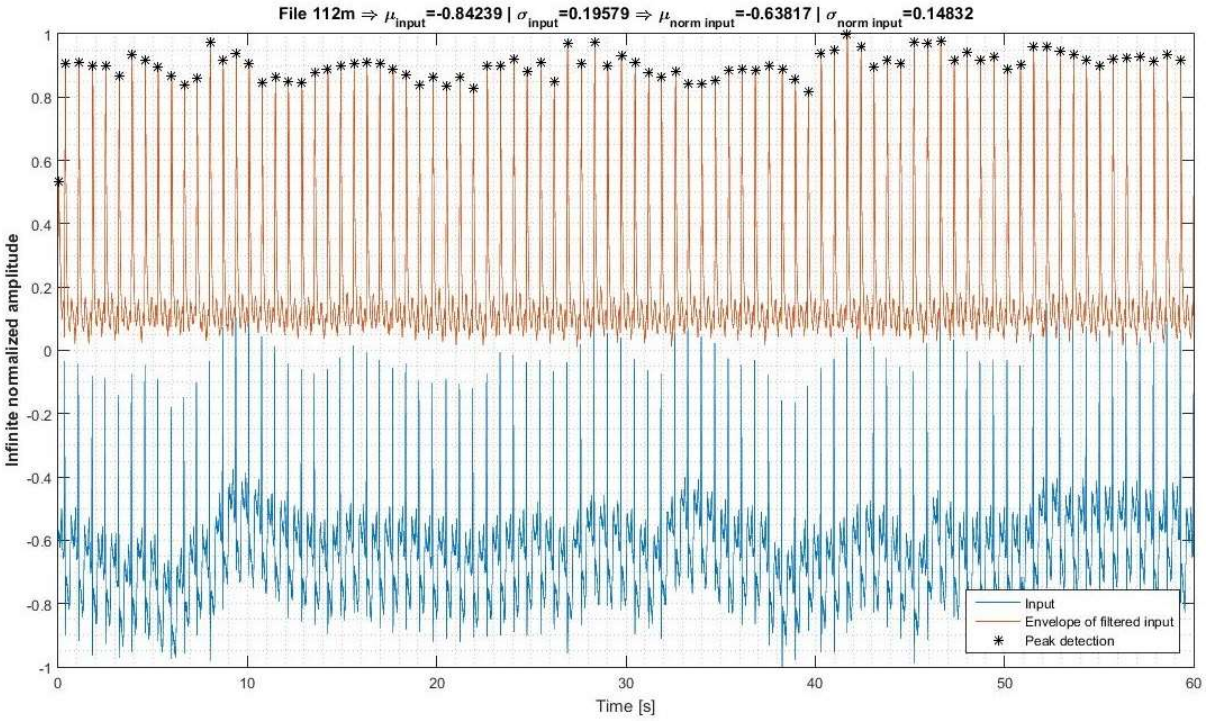
3.1) A tabela abaixo resume os resultados:

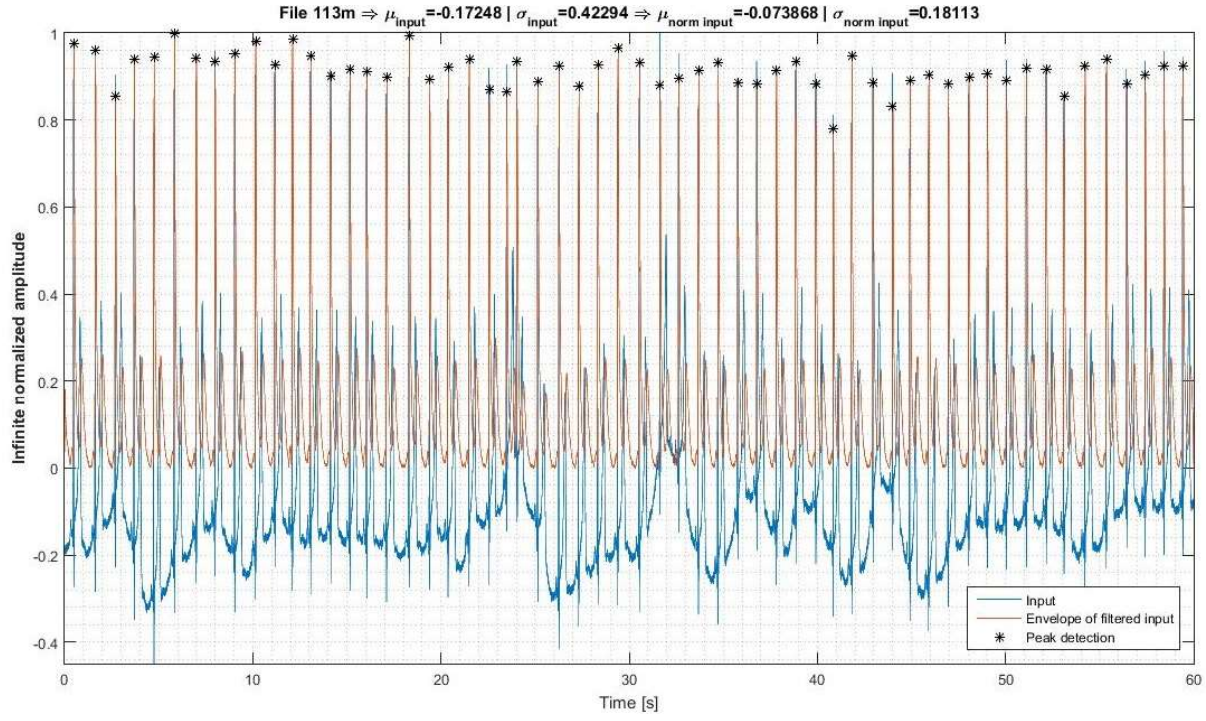
Caso	Média	Desvio padrão	Média norm	DP Norm	Nv	Nd	FP	%FP	FN	%FN	Média IRR	DP IRR
111	-0.1729	0.2154	-0.1471	0.1833	69	69	0	0	0	0	0.8602	0.0434
112	-0.8424	0.1958	-0.6382	0.1483	85	86	1	1.1765	0	0	0.6899	0.0828
113	-0.1725	0.4229	-0.0739	0.1811	57	58	1	1.7544	0	0	1.025	0.1153
114	0.0421	0.2938	0.0183	0.1278	54	54	0	0	0	0	1.1015	0.0497

