Universidade Federal de Santa Catarina

Programa de Pós-Graduação em Engenharia Elétrica

Processamento de Sinais Biomédicos

Gustavo Leão Mourão
Pedro Henrique Kappler Fornari
Rafael Gonçalves Licursi de Mello
Ricardo Filipe Dalri

Florianópolis, 2016

1. Introdução

Eletrocardiograma (ECG) trata-se de uma técnica capaz de representar e armazenar de forma gráfica a atividade elétrica referente ao coração humano. Trata-se de um método simples e não-evasivo. Um ECG consiste em eletrodos dispostos sobre a superfície do corpo do indivíduo. O equipamento de ECG mais comum utiliza 12 eletrodos os quais amostram 6 sinais provenientes de terminações pulmonares e 6 relacionadas à membros.

Um complexo de ECG é composto por cinco (5) ondas: P, Q, R, S e T, conforme representado na Figura 1. Tais sinais são obtidos através de circuitos simples, contendo amplificadores e conversores analógico-digitais.

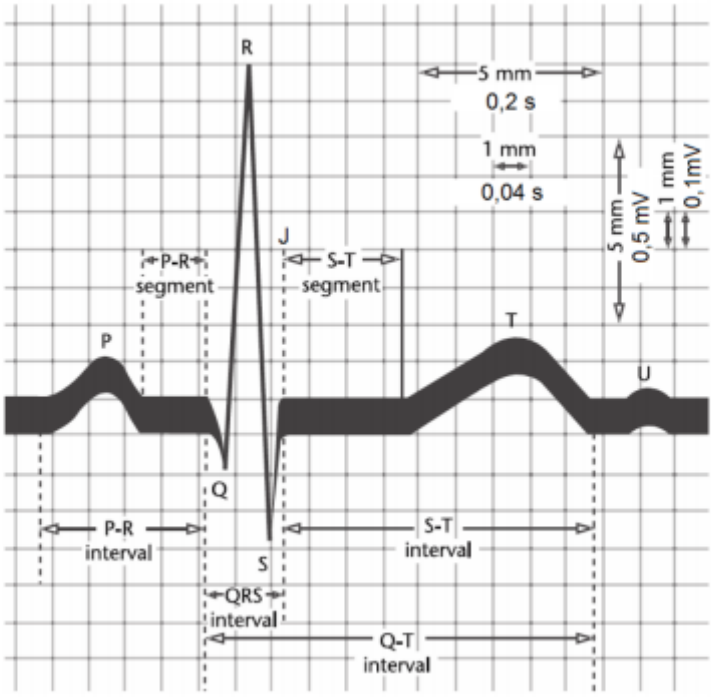


Figura 1. Complexo PQRST [6].

Considerando-se medições provenientes de pacientes com função cardíaca normal, os valores referentes aos intervalos representados na Figura 1 encontram-se representados na Tabela 1.

Tabela 1. Duração dos intervalos (Complexo PQRST). [7]

Intervalo	**RR**	**PR**	**QRS**	**ST**	**QT**
Duração (*ms*)	600-1200	120-200	80-100	320	420

Um ciclo cardíaco corresponde ao registro referente ao funcionamento de quatro câmaras (átrios direito e esquerdo e ventrículos direito e esquerdo). A onda P ocorre quando é criado um potencial de ação que despolariza o átrio. Posteriormente, o complexo QRS corresponde despolarização ventricular, de forma que o intervalo entre o início da onda P e o início do complexo QRS corresponde ao intervalo PR.

O intervalo PR, por sua vez, representa o tempo de ocorrência do impulso elétrico de despolarização do átrio ao início da onda R.

A onda T ocorre após o complexo QRS, sendo resultado da repolarização ventricular. Naturalmente é assimétrica, apresentando maior inclinação em sua terminação, comparando-a ao seu início.

Em termos gerais, através da leitura adequada proveniente de um ECG, é possível diagnosticar lesões miocárdicas e presenças de infarto, além de avaliações relacionadas a distúrbios metabólicos causados por efeito colaterais provenientes de farmacoterápicos.

Um problema inerente à aquisição e correta interpretação de tais sinais trata-se da presença de ruídos imbuídos e relativos ao sistema de medição, os quais alteram a característica verdadeira do sinal aquisitado. Como exemplo é possível citar a frequência relativa ao sistema de alimentação (60Hz), assim como frequências musculares e respiratórias. Neste sentido, é imprescindível destacar a necessidade referente à minimização de tais ruídos através de técnicas de filtragem digital para realização de posteriores análises.

Diferentes algoritmos e estratégias para detecção de batimentos cardíacos são baseados na detecção do complexo QRS, de forma que a frequência cardíaca pode ser computada a partir da distância entre os complexos QRS. O complexo QRS pode ser detectado utilizando, por exemplo, algoritmos baseados em Redes Neurais Artificiais, Transformadas de Wavelet ou banco de filtros [1]. Além disso, outra maneira de detecção do complexo QRS é através da utilização de complexos adaptativos [2]. Métodos diretos para detecção de batimentos cardíacos são análise espectral do sinal de ECG [3] e métodos baseados na autocorrelação do sinal [4].

Desvantagens relativas aos métodos descritos anteriormente envolvem basicamente limitação em termos de processamento em tempo real. O projeto de filtros digitais e algoritmos para detecção de batimentos cardíacos são simples de implementação, todavia robustos.

2. Objetivos

O objetivo do estudo encontra-se fundamentado em aplicar técnicas de processamento e filtragem digital em sinais de Eletrocárdiograma (ECG), com a finalidade de redução de ruídos de fundo presente nos mesmos.

Como técnicas serão utilizadas filtros Moving Average e IRR (Infinite Impulse response) passa-altas. Ainda será utilizado o conceito de promediação para detecção de batimentos e do complexo PQRST, assim como estimação dos tempos dos intervalos referentes às ondas presentes em tal complexo.

3. Metodologia

Nesta seção serão apresentadas as técnicas e aporte metodológico utilizados no processamento dos sinais apresentados. Foram abordadas diferentes técnicas com o objetivo de redução de ruído sobre sinal amostrado e detectar pontos importantes para a avaliação dos exames. Para facilitar a portabilidade, todo o desenvolvimento se baseou em criar funções que sejam modulares, utilizadas apenas quando necessário e, além disso, aceitem qualquer sinal de ECG coletado.

O processo de filtragem é iniciado por uma reamostragem do sinal, assim é possível fazer com que a frequência de amostragem do exame não seja de grande importância para o restante do programa e que um número maior de amostras esteja disponível para ser trabalhado, resultando em funções de transferência mais precisas. Porém isso também faz com que o tempo de execução do programa aumente significativamente. A frequência de amostragem passou de 240 amostras/s para 4000 amostras/s, como solicitado pelo roteiro do trabalho.

Com o sinal agora reamostrado, foi desenvolvido um algoritmo para promediar o sinal. A Promediação é uma das técnicas estudadas para redução de ruído aleatório sobre o sinal, sendo que para isso é feita a separação do sinal por batimentos, ou seja, são selecionados períodos de ECG tendo como referência o pico da onda R, selecionando um número de amostras anteriores e um número de amostras posteriores a este pico. Com conjuntos de amostras selecionados, é possível tratar cada conjunto como um período do ECG, assim é feita a correlação cruzada entre os períodos a partir de uma amostra de referência, aplicando um deslocamento nas amostras, a fim de encontrar o ponto de máxima correlação entre um período de referência e os conjuntos posteriores.

Dado um resultado maior que 95% de correlação entre dois períodos, este conjunto é armazenado e no fim todos são somados e então normalizados pelo número de amostras, conforme a Equação 1, onde M é o número de conjuntos de amostras representando períodos do sinal utilizados e n é a amostra atual do filtro. Por conseguinte as amostras aleatórias, como é o caso do ruído, tendem a ser eliminadas, pois são diferentes em diferentes períodos e sua soma sucessiva tende a zero quando o número de amostras somadas tende ao infinito. Outro ponto positivo é que deslocamentos da média do sinal também são eliminados, o que ocorre durante a respiração quando a impedância do corpo aumenta e diminui ao longo do tempo, por exemplo, resultando em um sinal com características estacionárias e bem definidas.

$$Prom[n] = \frac{1}{M} * \sum_{i=0}^{i=M} Periodo(i[n]) \tag{1}$$

Uma vez realizada a promediação do sinal, foram aplicados filtros digitais, previamente desenvolvidos em MATLAB. A topologia do filtro escolhido foi o Butterworth, dado que apresenta maior linearidade na banda passante além de ter a atenuação suficiente na banda de rejeição. Os filtros aplicados foram passa altas e passa baixas nas frequências de 25, 40 e 80 Hz. Os resultados e discussão sobre os mesmos serão apresentados na próxima seção.

Os filtros utilizados foram filtros IIR (*Infinite Impulse Response*), que possuem resposta ao impulso infinita, e para isso fazem uso de funções recursivas. Ainda, os filtros IIR podem alcançar atenuações maiores que filtros FIR (*Finite Impulse Response*) além de consumirem capacidade de processamento menor e com isso serem mais ágeis do que filtros FIR [9] [10], o que no caso desse trabalho em específico não é muito relevante, dado que os dados estão armazenados no computador e a capacidade de processamento de um processador utilizado em desktops e notebooks excede as especificações e requisitos necessários para um processamento mais profundo, porém é sabido que o interesse em filtros que sejam aplicados diretamente em microprocessadores e controladores com capacidade de processamento muito inferior e com memória finita, que possam ser instalados junto aos equipamentos de aquisição de dados e assim apresentar gráficos, marcadores e informações interessantes em tempo real é muito maior, logo isso foi considerado nas etapas do projeto.

Feito isso, foram analisadas algumas característica que fornecem informações importantes sobre a saúde do paciente, nesse caso foram consideradas as ondas RR e QT, avaliando seus períodos. Segundo [11] essas informações podem eficientemente diagnosticar doenças em pacientes a partir de exames não intrusivos, identificando doenças como arritmia, taquicardia, palpitações, entre outros. O artigo ainda afirma que o período de repolarização, que é utilizado para avaliar a freqüência cardíaca pode em suma caracterizar todas as doenças citadas acima.

Para isso foi avaliado o período RR do sinal, avaliando o tempo entre os picos R já separados anteriormente, assim foi analisado o inverso do período para se obter a freqüência cardíaca dos pacientes, estimando uma normalidade que pode variar com a idade dos pacientes, sendo entre 80 e 100 batimentos por minuto em adolescentes e caindo com o passar dos anos para uma faixa entre 50 e 60 batimentos por minuto em idosos.

O período RR é de altíssima importância para o exame, porém ainda existe outro período que pode indicar anomalias no exame, este período é conhecido como QT, ou QTc, como será explicado a seguir. Segundo [12], o intervalo QT é uma medida relativa à frequência cardíaca do paciente, ou seja, relativa ao período RR, sendo diretamente proporcional ao mesmo. Assim, com o aumento da freqüência cardíaca, temos uma diminuição tanto do período RR quanto do período QT e vice versa. Por isso a análise de eletrocardiogramas passou a considerar esse fator nos exames, a fim de apresentar resultados mais concretos e fáceis de analisar. Feito isso, Bazett formulou a Equação 2, onde normalizou o período QT pela freqüência cardíaca do paciente, nomeando a nova medida de QTc, ou QT corrigido. [12] Ainda afirma que valores normais para o intervalo QTc giram em torno de 0.35 a 0.45 segundos, sendo maiores em mulheres do que em homens.

$$QTc = \frac{QT}{\sqrt{RR}} \tag{2}$$

Para caracterizar essa medida o seguinte procedimento foi seguido: Aplicou-se um filtro passa faixas, com freqüências de corte entre 0.5 e 45 Hz, a fim de selecionar apenas o intervalo QT. Definiram-se dois limiares de decisão, um para retirar variações na linha de base do sinal,

possivelmente não removidas pela promediação e um segundo limiar para detectar a onda T. Assim utilizam-se os picos R para encontrar a onda Q, que é definida pelo mínimo valor do sinal entre o pico R e os 0.040 segundos anteriores a ele. Assim que é encontrado o pico da onda T, encontra-se a onda S, detectada quando o limiar médio da onda é ultrapassado após a onda R, obtendo outro pico negativo do sinal. Por fim, mas não menos importante, é encontrada a onda T, a qual deve ser encontrada até 0.3 segundos após a detecção da onda S, sendo que para isso dois limiares são levados em consideração, a linha de base e a amplitude da ultima onda S detectada.