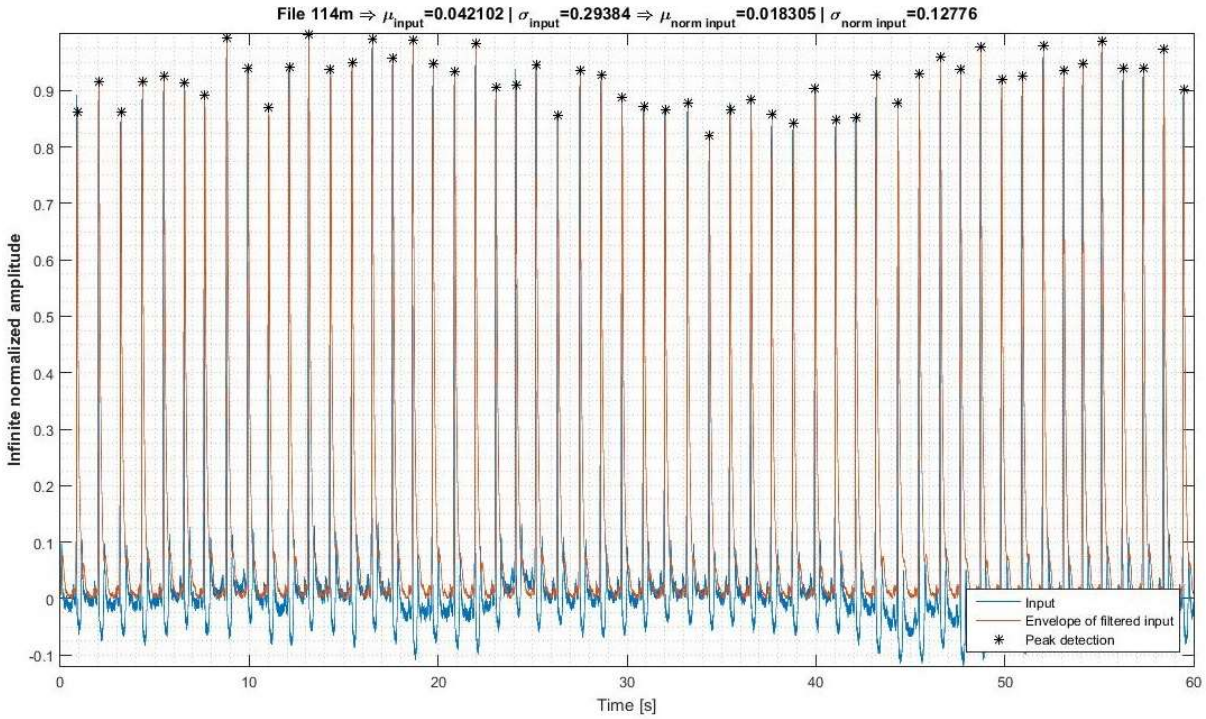
115	-0.5519	0.344	-0.2867	0.1787	63	63	0	0	0	0	0.9481	0.0897
116	-0.9425	0.5972	-0.3611	0.2288	78	79	1	1.2821	0	0	0.7536	0.078
117	-0.7977	0.2287	-0.4144	0.1188	50	51	1	2	0	0	1.1669	0.1925
118	-0.9883	0.4178	-0.3787	0.1601	72	72	0	0	0	0	0.8211	0.0764
119	-0.85	0.5906	-0.3366	0.2339	46	65	19	41.3043	0	0	0.9116	0.2407
Média	N	N	N	N	N	N	N	5.2797	N	0	N	N
DP	N	N	N	N	N	N	N	13.5336	N	0	N	N

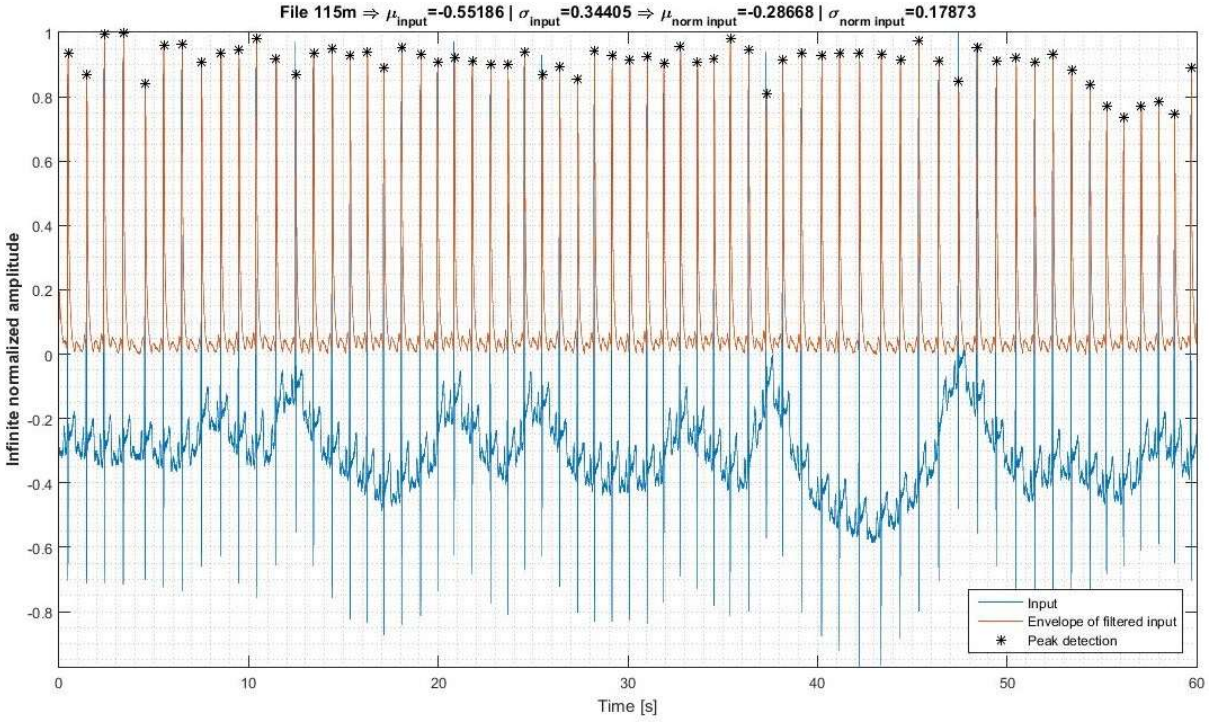
3.2) Mas também é interessante ver cada sinal:

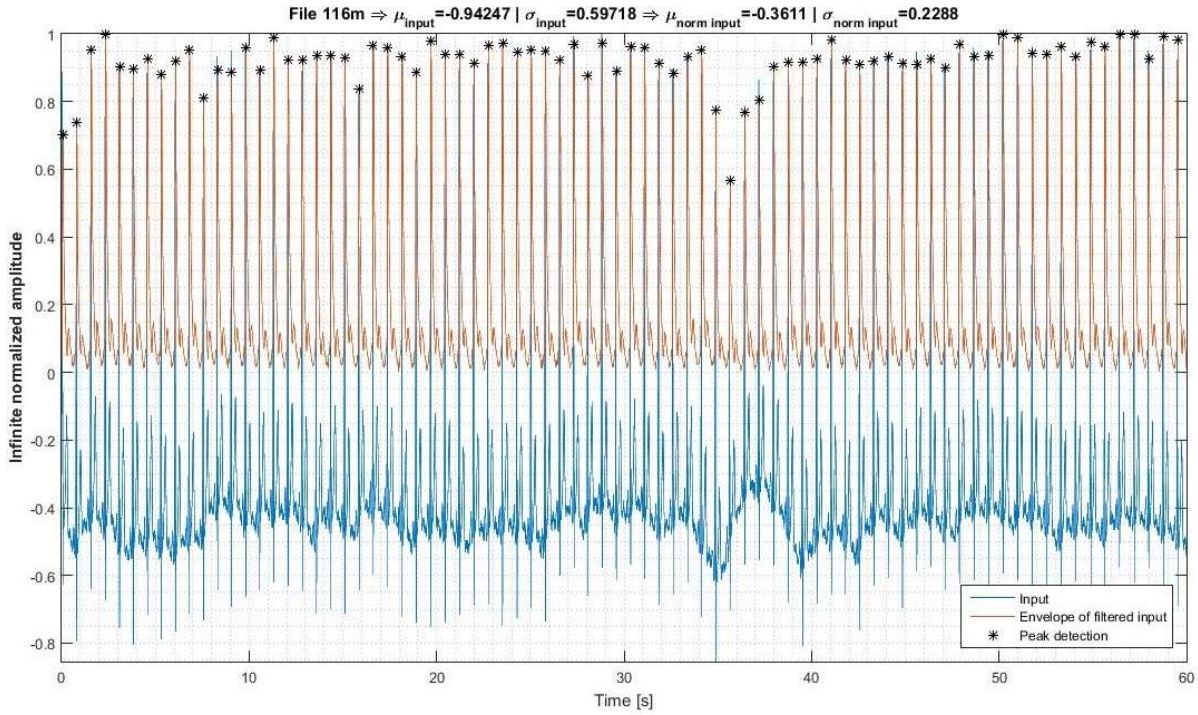


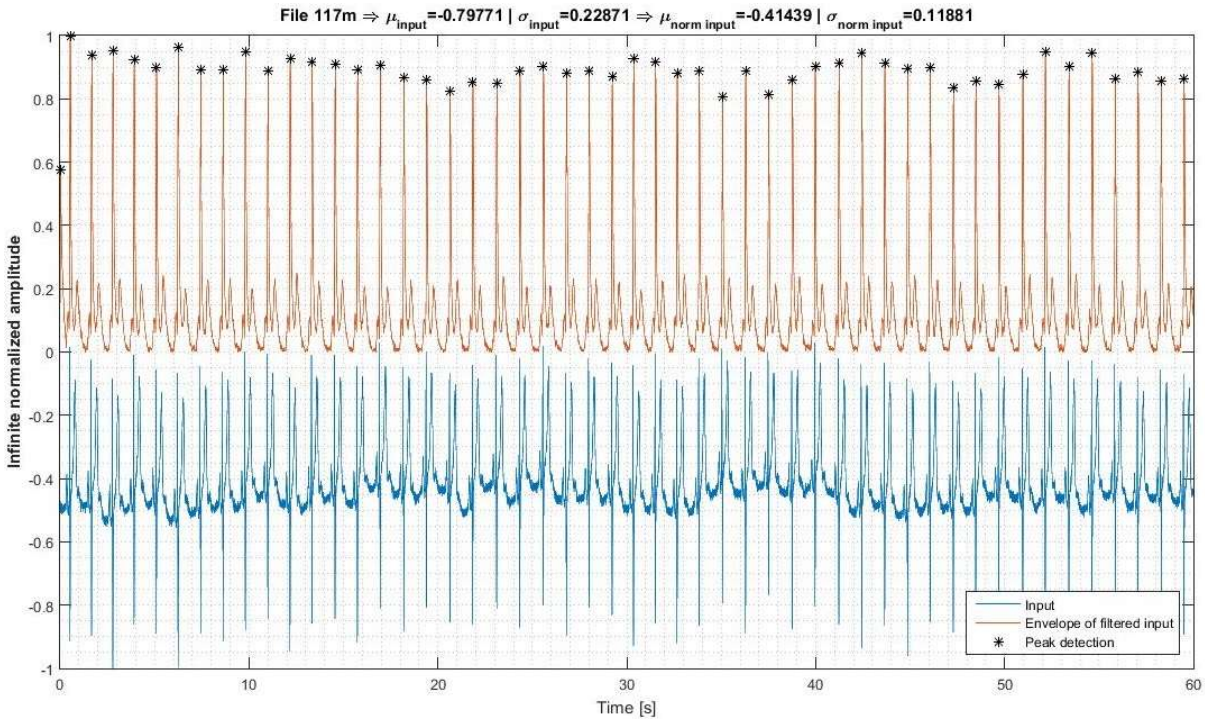

File 112m ⇒ μ_{input}=-0.84239 | σ_{input}=0.19579 ⇒ $\mu_{norm\ input}$=-0.63817 | $\sigma_{norm\ input}$=0.14832
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection

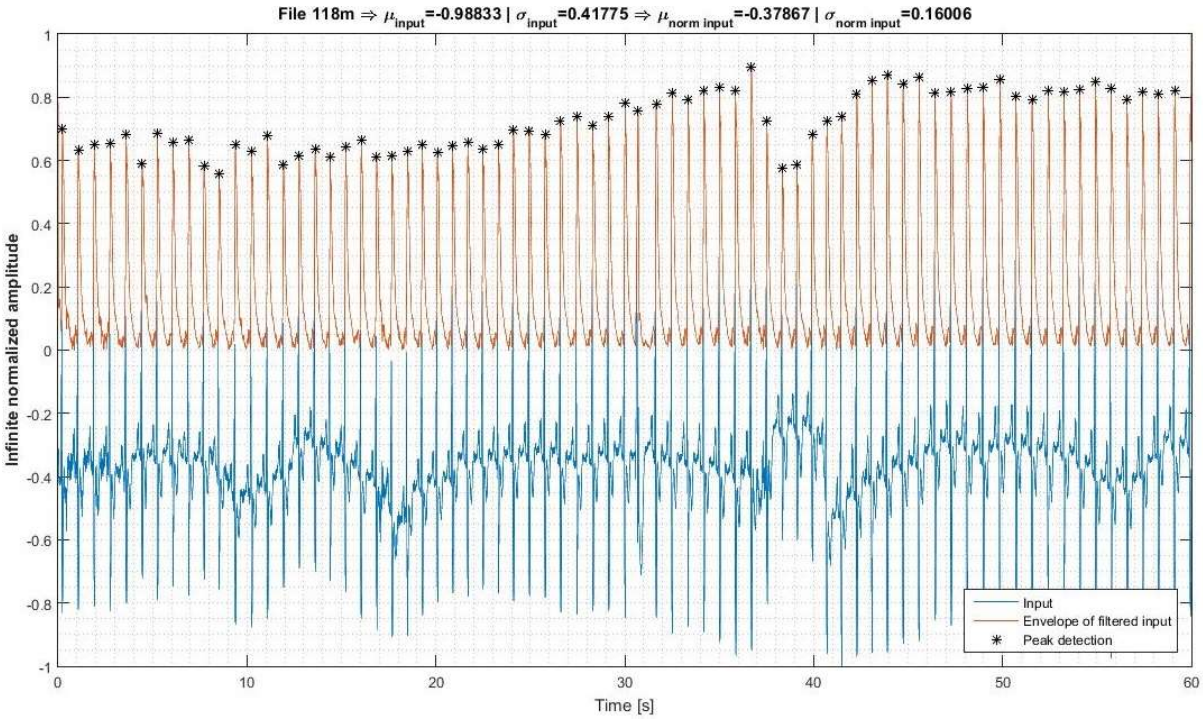

File 113m ⇒ μ_{input}=-0.17248 | σ_{input}=0.42294 ⇒ $\mu_{norm\ input}$=-0.073868 | $\sigma_{norm\ input}$=0.18113
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection

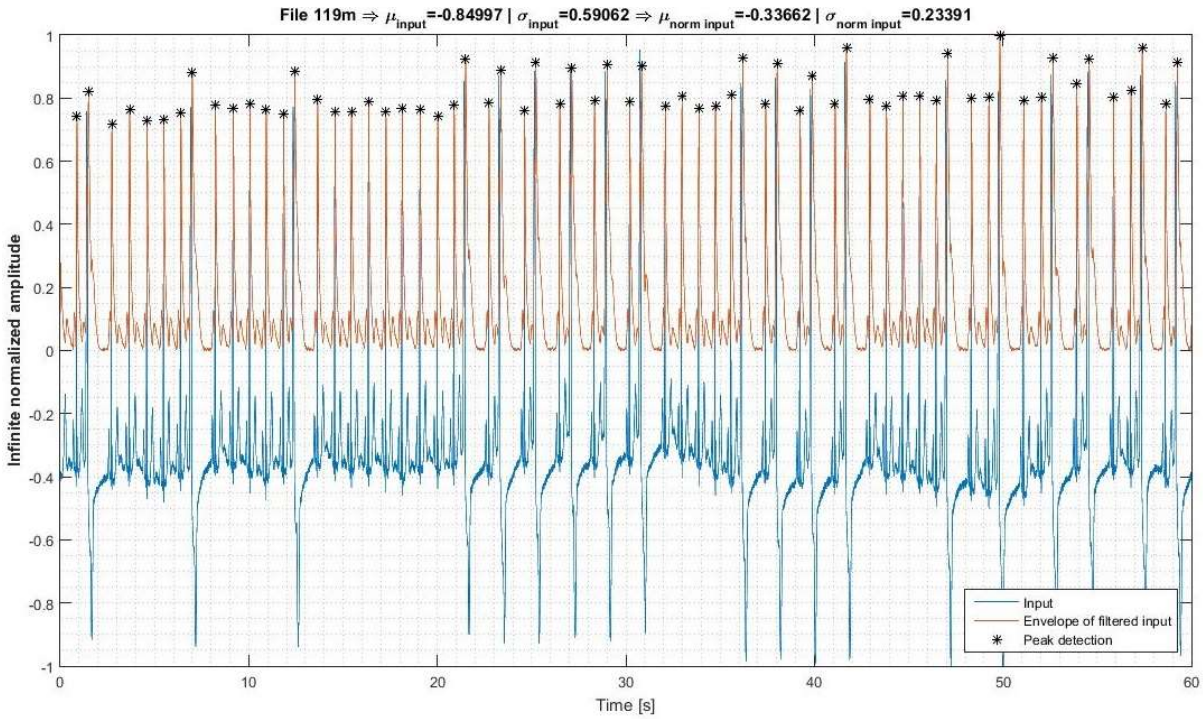

File 114m ⇒ μ_{input}=0.042102 | σ_{input}=0.29384 ⇒ $\mu_{norm\ input}$=0.018305 | $\sigma_{norm\ input}$=0.12776
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection


File 115m ⇒ μ_{input}=-0.55186 | σ_{input}=0.34405 ⇒ $\mu_{norm\ input}$=-0.28668 | $\sigma_{norm\ input}$=0.17873
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection

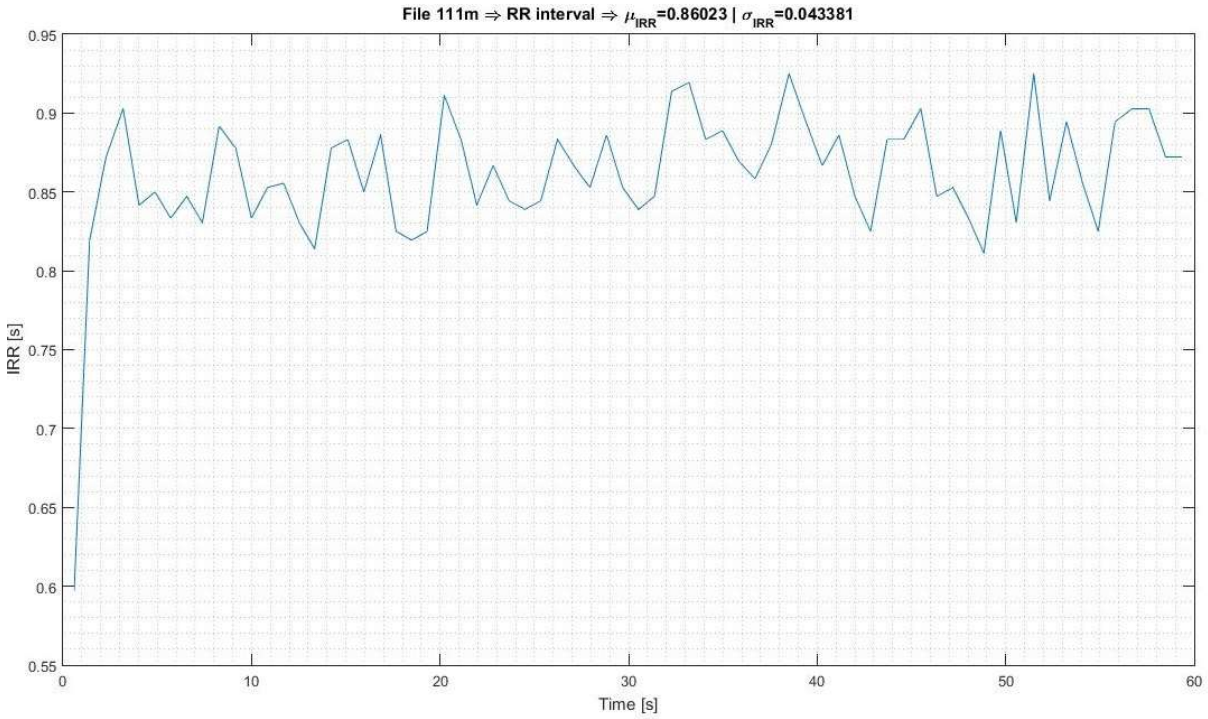

File 116m ⇒ μ_{input}=-0.94247 | σ_{input}=0.59718 ⇒ $\mu_{norm\ input}$=-0.3611 | $\sigma_{norm\ input}$=0.2288
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection

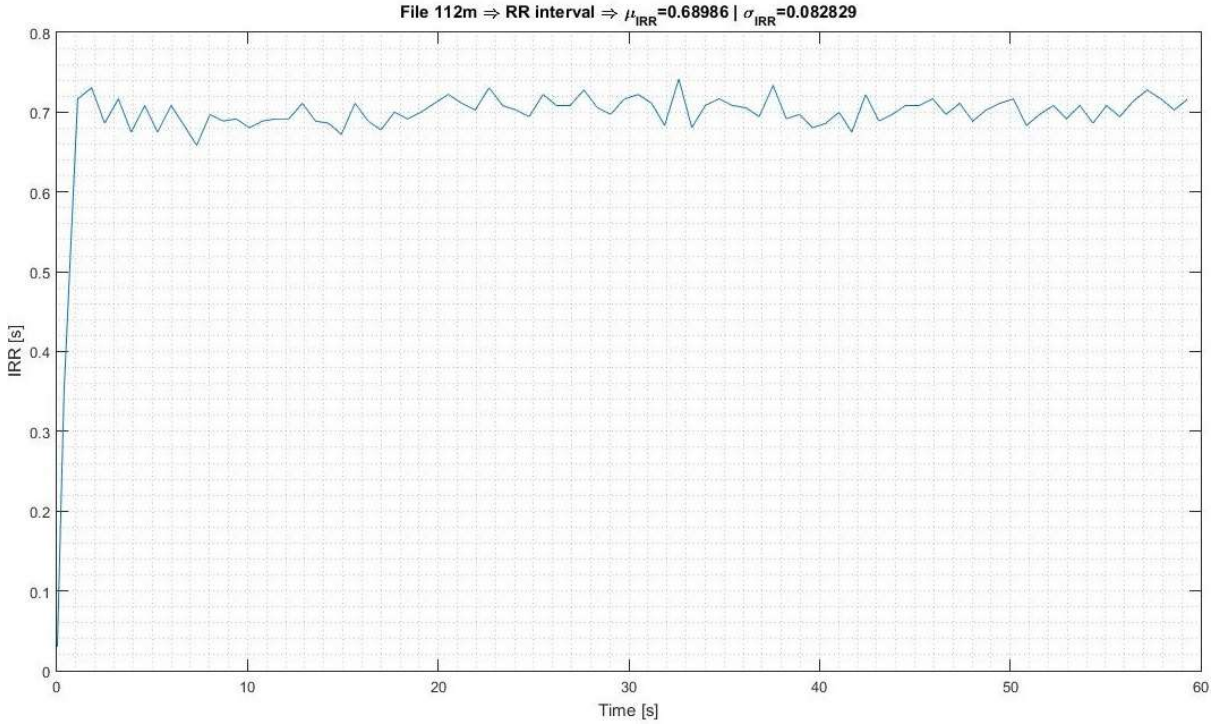

File 117m ⇒ μ_{input}=-0.79771 | σ_{input}=0.22871 ⇒ $\mu_{norm\ input}$=-0.41439 | $\sigma_{norm\ input}$=0.11881
Infinite normalized amplitude
Time [s]
Input
Envelope of filtered input
Peak detection