Finalmente, ainda foi desenvolvido um algoritmo para aplicar um filtro de média móvel ao sinal de ECG, avaliando as vantagens e desvantagens desse tipo de abordagem. Para isso foi feito um breve estudo sobre esse tipo de filtro.

Filtros de média móvel são muito similares a filtros FIR, onde um número de amostras são convoluídas com os coeficientes do filtro, porém neste caso, os coeficientes do filtro são todos idênticos e dados pela Equação 3, onde M é o número de termos do filtro e c[n] são os respectivos coeficientes. A saída do filtro pode ser observada na Equação 4, onde y[n] representa a saída do filtro, M representa o número de coeficientes do filtro, x[n] representa a entrada do filtro e j representa o atraso aplicado ao sinal.

$$c[n] = 1/M \tag{3}$$

$$y[n] = \frac{1}{M} * \sum_{j=0}^{M-1} x[n+j] \tag{4}$$

Assim esses filtros, por mais simples que sejam, apresentam resultados incríveis no domínio do tempo, visto que normalizam as amostras presentes utilizando M amostras passadas, eliminando assim componentes de altas frequências e atuando como um filtro passa baixas. Isso fica ainda mais evidente quando verificamos que a resposta do filtro no domínio do tempo é retangular, ou seja, no domínio da freqüência o filtro é representado pela Equação 5.

$$H(f) = \frac{\sin[\pi f M]}{M \sin[\pi f]} \tag{5}$$

Como é possível verificar essa equação ilustra que, conforme o número de termos aumenta maior é o número de frequências filtradas, ou seja, como cita [13], uma boa performance no domínio do tempo resulta em uma performance ruim no domínio da frequência e vice versa, de forma que o projetista deva determinar corretamente o número de termos do filtro, fator determinante na qualidade do filtro. A Figura 2 ilustra bastante a eficiência espectral do filtro conforme o número de termos varia, assim como a Figura 3 apresenta a variação do sinal no domínio do tempo conforme o número de termos do filtro varia. É possível observar a diferença sobre o sinal filtrado em função do aumento do número de amostras. Porém também fica visível

a grave atenuação relacionadas à componentes intrínsecas ao sinal, sobretudo em altas frequências.

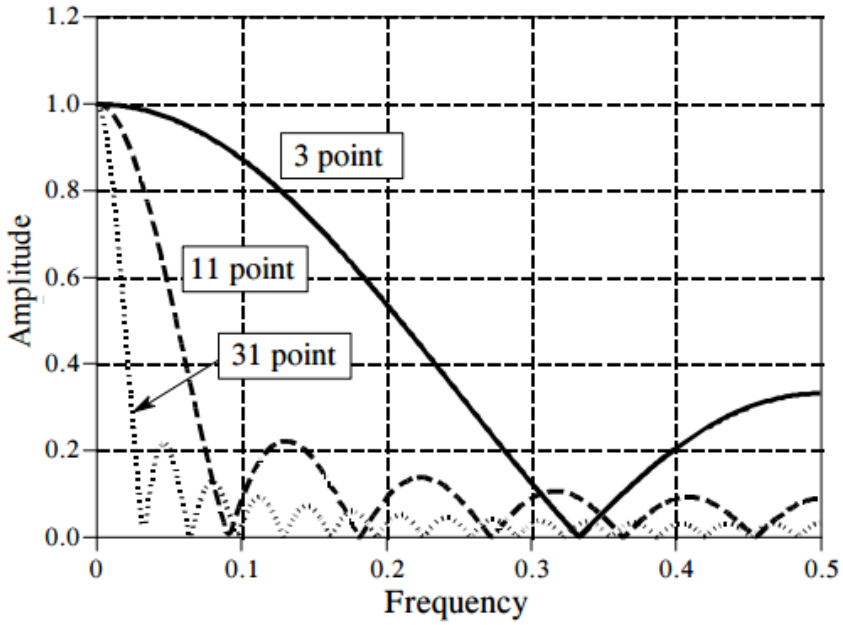


Figura 2 – Eficiência espectral do filtro de média móvel [13]

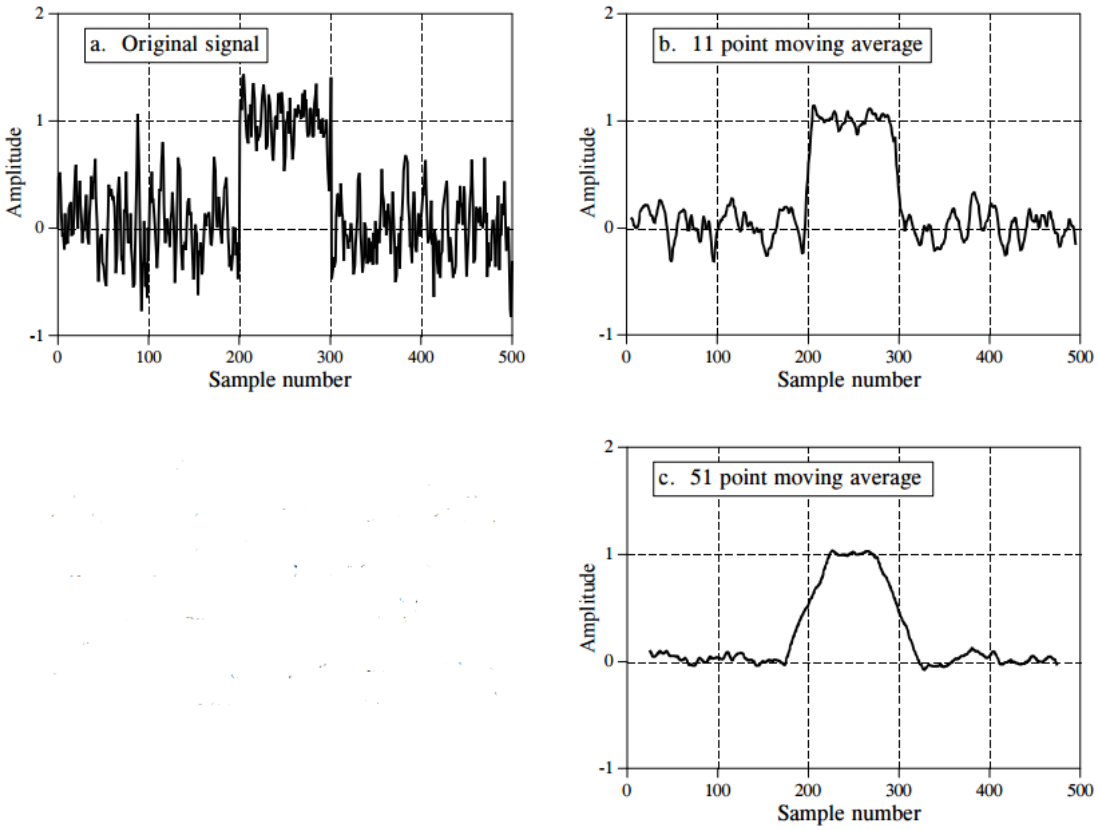


Figura 3 – Filtro de média móvel no domínio do tempo [13]

Assim, foi implementado um filtro de média móvel, utilizando sempre amostras passadas para avaliar a média, de forma que o filtro seja causal e implementável na prática [10], implementado em sua forma direta, utilizando atrasos e coeficientes constantes e idênticos, conforme o diagrama de blocos apresentado na Figura 4, retirado de [14].

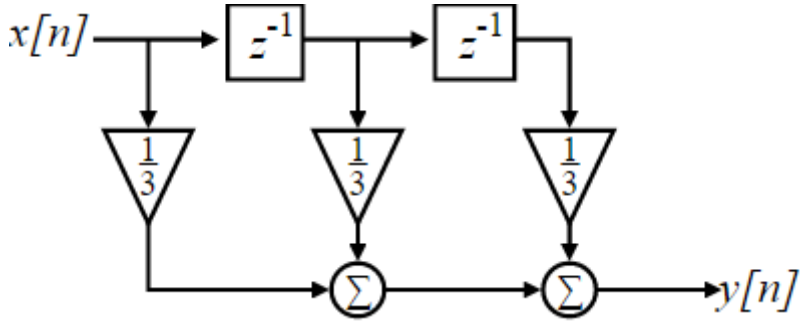


Figura 4 – Diagrama de blocos do filtro de média móvel implementado.

Foi escolhido um número de amostras diferente ao indicado no roteiro, visto que utilizando-se 3 amostras não foi observada alteração sobre o sinal processado. A Equação 6 foi utilizada para escolher o número de amostras, onde T é o tempo total utilizado no filtro dado em segundos, M é o número de termos e Fs a frequência de amostragem do sinal.

$$T = \frac{M}{Fs} \tag{6}$$

Assim o filtro escolhido levou em consideração números de amostras iguais a 100, 150 e 200, os resultados serão apresentados na próxima seção. Ainda foi separado o sinal completo em conjuntos de um minuto cada, para que, com essas amostras fossem realizados cálculos estatísticos que posteriormente foram comparados a amostra completa contendo todos os conjuntos, assim dados estatísticos do sinal foram levantados e maiores discussões serão apresentadas juntamente aos resultados.

4. Resultados e Discussão

Nas próximas seções serão apresentados os resultados obtidos a partir da aplicação dos procedimentos metodológicos apresentados na seção anterior.

Inicialmente o sinal referente à amostragem do ECG (*ECG01_5min@240Hz.dat*) foi reamostrado para uma frequência de $4000Hz$. Uma janela de $10s$ referente ao sinal reamostrado encontra-se representado na Figura 5.

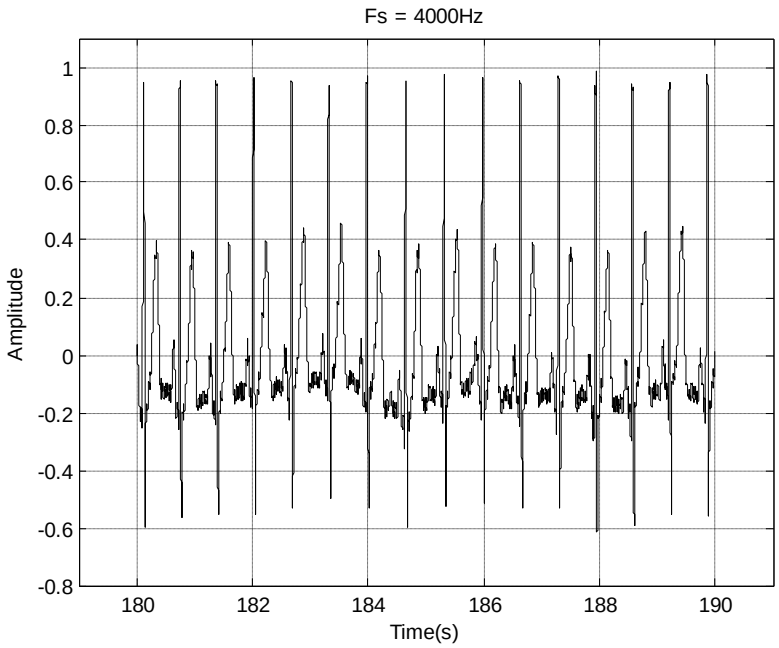


Figura 5. Janela de 10s referente ao sinal de ECG reamostrado para $4000Hz$
(ECG01_5min@240Hz.dat)

Posteriormente com o objetivo de detectar os batimentos cardíacos (*onda R*) foi desenvolvido um algoritmo fundamentado na obtenção da derivada do sinal reamostrado, conforme sintetizado na Figura 6.

Posterior à fase de reamostragem do sinal foi obtida a derivada do mesmo. Uma vez obtida a derivada do sinal reamostrado, foi obtido a energia de tal resposta sendo determinado um limiar de detecção (Threshold) [5]. A referência referente ao limite de detecção foi determinada conforme representado na Equação 7.

$$Threshold = \left| max\left(\frac{d(x(t)}{dt}\right)\right| \times \frac{1}{2} \tag{7}$$

Onde $x(t)$ representa o sinal de ECG.

Figura 6. Fluxograma: Algoritmo de detecção de batimentos cardíacos

Na Figura 7 encontra-se representado a derivada do sinal de ECG, durante o intervalo de tempo analisado, assim como o limiar de detecção estabelecido.

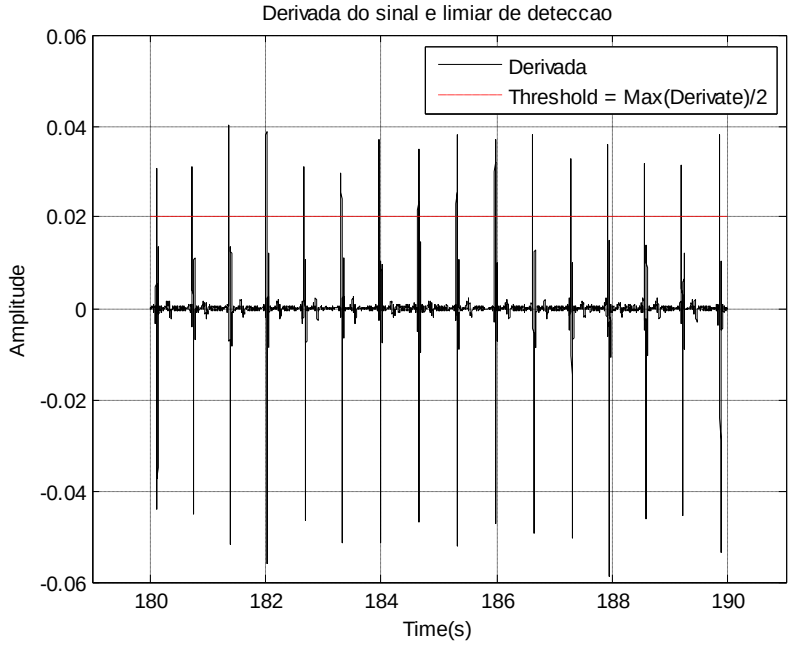


Figura 7. Derivada do sinal de ECG e limiar de detecção estabelecido.
(ECG01_5min@240Hz.dat)

Através da Figura é possível delinear os pontos os quais a derivada do sinal de ECG é superior ao limiar de detecção estabelecido. Tais pontos correspondem aos locais de máxima derivada, sendo porventura os picos de batimento cardíaco. Sendo assim, nos locais em que a derivada do sinal de ECG for superior ao limiar de detecção pode-se afirmar que tal região trata-se de um batimento cardíaco.

Na Figura 8 encontra-se representado o sinal de ECG e os relativos batimentos cardíacos identificados, considerando-se a estratégia descrita, referente á janela de $10s$ analisada.

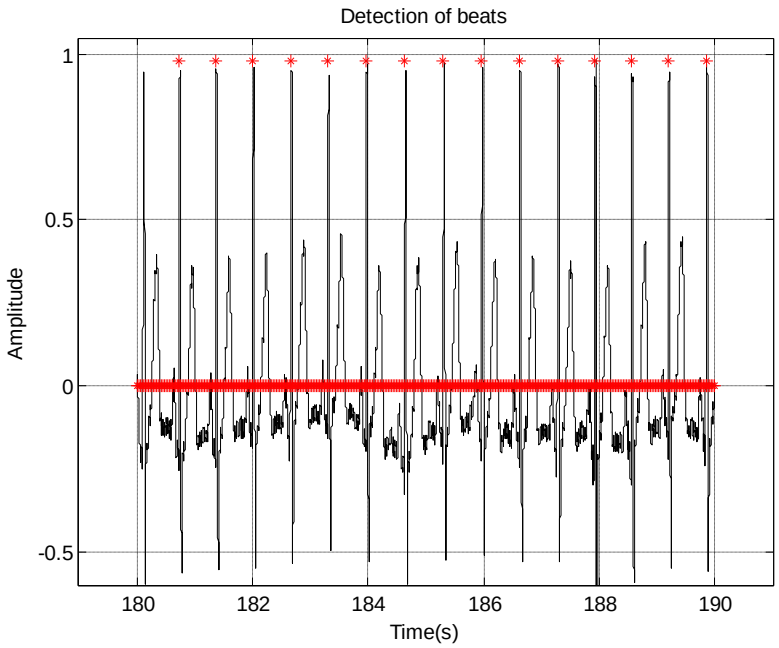


Figura 8. Janela de 10s referente ao sinal de ECG e identificação dos batimentos cardíacos. *(ECG01_5min@240Hz.dat)*

Uma vez detectados os batimentos, objetivou-se promediar o complexo PQRST. Em termos gerais, um sinal de EEG apresenta componentes bem definidas e estacionárias, relacionadas às características do processo analisado (funcionamento cardíaco - $x(t)$) somado às componentes ruidosas que podem ser associadas a um ruído branco representativo de um processo gaussiano, ($n(t)$), de média zero e variância σ^2, conforme representado em (8).

$$y_k(t) = x(t) + n_k(t) \tag{8}$$

Neste sentido, promediação significa manter a amplitude de um sinal que apresenta componentes definidas e estacionárias e reduzir a variância do ruído por um fator N, conforme representado na Equação 9.

$$y(t) = \frac{1}{N} \cdot \sum_{k=1}^{N} y_k(t) = x(t) + \frac{1}{N} \cdot \sum_{k=1}^{N} n_k(t) \tag{9}$$

Para promediação do complexo PQRST é notório destacar a necessidade em, inicialmente, alinhar o conjunto de complexos presentes. Para tanto foi utilizado o conceito de correlação cruzada em torno de cada batimento detectado.

O conceito de correlação cruzada mede a similaridade entre dois sinais em função do atraso entre ambos. A Equação 10 sintetiza o conceito de correlação cruzada, enquanto a Equação 11 representa o conceito de coeficiente de correlação cruzada.

$$r_{12}(j) = \frac{1}{N} \cdot \sum_{n=1}^{N-1} x_1(n) \cdot x_2(n+j) \tag{10}$$

$$r_{xy} = \frac{\sum_{n=1}^{N}(x(n) - \bar{x}) \cdot (y(n) - \bar{y})}{\sqrt{\sum_{n=1}^{N}(x(n) - \bar{x})^2 \cdot \sum_{n=1}^{N}(y(n) - \bar{y})^2}} \tag{11}$$

Sendo assim, a partir dos picos identificados, foi obtida a correlação em torno de cada batimento. Neste aspecto, foi obtido o coeficiente de correlação entre janelas de $25ms$ em torno do pico de detecção, sendo promediadas janelas que apresentavam índice de correlação acima de 99.5%. Um exemplo referente à correlação entre duas janelas de $25ms$ em torno do pico de detecção encontra-se representado na Figura 9.

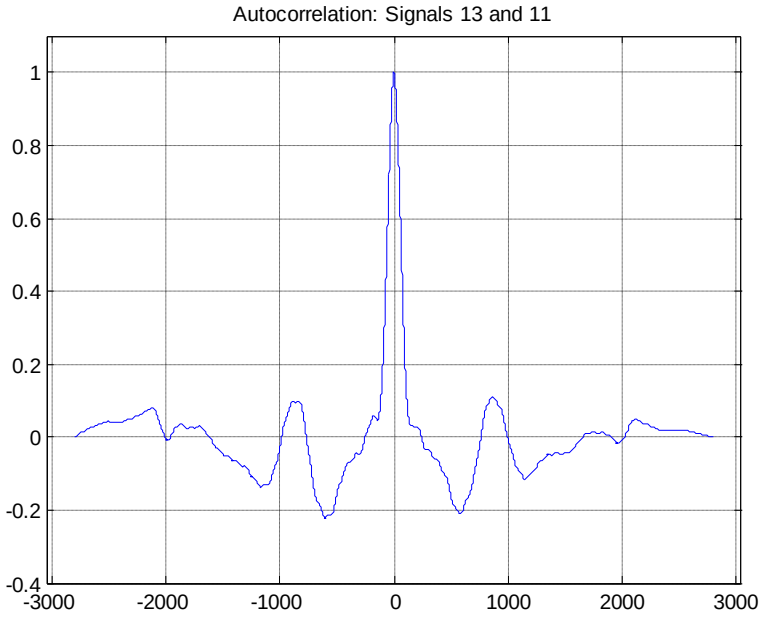


Figura 9. Correlação cruzada entre duas janelas com correlação maior que 99.5%.
(ECG01_5min@240Hz.dat)

Na Figura 10 encontra-se representado o conjunto de sinais não promediados referente à janela de 10s, separados em função do complexo PQRST.

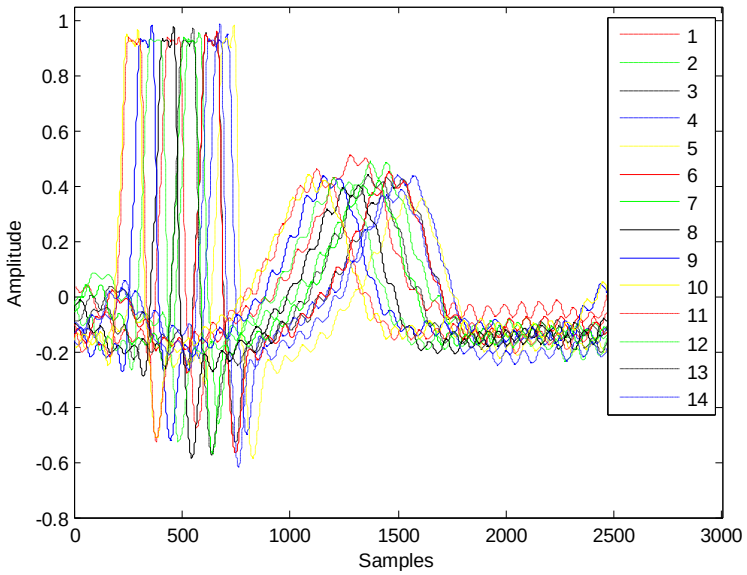


Figura 10. Conjunto de janelas de sinais Não Promediados (Janela de 10s).
(ECG01_5min@240Hz.dat)

Obtendo-se a correlação em torno do pico de detecção de batimento, e selecionando-se os sinais com coeficiente de correlação maior que 99.5%, é possível selecionar os sinais utilizados para promediação, conforme representado na Figura 11. Posteriormente é realizado a promediação dos sinais que apresentam maior correlação

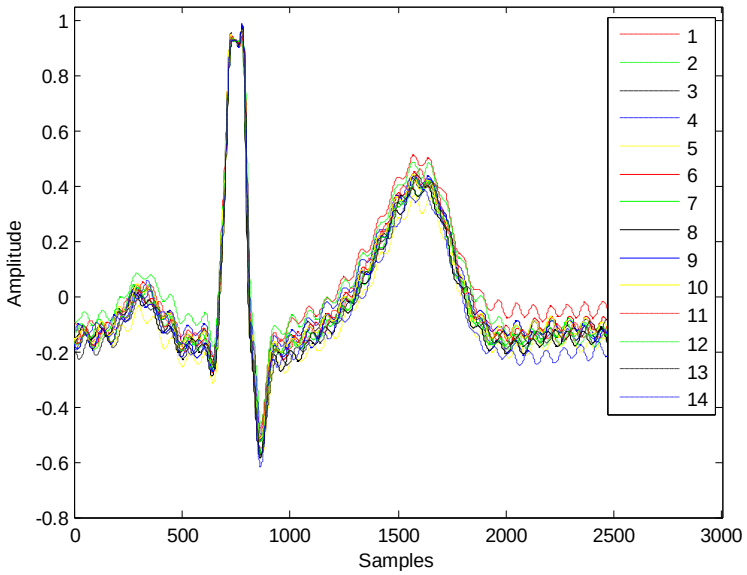


Figura 11. Complexos PQRST utilizados para promediação em uma janela de 10s.
(ECG01_5min@240Hz.dat)

A Figura 12 representa o sinal promediado.