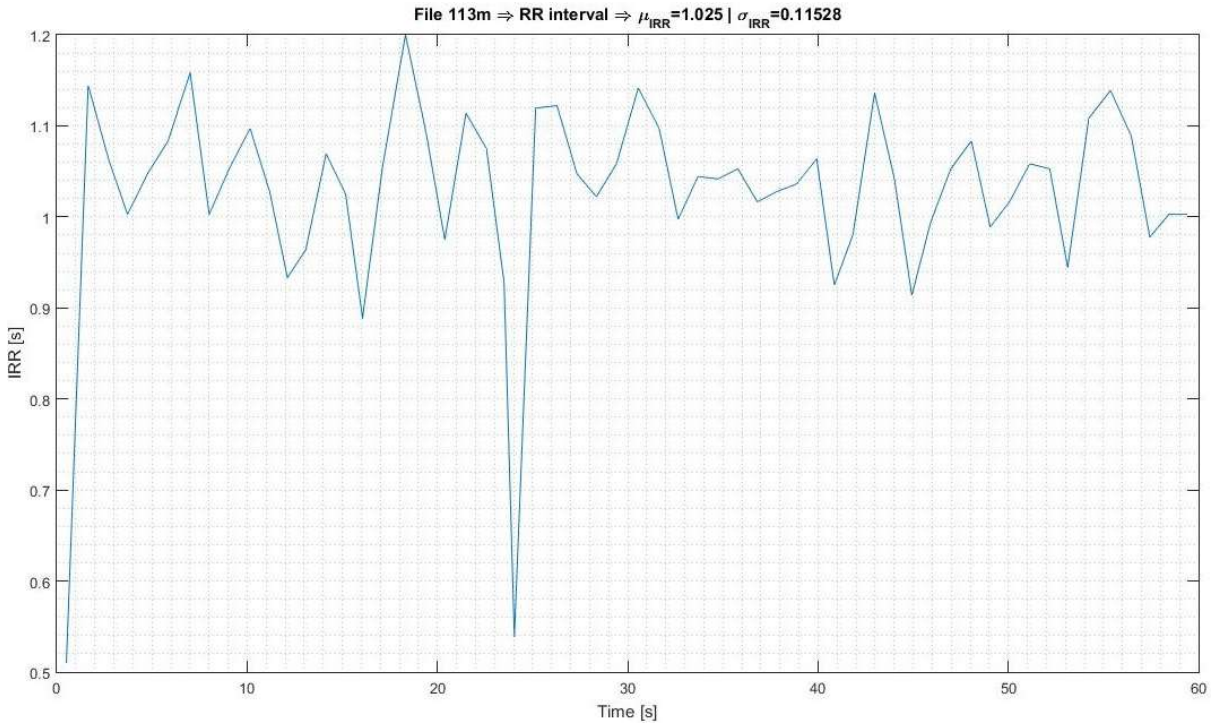


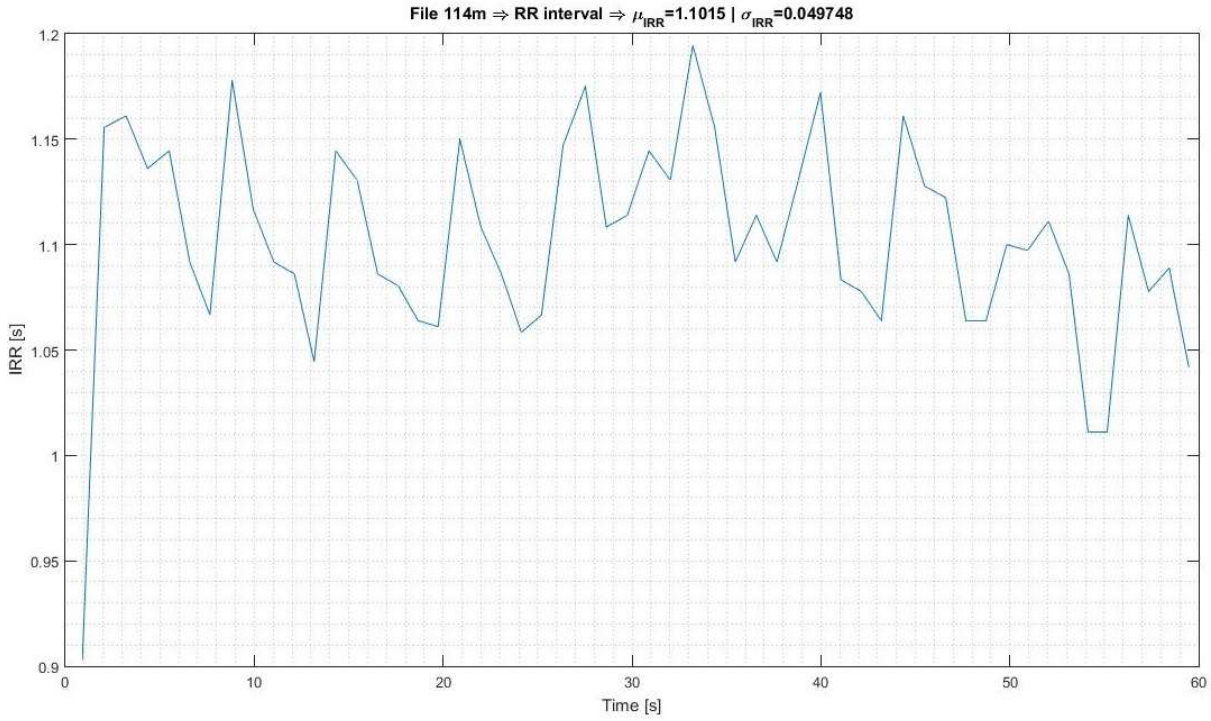


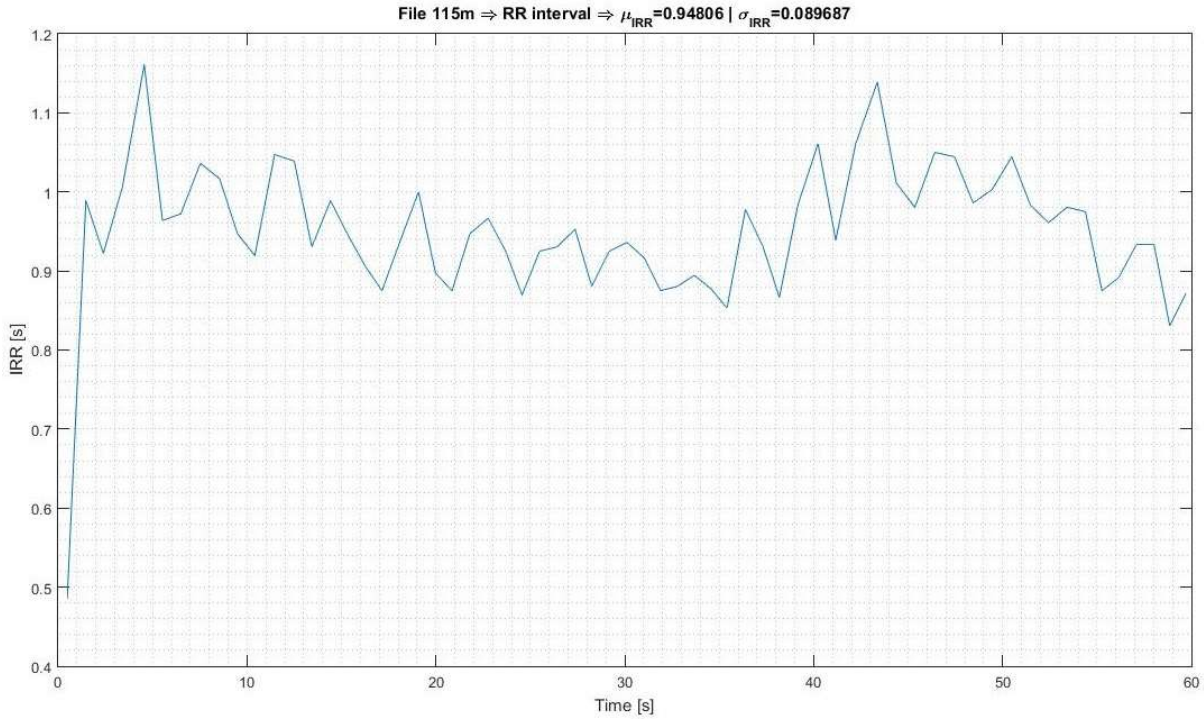
3.3) Por fim, os gráficos dos intervalos RR:

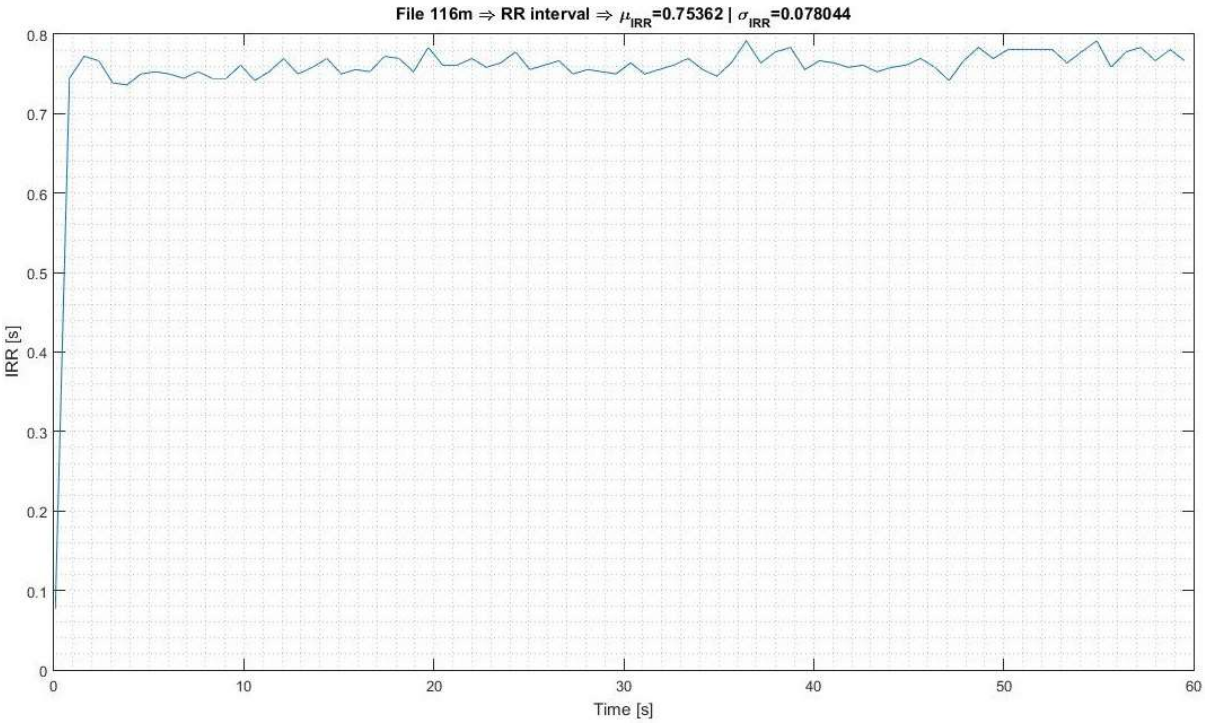

File 111m ⇒ RR interval ⇒ μ_{IRR}=0.86023 | σ_{IRR}=0.043381
IRR [s]
Time [s]
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
0
10
20
30
40
50
60

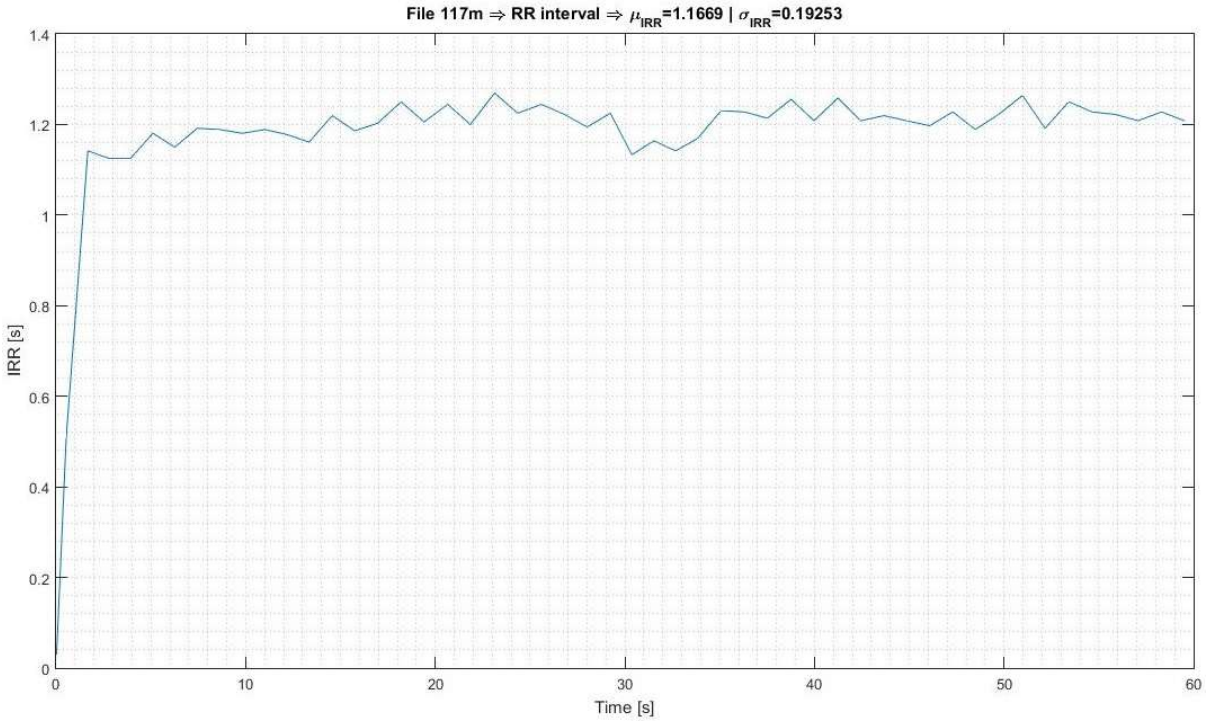

File 112m ⇒ RR interval ⇒ μ_{IRR}=0.68986 | σ_{IRR}=0.082829
IRR [s]
Time [s]
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
60