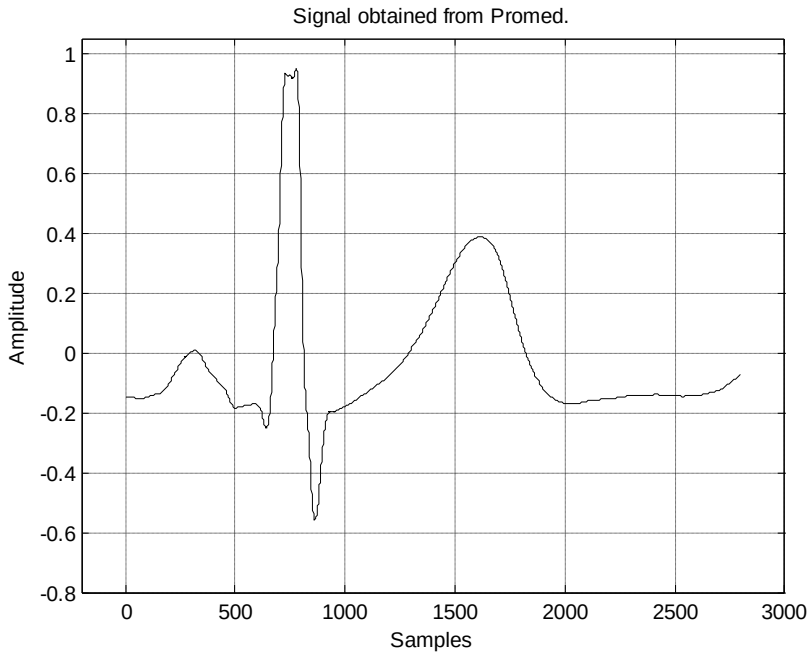
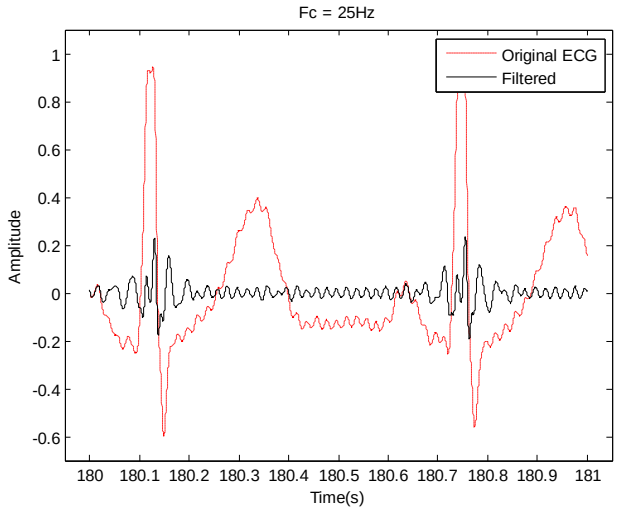


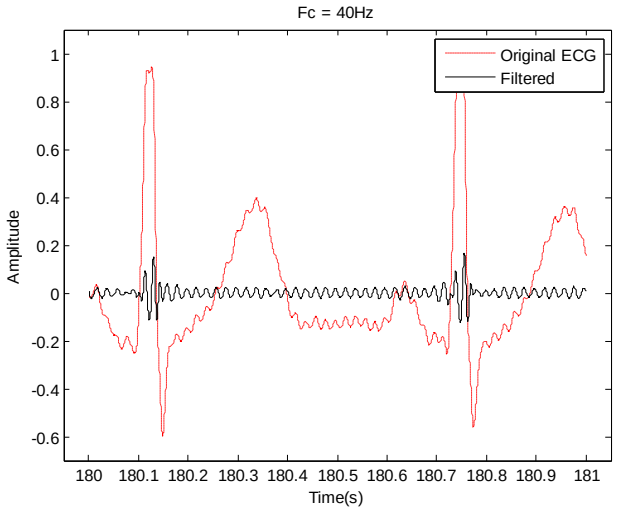
Figura 12. Sinal promediado. *(ECG01_5min@240Hz.dat)*

A partir dos sinais não promediados é possível estimar a o ruído residual presente no sinal. Para tanto foi calculada a variância do sinal do sinal em regiões as quais não ocorrera atividade cardíaca, sendo obtido um valor de 0.035, referente ao sinal *ECG01_5min@240Hz.dat*.

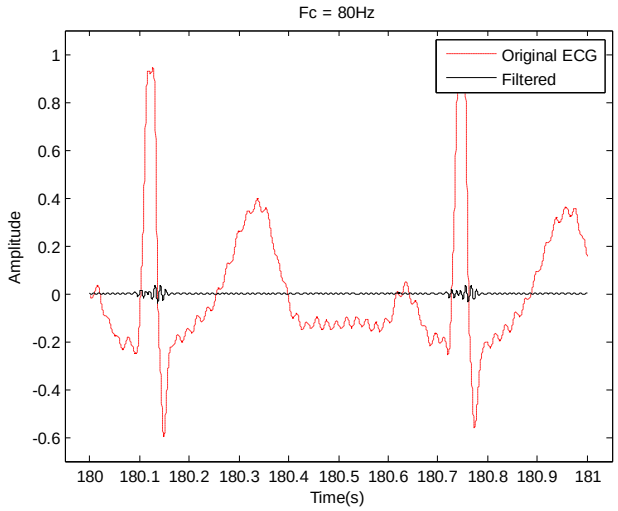
Ainda, foi obtido o vetor magnitude referente aos sinais processados por filtros passa-altas com frequências de corte em 25, 40 e $80Hz$, conforme representado na Figura 13. Foi implementado um filtro IIR(Butter) com 8 coeficientes.



(a) $Fc = 25Hz$



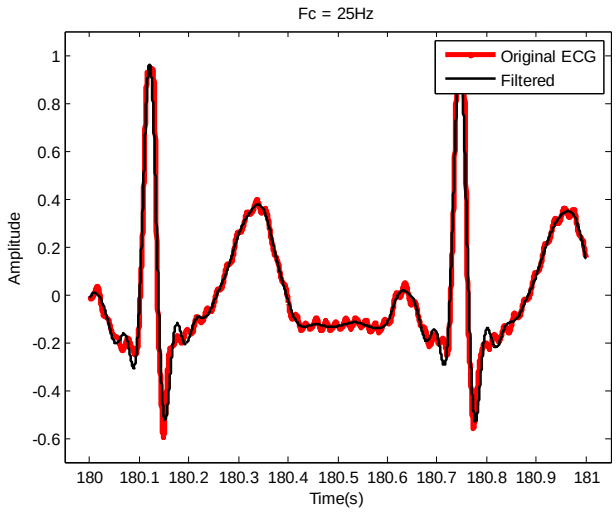
(b) $Fc = 40Hz$



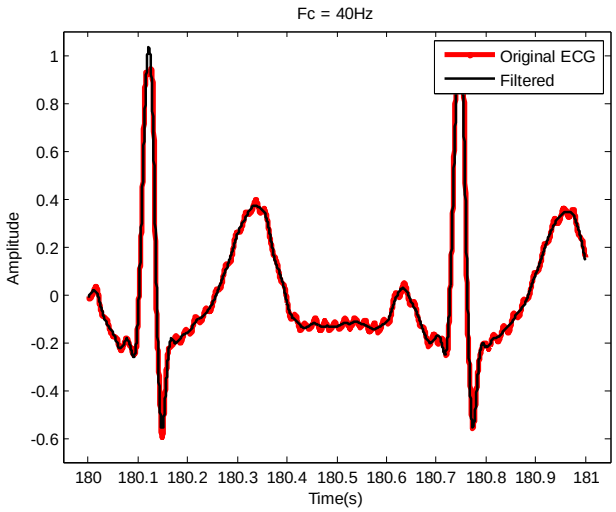
(c) $Fc = 80Hz$

Figura 13. Sinais submetidos à filtragem Passa-Alta. *(ECG01_5min@240Hz.dat)*

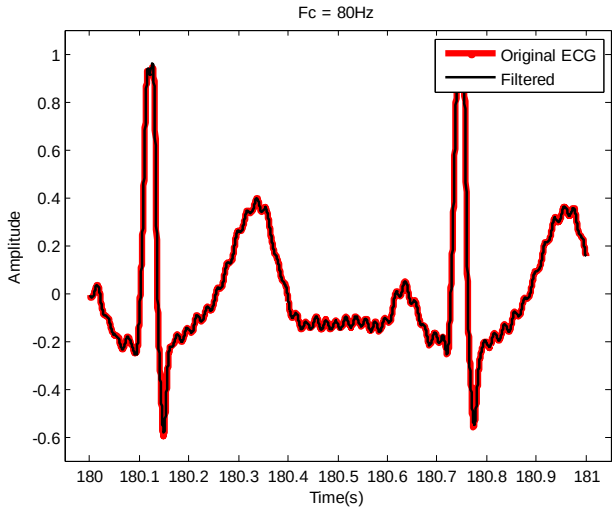
Ainda, na Figura 14 encontra-se o mesmo sinal submetido à filtragem passa-baixas.



(a) $Fc = 25Hz$



(b) $Fc = 40Hz$



(c) $Fc = 80Hz$

Figura 14. Sinais submetidos à filtragem Passa-Baixa. *(ECG01_5min@240Hz.dat)*

Em relação ao conjunto de sinais processados, uma vez obtido a relação de batimentos, foi determinada a relação entre as ondas RR, em ms, em função dos batimentos identificados.

Para identificação da distância RR foi computada a distância euclidiana entre dois picos R, previamente identificados através da estrutura descrita anteriormente.

Na Figura 15 encontra-se representado tal relação graficamente. No caso tratado, em um sinal de 5 minutos, foram identificados 468 batimentos, segundo a metodologia adotada.