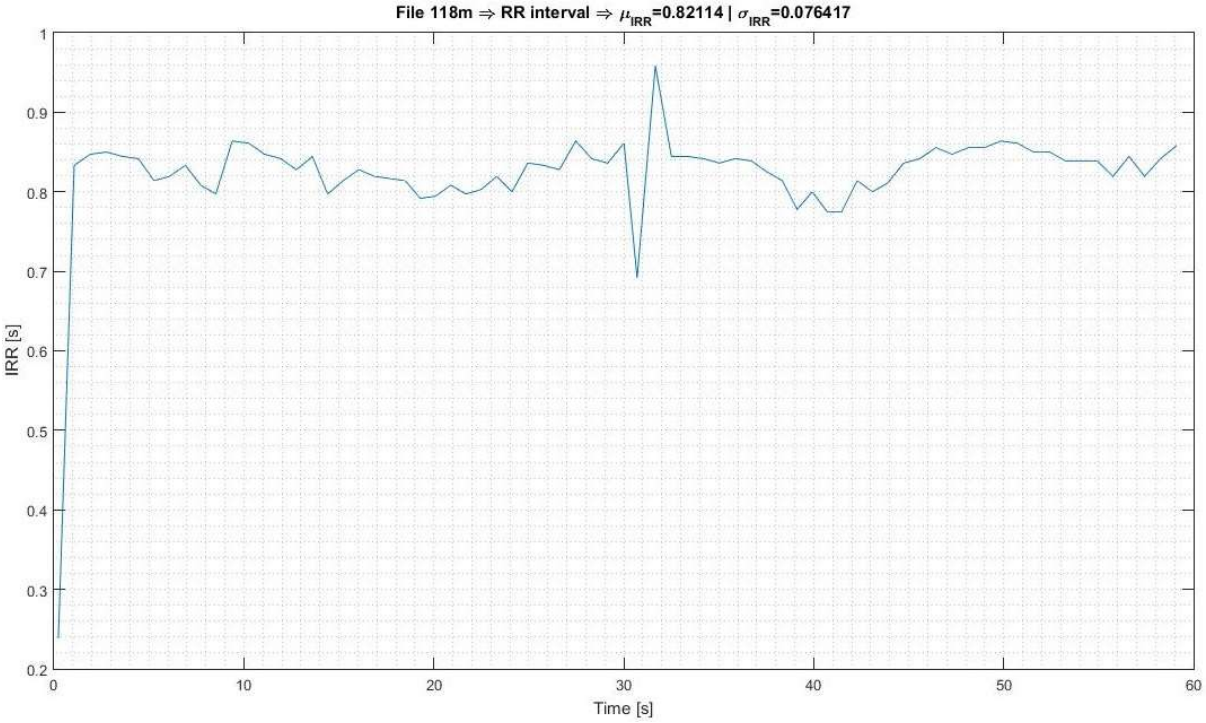

File 113m ⇒ RR interval ⇒ μ_{IRR}=1.025 | σ_{IRR}=0.11528
IRR [s]
Time [s]
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0
10
20
30
40
50
60

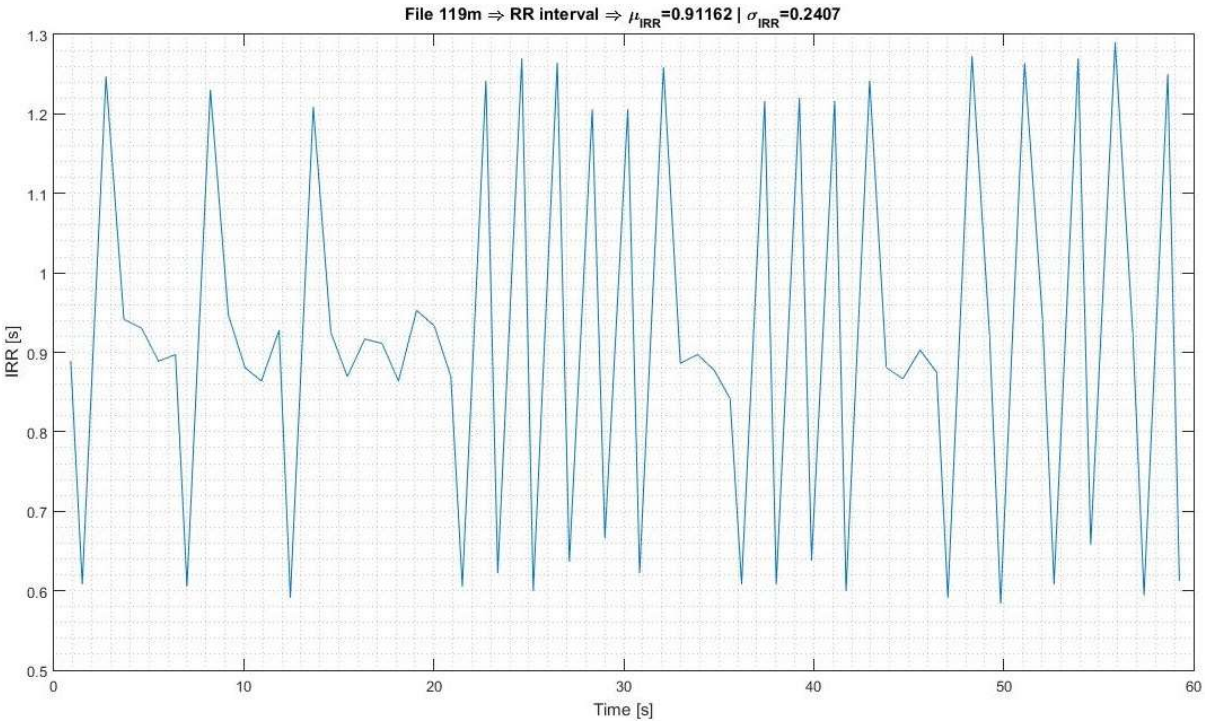

File 114m ⇒ RR interval ⇒ μ_{IRR}=1.1015 | σ_{IRR}=0.049748
IRR [s]
Time [s]
1.2
1.15
1.1
1.05
1
0.95
0.9
0
10
20
30
40
50
60


File 115m ⇒ RR interval ⇒ μ_{IRR}=0.94806 | σ_{IRR}=0.089687
IRR [s]
Time [s]
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
0
10
20
30
40
50
60


File 116m ⇒ RR interval ⇒ μ_{IRR}=0.75362 | σ_{IRR}=0.078044
IRR [s]
Time [s]
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
60


File 117m ⇒ RR interval ⇒ μ_{IRR}=1.1669 | σ_{IRR}=0.19253
IRR [s]
Time [s]
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
10
20
30
40
50
60


File 118m ⇒ RR interval ⇒ μ_{IRR}=0.82114 | σ_{IRR}=0.076417
IRR [s]
Time [s]
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
10
20
30
40
50
60



4) Conclusões

O meu algoritmo de detecção, em geral, funciona, mas, da maneira que eu implementei, não é possível executá lo em tempo real. Além disso, ele não é adaptativo, então, para casos gerais, ele não funcionará.

Embora a transformada de Hilbert pode ser executada usando FFT, que é bem rápida, ela necessita da multiplicação de números complexos seguido de uma IFFT para gerar a função analítica, além de ter que calcular o módulo desta para obter o envelope.

Assim, da maneira como eu implementei, o meu algoritmo certamente tem maior complexidade que o algoritmo Pan-Tompkins, tanto em relação fase de filtragem (pré-detecção de picos), pois este último se trata apenas da aplicação de filtros lineres e do cálculo do quadrado, quanto ao algoritmo adaptativo de detecção de picos, pois envolve uma menor quantidade de comparações e operações algébricas. [1, Cap 4]

Por fim, embora o meu algoritmo tenha produzido resultados interessantes, ele não pode ser executado em tempo real como o algoritmo de Pan-Tompkins, sendo que este também é muito preciso.

5) Referências

[1] Biomedical signal analysis - 2e - Rangayyan (Wiley)

[2] Discrete-time signal processing - 3e - Oppenheim (Pearson)

6) Código em Matlab

6.1) MA_fir.m (resposta impulsiva do filtro MA FIR passa-baixa)

```
%%% MA FIR filter
close; clear; clc;
fo = 10; % filter order
fs = 360; % sample frequency
n = 2^12; % number of points
imp = zeros(1,n); % impulse
imp(1) = 1;
h = filter(ones(1,fo)/fo, 1, imp); % impulse response

H = fftshift(fft(h));
H_mod = 20*log10(abs(H)); % amplitude response in dB
H_ang = rad2deg(angle(H)); % phase response in degrees
f = (-n/2:n/2-1)*(fs/n); % frequency

subplot(2,1,1);
plot(f,H_mod,f,-3*ones(1,n));
title(['Amplitude response of MA FIR filter of order ' num2str(fo)]);
xlabel('f [Hz]'); ylabel('20*log_{10}(|H(f)|) [dB]');
legend('20*log_{10}(|H(f)|)','-3dB');
xlim([f(1),f(end)]); grid minor;

subplot(2,1,2);
plot(f,H_ang);
title(['Phase response of MA FIR filter of order ' num2str(fo)]);
xlabel('f [Hz]'); ylabel('\angle(H(f)) [°]');
xlim([f(1),f(end)]); grid minor;
```

6.2) High_pass_Butterworth.m (resposta impulsiva do filtro Butterworth passa-alta)

```
%%% Butterworth high-pass filter order=4 fc=5
close; clear; clc;
fs = 360; % sample frequency
fc = 5; % cutting frequency
fo = 4; % filter order
[H_num, H_den] = butter(fo, 2*fc/fs, 'high');

n = 2^12; % number of points
imp = zeros(1,n); % impulse
imp(1) = 1;
h = filter(H_num, H_den, imp); % impulse response

H = fftshift(fft(h));
H_mod = 20*log10(abs(H)); % amplitude response in dB
H_ang = rad2deg(angle(H)); % phase response in degrees
f = (-n/2:n/2-1)*(fs/n); % frequency

subplot(2,1,1);
plot(f,H_mod,f,-3*ones(1,n));
title(['Amplitude response of Butterworth high-pass filter of order '...
    num2str(fo) ' with cutting frequency of ' num2str(fc) 'Hz']);
xlabel('f [Hz]'); ylabel('20*log_{10}(|H(f)|) [dB]');
```

```
legend('20*log_{10}(|H(f)|)','-3dB');
xlim([f(1),f(end)]); grid minor;

subplot(2,1,2);
plot(f,H_ang);
title(['Phase response of Butterworth high-pass filter of order '...
    num2str(fo) ' with cutting frequency of ' num2str(fc) 'Hz']);
xlabel('f [Hz]'); ylabel('\angle(H(f)) [°]');
xlim([f(1),f(end)]); grid minor;
```

6.3) PTC3456_QRS_Detection_My.m (meu algoritmo de detecção de QRS)

```
%%% ECG signal QRS complex detection
close; clear; clc; name = '119m';
fid = fopen([name '.info'], 'rt');
fgetl(fid); fgetl(fid); fgetl(fid);
freqint = sscanf(fgetl(fid), 'Sampling frequency: %f Hz  Sampling interval:
%f sec');
fs = freqint(1); fgetl(fid);
[row, signal, gain, base, unit] =
strread(fgetl(fid),'%d%s%f%f%s','delimiter','\t');
fclose(fid);

%%% Input ECG signal
data = load([name '.mat']);
x = (data.val-base)/gain;
l = length(x);
t = (0:l-1) / fs;
mean_x = mean(x);
std_x = std(x);

%%% MA FIR low-pass filter
MA_fo = 10; % filter order
x_ma = filter(ones(1,MA_fo)/MA_fo, 1, x);

%%% Butterworth high-pass filter
B_fc = 5; B_fo = 4; % cutting frequency and filter order
[H_B_num, H_B_den] = butter(B_fo, 2*B_fc/fs, 'high');
x_ma_b = filter(H_B_num, H_B_den, x_ma);

%%% Hilbert transform
x_ma_b_h = abs(hilbert(x_ma_b));

%%% My detection algorithm
dc = 0; % detection counter
cr = 20; % comparison range + 1
det_v = zeros(1,l); % detected points vector
tshd = 0.35 * max(x_ma_b_h); % threshold
for i = 2:l-1
    if i < cr+1
        c1 = 1;
        c2 = i + cr - 1;
    elseif i < l-cr
        c1 = i - cr + 1;
        c2 = i + cr - 1;
    else
        c1 = i - cr + 1;
        c2 = l;
    end
```

```
    svb = max(x_ma_b_h(c1:i-1)); % before
    svn = x_ma_b_h(i); % now
    sva = max(x_ma_b_h(i+1:c2)); % after
    if (svn>svb) && (svn>sva) && (svn>tshd)
        dc = dc + 1;
        det_v(dc) = i;
    end
end
det_v = det_v(1:dc);

% RR interval
IRR = zeros(1,dc-1);
IRR(1) = det_v(1);
for i = 2:dc
    IRR(i) = det_v(i) - det_v(i-1);
end
IRR = IRR / fs;
mean_IRR = mean(IRR);
std_IRR = std(IRR);

%%% Comparison
x = x / norm(x, 'inf'); % x = x/max(x)
mean_xn = mean(x);
std_xn = std(x);
x_ma_b_h = x_ma_b_h / norm(x_ma_b_h, 'inf');

figure(1);
plot(t, x, t, x_ma_b_h);
hold on;
scatter(t(det_v), x_ma_b_h(det_v), '*', 'k');
hold off;
title([ 'File ' name ' \Rightarrow \mu_{input}=' num2str(mean_x)...
    ' | \sigma_{input}=' num2str(std_x) ' \Rightarrow \mu_{norm input}='...
    num2str(mean_xn) ' | \sigma_{norm input}=' num2str(std_xn)]);
xlabel('Time [s]');
ylabel('Infinite normalized amplitude','FontWeight','bold');
legend('Input','Envelope of filtered input','Peak detection',...
    'Location','southeast');
ylim([min(min(x),min(x_ma_b_h)), max(max(x),max(x_ma_b_h))]);
grid minor;

figure(2);
plot(t(det_v), IRR);
title(['File ' name ' \Rightarrow RR interval \Rightarrow \mu_{IRR}='...
    num2str(mean_IRR) ' | \sigma_{IRR}=' num2str(std_IRR)]);
xlabel('Time [s]');
ylabel('IRR [s]');
grid minor;
```