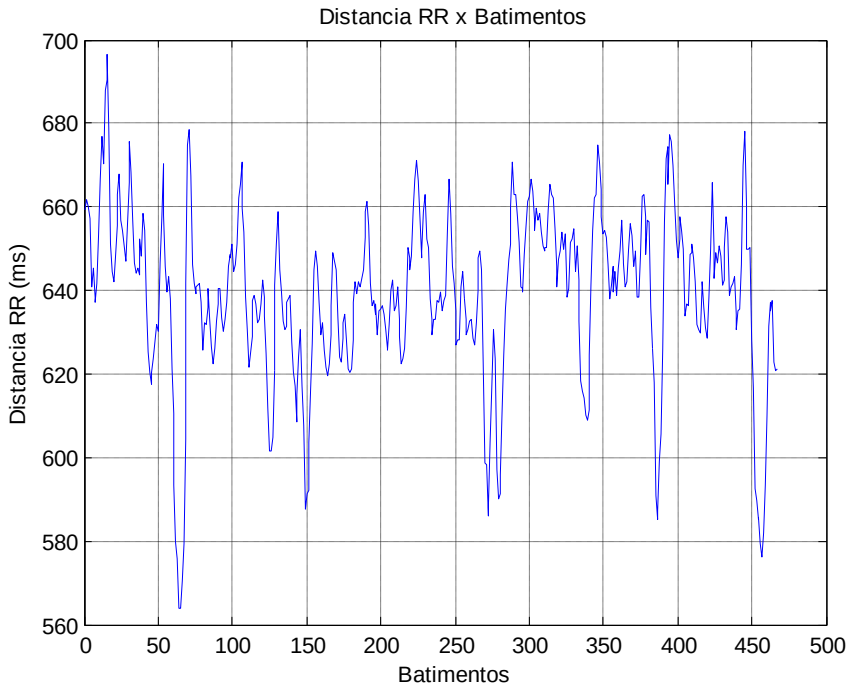
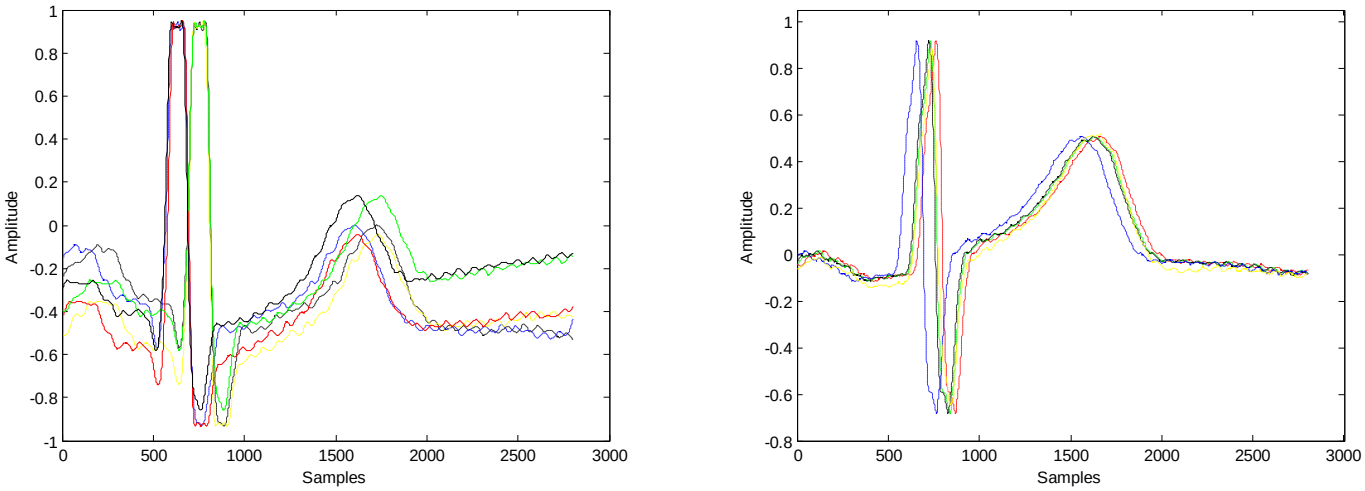


Figura 15. Relação entre distância RR e batimentos. *(ECG01_5min@240Hz.dat)*

Posteriormente foram aplicados os mesmos procedimentos metodológicos descritos aos sinais *ECG02_5min@240Hz.dat* e *ECGxyzData.dat*. Em relação ao conjunto de dados *ECGxyzData.dat*, o mesmo tratava-se de dois (3) canais: X, Y e lead. Foi obtida a média de ambos.

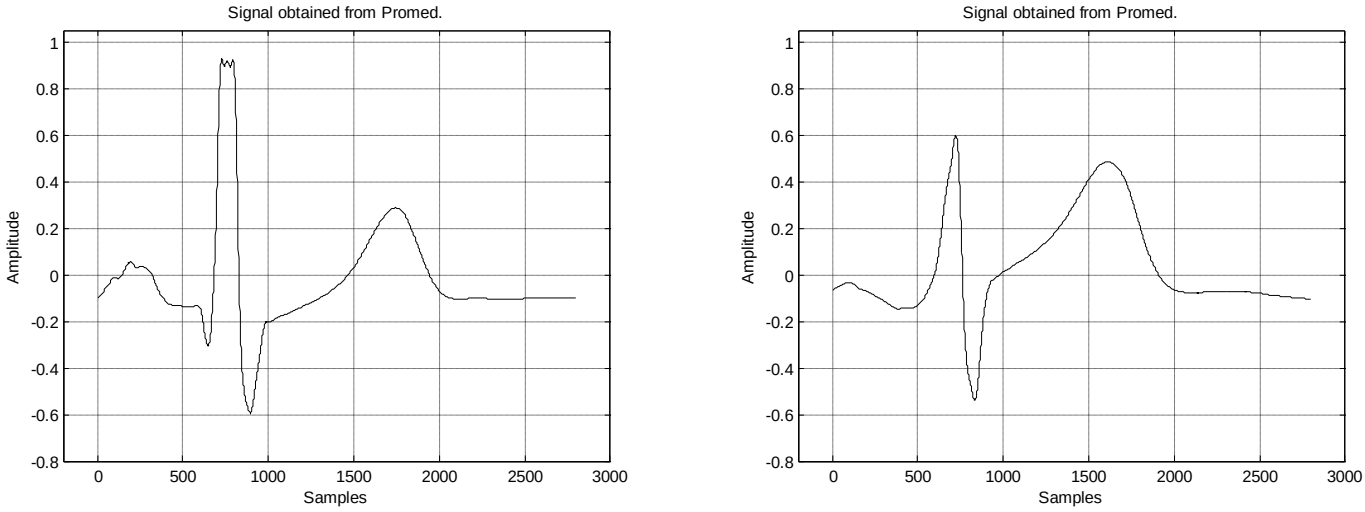
Na Figura 16 encontra-se representado um conjunto de sinais utilizados na promediação referente aos sinais *ECG02_5min@240Hz.dat* e *ECGxyzData.dat.*



(b) *ECGxyzData.dat*

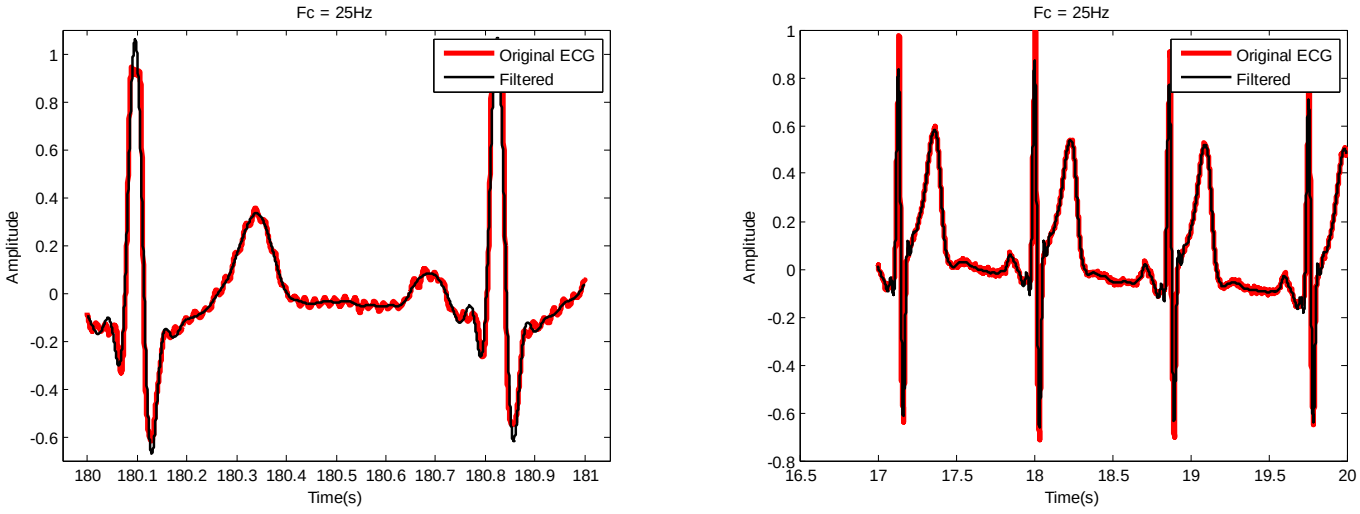
Figura 16. Amostras de complexos promediados

Em relação aos complexos selecionados para promediação, foram obtidos os sinais promediados, referente ao conjunto de dados *ECG02_5min@240Hz.dat* e *ECGxyzData.dat*, representado na Figura 17. A Figura 18 mostra um trecho de tais sinais submetidos à filtragem passa-baixa, com frequência de corte de 25Hz.



(a) *ECG02_5min@240Hz.dat* (b) *ECGxyzData.dat*

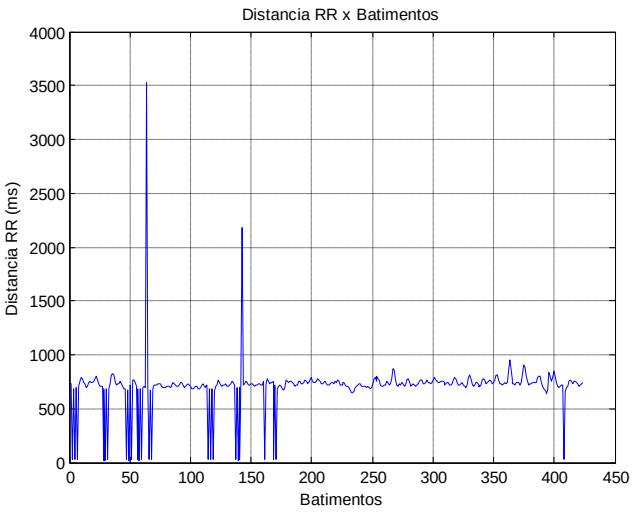
Figura 17. Sinais Promediados



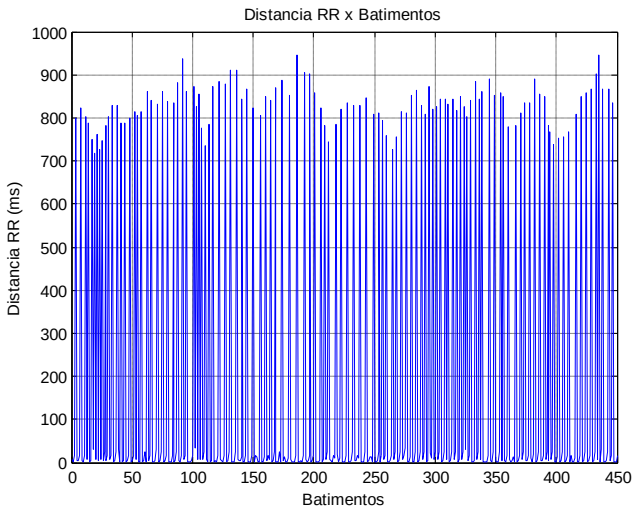
(a) *ECG02_5min@240Hz.dat* (b) *ECGxyzData.dat*

Figura 18. Sinais Filtrados (fc = 25Hz)

A relação da distância RR, em ms, em função da relação de batimentos encontra-se representado na Figura 19, novamente respectiva aos sinais mencionados anteriormente.



(a) *ECG02_5min@240Hz.dat*



(b) *ECGxyzData.dat*

Figura 19. Relação entre distância RR e batimentos

Por fim, na Tabela 2 encontra-se representado a relação dos complexos RR, QT e QTc referente aos sinais *ECG01_5min@240Hz.dat*(1), *ECG02_5min@240Hz.dat*(2) e *ECGxyzData.dat*(3). Para obtenção de tais características foi desenvolvido algoritmo, representado no fluxograma da Figura 20.

Para obtenção do complexo QTc foram utilizadas duas aproximações: Framingham (Equação 12) e Bazett (Equação 13).

$$QT_c = QT + 0.154(1 - RR) \tag{12}$$

$$QT_c = \frac{QT}{\sqrt{RR}} \tag{13}$$

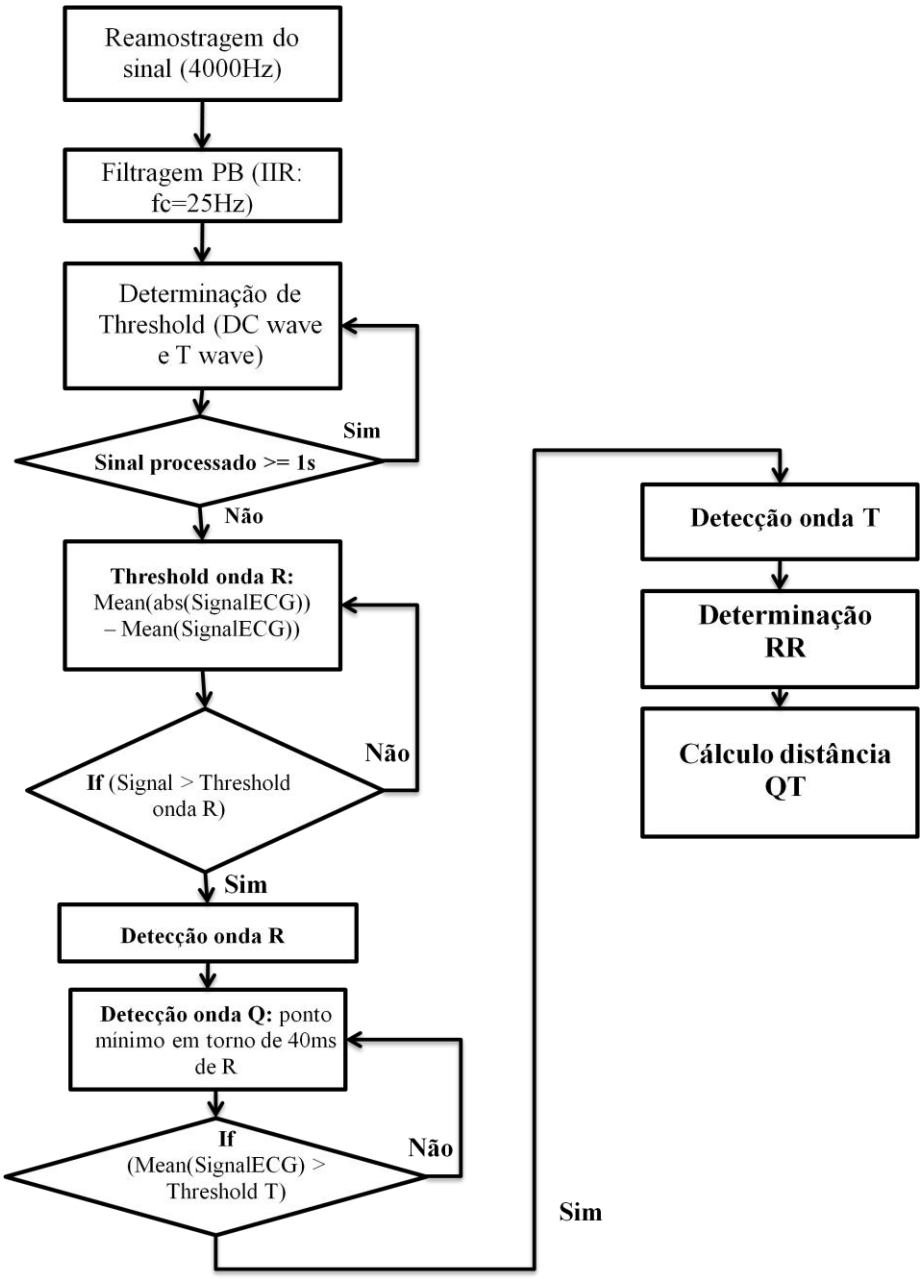


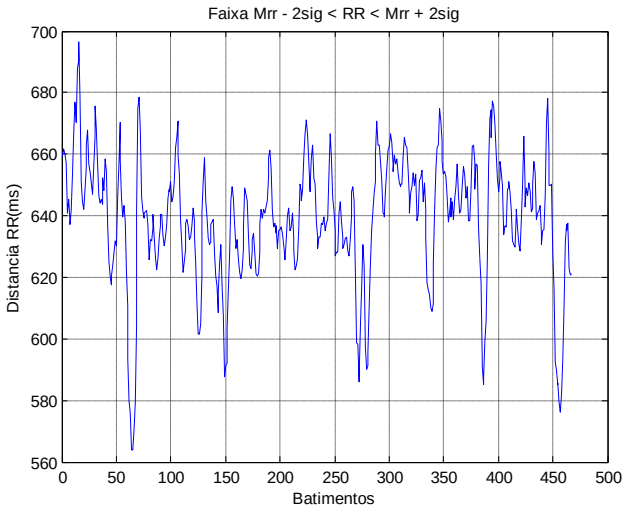
Figura 20. Software: Detecção distância QT

Tabela 2. Informações referentes aos complexos PQRST

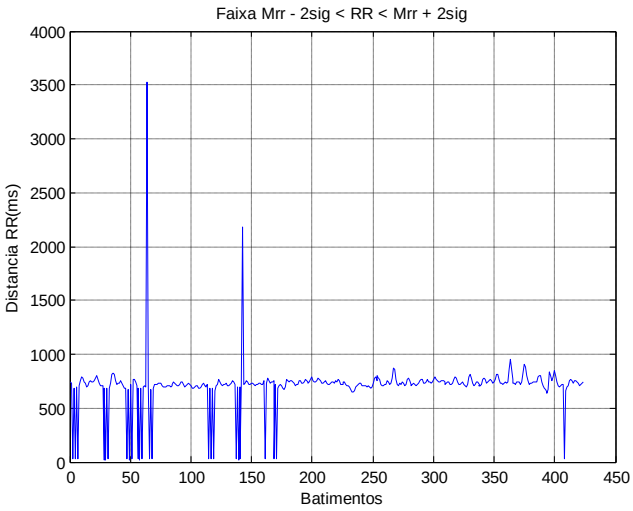
Signal	1	2	3 [(CH1+Ch2+Ch3)/3]
RR (Average)	636.2	700.5	217.0
QT (Average)	286.7	312.3	309.6
QTc (Bazett)	358.4	362.7	332.1
QTc (Framingham)	342.1	352.1	329.8

Ainda, em relação à identificação do segmento RR em função dos batimentos, foram obtidas as relações que obedeciam a faixa estipulada pela Equação (14). Na Figura 21 encontra-se representado as relações obtidas.

$$M_{RR} - 2\sigma < RR < M_{RR} + 2\sigma \tag{14}$$

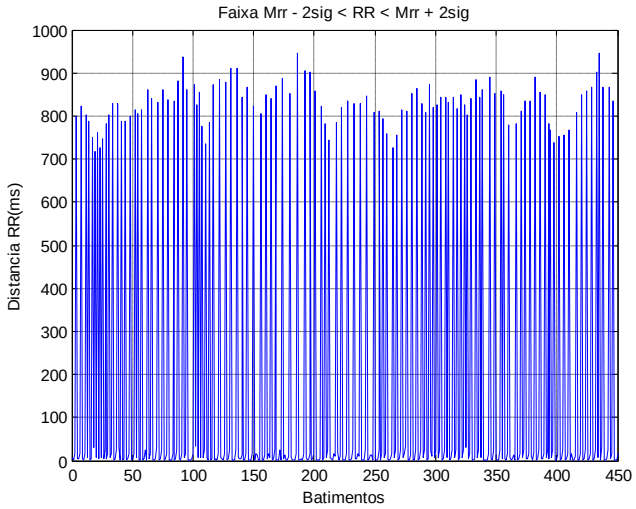


(a) RR (mean) = 638.9



(b) RR (mean) = 703.8

,



(c) RR (mean) = 218.0

Figura 1. RR em função dos batimentos. Calculado a partir do critério estabelecido na Equação 4. (a): *ECG01_5min@240Hz.dat*. (b) *ECG02_5min@240Hz.dat*. (c) *ECGxyzData.dat*

Foi observada pequena diferença entre os valores calculados sem a ponderação adotada.

Concluindo isso, deu-se início a análise dos resultados do filtro de média móvel e das estatísticas levantadas. Inicialmente pode-se observar nas Figuras 21, 22 e 23 o comportamento do filtro de média móvel com 100, 150 e 200 amostras, respectivamente, correspondendo também a intervalos de tempo de 0.025; 0.0375; e 0.05 utilizados pelo filtro de média móvel.

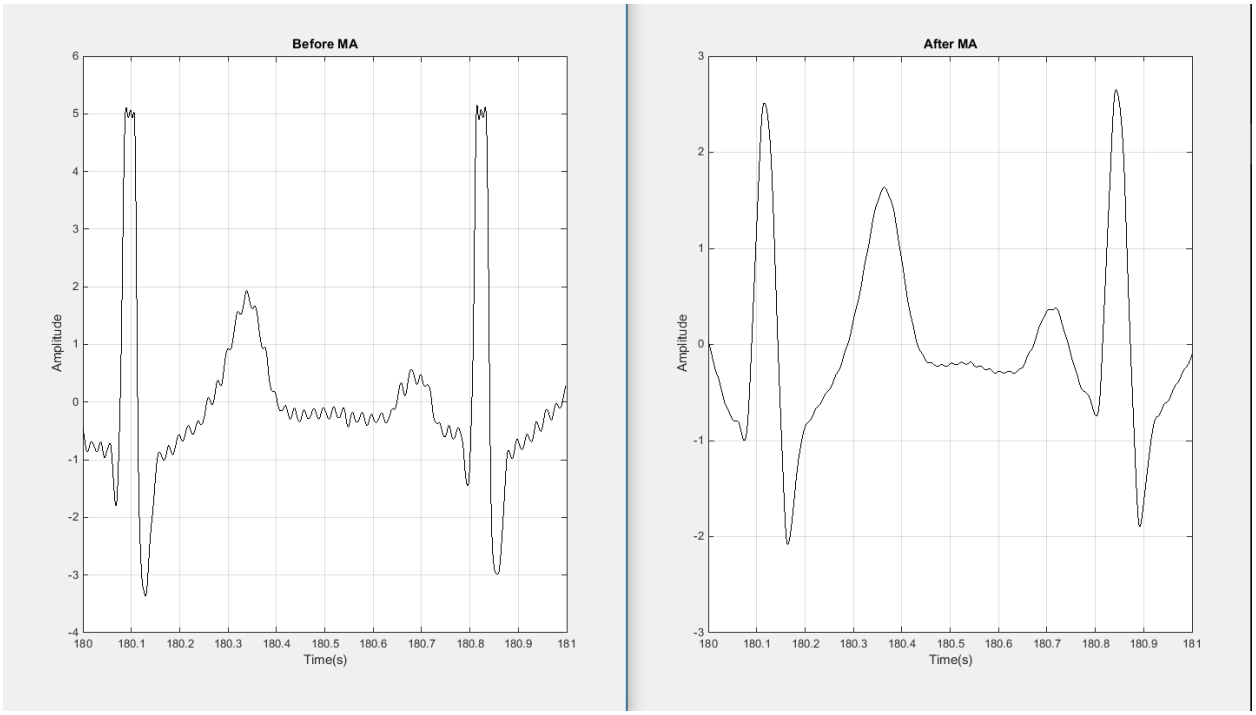


Figura 21 – Sinal *ECG02_5min@240Hz.dat* entre 180 e 181 segundos antes e depois do Filtro MA com 100 amostras.

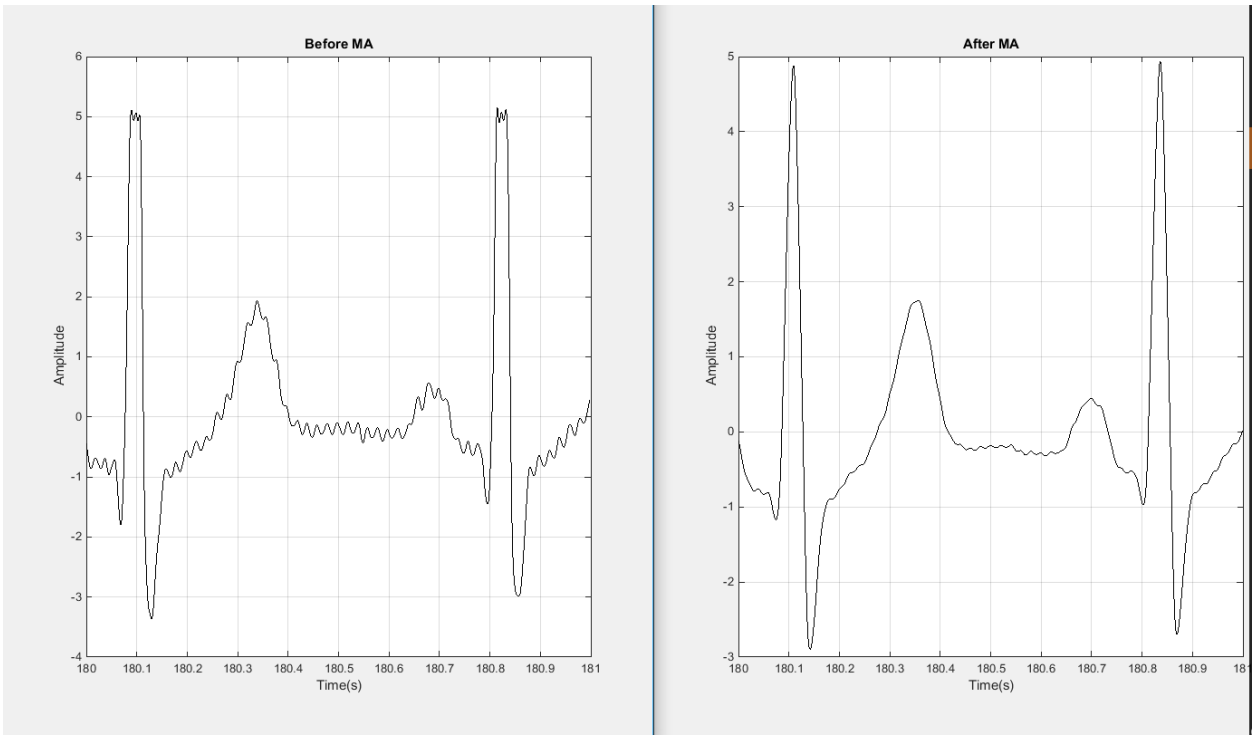


Figura 22 – Sinal *ECG02_5min@240Hz.dat* entre 180 e 181 segundos antes e depois do Filtro MA com 150 amostras.

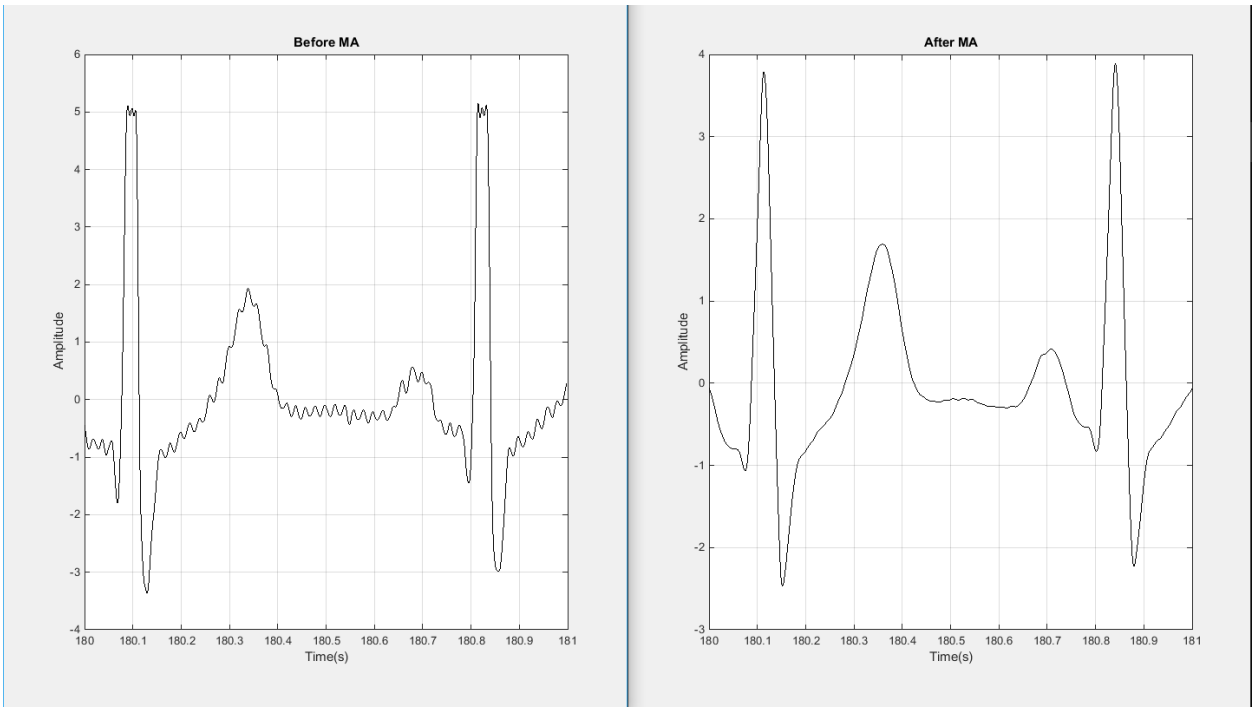


Figura 23 – Sinal *ECG02_5min@240Hz.dat* entre 180 e 181 segundos antes e depois do Filtro MA com 200 amostras.

Analisando-se o processo de filtragem por media móvel, pode-se observar a atenuação do sinal em termos de componentes de alta frequência. Comportamento este esperado. É possível observar também, a partir da Figura 22, cujo número de coeficientes do filtro é igual a 150, foram obtidos sinais mais coerentes considerando-se a amplitude do sinal original assim como em à eliminação de ruído, apresentado distorção mínima sobre o sinal desejado.

Para validar essa informação foram levantadas estatísticas do sinal antes e depois do filtro em cada conjunto de amostras de 1 minuto. Os resultados podem ser observados nas Tabelas 3, 4 e 5, trazendo resultados para os filtros com 100, 150 e 200 amostras respectivamente.

Tabela 3 – Estatísticas do sinal *ECG02_5min@240Hz.dat* antes e depois do filtro de média móvel com 150 amostras.

	Antes MA 1min	**Antes MA 2min**	**Antes MA 3min**	**Antes MA 4min**	**Antes MA 5min**	**Após MA 1min**	**Após MA 2min**	**Após MA 3min**	**Após MA 4min**	**Após MA 5min**
Variância	5,5583	3,6856	1,8397	1,6352	1,7468	5,1381	3,2658	1,4244	1,2329	1,3527
Desvio Padrão	2,3576	1,9198	1,3563	1,2787	1,3216	2,2667	1,8071	1,1935	1,1103	1,1631
RMS	2,3576	1,9235	1,3669	1,2860	1,3301	2,2667	1,8111	1,2055	1,1187	1,1726
Média	-0,0152	-0,1205	-0,1696	-0,1366	-0,1494	-0,0151	-0,1205	-0,1696	-0,1365	-0,1496

Tabela 4 – Estatísticas do sinal *ECG02_5min@240Hz.dat* antes e depois do filtro de média móvel com 150 amostras.

	Antes MA 1min	**Antes MA 2min**	**Antes MA 3min**	**Antes MA 4min**	**Antes MA 5min**	**Após MA 1min**	**Após MA 2min**	**Após MA 3min**	**Após MA 4min**	**Após MA 5min**
Variância	5,5583	3,6856	1,8397	1,6352	1,7468	4,8551	2,9768	1,1345	0,9519	1,0766
Desvio Padrão	2,3576	1,9198	1,3563	1,2787	1,3216	2,2034	1,7254	1,0651	0,9757	1,0376
RMS	2,3576	1,9235	1,3669	1,2860	1,3301	2,2034	1,7295	1,0785	0,9851	1,0483
Média	-0,0152	-0,1205	-0,1696	-0,1366	-0,1494	-0,0150	-0,1205	-0,1697	-0,1365	-0,1497

Tabela 5 – Estatísticas do sinal *ECG02_5min@240Hz.dat* antes e depois do filtro de média móvel com 200 amostras.

	Antes MA 1min	**Antes MA 2min**	**Antes MA 3min**	**Antes MA 4min**	**Antes MA 5min**	**Após MA 1min**	**Após MA 2min**	**Após MA 3min**	**Após MA 4min**	**Após MA 5min**
Variância	5,5583	3,6856	1,8397	1,6352	1,7468	4,6336	2,7489	0,9037	0,7281	0,8560
Desvio Padrão	2,3576	1,9198	1,3563	1,2787	1,3216	2,1525	1,6580	0,9506	0,8532	0,9252
RMS	2,3576	1,9235	1,3669	1,2860	1,3301	2,1526	1,6623	0,9657	0,8641	0,9372
Média	-0,0152	-0,1205	-0,1696	-0,1366	-0,1494	-0,0150	-0,1205	-0,1698	-0,1364	-0,1499

Destes resultados é possível concluir que conforme se aumenta o número de amostras do filtro de média móvel mais estacionário é o sinal. O sinal tende a perder altas frequências, aproximando-se cada vez mais de uma média nula sem variação, contendo apenas a componente DC do sinal. Também é possível observar que a média do sinal após o filtro praticamente não se altera, visto que o ruído é eliminado praticamente por completo.

5. Conclusão

A partir dos resultados obtidos é possível concluir a aplicabilidade relacionada à aplicação de conceitos de filtragem digital de sinais à sinais biomédicos, a partir de diferentes técnicas e algoritmos. Em termos gerais, facilita-se o processo de identificação de características específicas aos sinais, de forma rápida e segura.

É notório destacar a relação entre o custo computacional e a técnica utilizada, assim como em termos de qualidade sobre a resposta obtida. Em termos de veracidade e precisão, comparando-se as respostas obtidas pelos métodos empregados, pode-se concluir que, em média, as técnicas de filtragem IIR retornam sinais mais fidedignos quando comparados à promediação e média móvel.

Em termos de desempenho computacional, a promediação está diretamente relacionada ao número de termos promediados, de forma que, aumentando-se o número de sinais promediados, aumenta-se o tempo de processamento. Entretanto, aumenta-se o mesmo parâmetro, obtêm-se respostas melhores. A mesma característica foi observada pelo método da média móvel.

Sendo assim, é necessário relacionar um *trade-off* entre número de sinais utilizados para promediação, assim como termos utilizados para realização da média móvel, e o desempenho computacional da mesma, aliada à qualidade dos sinais filtrados obtidos.

6. Referências Bibliográficas

[1] Kohler, B.-U.; Hennig, C.; Orglmeister, R. The principles of software QRS detection. Engineering in Medicine and Biology Magazine IEEE, vol. 21, pp.42 – 57, January -February 2002.

[2] I. I. Christov. Real time electrocardiogram QRS detection using combined adaptive threshold. BioMedical Engineering OnLine, 2004. [cit: 2011-10-16]. [Online]. Available on internet: http://www.biomedical-engineering-online.com/content/3/1/28.

[3] Surda, J.; Lovas, S.; Pucik, J.; Jus, M. Spectral Properties of ECG Signal.Radioelektronika, 2007. 17th International Conference, Brno, Czech Republic, 24th– 25th April 2007, pp. 1 – 5.

[4] Piotrowskia Z.; Rózanowski K. Robust Algorithm for Heart Rate (HR) Detection and Heart Rate Variability (HRV) Estimation. ACTA PHYSICA POLONICA, vol. 118, pp. 131 – 135,No. 1/2010.

[5] Jacobson, A.L. Auto-threshold peak detection in physiological signals.23rd Annual International Conference of the IEEE, 25th– 28thOctober 2001, vol. 3, pp. 2194 – 2195.

[6] Irish Nurses and Midwires Organisation - INMO. Lisa Browne & Mary Mooney. https://www.inmo.ie/magazinearticle/printarticle/5926.

[7] Christense, B. (2014). Normal Electrocardiography (ECG) Intervals. Retrieved from http://emedicine.medscape.com/article/2172196-overview.

[8] The raw ECG signal processing and the detection of QRS complex. 2015 IEEE European Modelling Symposium. A. Peterkova, M. Stremy.

[9] Dimitris K Manolakis. Digital Signal Processing, 4th Edition. 2007

[10] Slides Professor

[11] Sufi F. Fang Q. Cosic I., ECG R-R Peak Detection on Mobile Phones, Conference of the IEEE EMBS, Agosto 2007

[12] Houghton A. e Gray D. Making Sense of the ECG 4th Edition, 2014

[13] Smith, Steven W. "The scientist and engineer's guide to digital signal processing." (1997).

[14] Wikimedia: Retrieved from https://commons.wikimedia.org/wiki/File:FIR_Filter_(Moving_Average).svg

7. Anexos

7.1. Código *main.m*. Script principal

```
clc
close all
clear all

%% Trabalho 1 - EEL510291 - PSD
%% 1. Interpolate signal to 4000Hz and plot

%%% Data 1
dataRead = 'ECG01_5min@240Hz.dat';
data1 = csvread(dataRead);
fs = 240;
[ecgit1]=interpolated(data1,fs,4000);

%%% Data 2
dataRead = 'ECG02_5min@240Hz.dat';
data2 = csvread(dataRead);
[ecgit2]=interpolated(data2,fs,4000);

%%% Data 3
dataRead = 'ECGxyzData.dat';
datax = csvread(dataRead);
datax =
(datax(1:round(end/3),2)+datax(1:round(end/3),3)+datax(1:round(end/3),4))/3;
% datax = (datax(1:end,2)+datax(1:end,3)+datax(1:end,4))/3;
fs = 1000;
[ecgit3]=interpolated(datax,fs,4000);



%% 2.
% 2.1: Detection of each beat/Promediate/Filter

    %%% Identificate peaks, using derivative method
    ecgit = ecgit1;   % Define which signal will be analysed
    fs = 4000;
    [pos, SigWdn] = IdentBeats(ecgit,fs);


    %%% Get cell representing separated signals and identified peaks
%     [SigPromed] = Promed(pos,SigWdn);
    [SigPromed] = Promed_faster(pos,SigWdn);


    %%% Define IIR parameter, then call filter function
    Type = 'Low';
    order = 8;
    fs = 4000;
    fc = 25;
    ECGIIRFilter(ecgit,fs,fc,order,'Low');
```

```
%% 3.
% Determinacao intervalos RR consecutivos (Grafico: RR x batimento)
    RRtimes = zeros(length(pos)+1,1);
    RRtimes(1)=0;

    for i=1:length(pos)-1
        time = pos(i+1)/4000 - pos(i)/4000;
        RRtimes(i) = time;
    end

    figure;
    plot(1000.*RRtimes(1:length(pos)-1))
    xlabel('Batimentos')
    ylabel('Distancia RR (ms)')
    title('Distancia RR x Batimentos')
    grid on

%% 6
%
    MeanRR = mean(RRtimes);
    stdRR = std(RRtimes);
    j = 1;
    for i=1:length(pos)-1
        time = pos(i+1)/4000 - pos(i)/4000;
        if((time < MeanRR + 2*(stdRR))||(time > MeanRR - 2*(stdRR)))
            posRRfine(j) = pos(i);
            RRtimeok(j) = time;
            j = j + 1;
        end
    end
    figure;
    plot(1000.*RRtimeok)
    xlabel('Batimentos')
    ylabel('Distancia RR(ms)')
    title('Faixa Mrr - 2sig < RR < Mrr + 2sig')
    grid on
%% 4

    %%% Function which returns QT, QTCc (Bazzet and Framingham method)
    [RR, RRMean, QT, QTB, QTF]=QTdetect(ecgit,fs-100);



%%
% Media movel - 5 e 6

% Leitura dos dados
    datax = csvread('ECG02_5min@240Hz.dat');
    fs = 240;

% Sinal
    tdata = 0:1/fs:(length(datax)-1)/fs;
    %figure;
    %plot(tdata,datax, 'k');
    %title('Sampling rate: 240Hz')
```

```
% Interpolacao: frequencia de amostragem de 240Hz para 4000Hz
    ecgit = resample(datax,4000,240);
    fsN = 4000;
    Nfs = round(4000/240);
    tecgit = 0:1/(fsN):(length(ecgit)-1)/(fsN);

% Sinal entre 3:00 and 3:10 minutos (180 - 190s)
% Intervalo de amostras: 4000samples/second*180seconds to
4000samples/second*190
    figure;
    plot(tecgit(1,(4000*180):(4000*181)),ecgit((4000*180):(4000*181),1),'k')
    xlabel('Time(s)')
    ylabel('Amplitude')
    title('Before MA')
    grid on;
    disp('Data without MA filter')

% Filtro de media movel
    MAecgit = Moving_Avarage_Filter(ecgit, 200);
    figure;

plot(tecgit(1,(4000*180):(4000*181)),MAecgit((4000*180):(4000*181),1),'k')
    xlabel('Time(s)')
    ylabel('Amplitude')
    title('After MA')
    grid on;
    disp('Data filtered')
% Separa sinal de ecg sem filtro em intervalos de 1 minuto
    minute_interval_samples = onemininterval(4000, ecgit, 5*60);
    MA_minute_interval_samples = onemininterval(4000, MAecgit, 5*60);
    figure('Name','Intervalos de 1 minuto do sinal nao filtrado');
    subplot(5, 1, 1);
    plot(tecgit(1,1:240000),minute_interval_samples(1,:),'g')
    subplot(5, 1, 2);
    plot(tecgit(1,1:240000),minute_interval_samples(2,:),'k')
    subplot(5, 1, 3);
    plot(tecgit(1,1:240000),minute_interval_samples(3,:),'r')
    subplot(5, 1, 4);
    plot(tecgit(1,1:240000),minute_interval_samples(4,:),'y')
    subplot(5, 1, 5);
    plot(tecgit(1,1:240000),minute_interval_samples(5,:),'b')
 % Separa sinal de ecg sem filtro em intervalos de 1 minuto
    figure('Name','Intervalos de 1 minuto do sinal filtrado');
    subplot(5, 1, 1);
    plot(tecgit(1,1:240000),MA_minute_interval_samples(1,:),'g')
    subplot(5, 1, 2);
    plot(tecgit(1,1:240000),MA_minute_interval_samples(2,:),'k')
    subplot(5, 1, 3);
    plot(tecgit(1,1:240000),MA_minute_interval_samples(3,:),'r')
    subplot(5, 1, 4);
    plot(tecgit(1,1:240000),MA_minute_interval_samples(4,:),'y')
    subplot(5, 1, 5);
    plot(tecgit(1,1:240000),MA_minute_interval_samples(5,:),'b')

% Calcula estatísticas de cada intervalo com e sem filtro
```

```
for i = 1:5
    Media(i) = mean(minute_interval_samples(i,:));
    Variancia(i) = var(minute_interval_samples(i,:));
    Desvio_Padrao(i) = std(minute_interval_samples(i,:));
    RMS(i) = rms(minute_interval_samples(i,:));
    MAMedia(i) = mean(MA_minute_interval_samples(i,:));
    MAVariancia(i) = var(MA_minute_interval_samples(i,:));
    MADesvio_Padrao(i) = std(MA_minute_interval_samples(i,:));
    MARMS(i) = rms(MA_minute_interval_samples(i,:));
end
```

7.2. Função *interpolated.m*. Função responsável pela reamostragem e normalização dos dados

```
function [ecgit]=interpolated(datax,fs,fsResampled)
% [ecgit]=interpolated(data,fs)
% input:        data (.dat)
% fs:           sampling frequency
% fsResampled:  resampled frequency
% ecgit:        ECG normalized/resampling to resampled frequency


% Show signal
    tdata = 0:1/fs:(length(datax)-1)/fs;
    figure;
    plot(tdata,datax, 'k');
    title('Sampling rate: 240Hz')
    xlabel('Time (s)')
    ylabel('Amplitude')

% Interpolating: fs to fsResampled
    ecgit = resample(datax, fsResampled, fs);
    tecgit = 0:1/(fsResampled):(length(ecgit)-1)/(fsResampled);

% Plot between 3:00 and 3:10 minutes (180 - 190 min)
% Normalize data
    ecgit = ecgit(:,:)./max(ecgit);
    figure;
%
plot(tecgit(1,(4000*180):(4000*190)),ecgit((4000*180):(4000*190),1),'k')
    plot(tecgit(1,(4000*18):(4000*28)),ecgit((4000*18):(4000*28),1),'k')
    title('Window: 180 - 190 min (4000Hz). Data normalized')
    xlabel('Time(s)')
    ylabel('Amplitude')
```

```
    grid on;
```

7.2. Função *IdentBeats.m*. Responsável pela identificação dos batimentos

```
function  [pos, SigWdn] = IdentBeats(ECGSignal,fs)
% [posBeats] = IdentBeats(ECGSignal)
% Input:
% ECGSignal: Signal
% fs:        Sampling frequency
% Output:
% pos:       Positions where occur beats
% SigWdn:    Separated signals

ecgit = ECGSignal;
tecgit = 0:1/(fs):(length(ecgit)-1)/(fs);

% Obtain derivation (window between 180 to 190s)
%     ecgDiff = diff(ecgit((4000*180):(4000*190),1).^2);
    ecgDiff = diff(ecgit((4000*28):(4000*38),1).^2);
    ecgDiffAll = diff(ecgit(:,1).^2);
%     ecgDiff= ecgDiffAll;

% Threshold
    th = abs(max(ecgDiffAll));
    th = th/2;

% Plot graph with derivation, also with the threshold used
    lineThAll = linspace(th,th,length(ecgDiffAll));
    figure;
    plot(tecgit(1,1:end-1),ecgDiffAll,'k');
    hold on
    plot(tecgit(1,1:end-1),lineThAll,'r--');
    grid on
    xlabel('Time(s)')
    ylabel('Amplitude')
    title('Derivate of the signal and threshold: All Signal')
    h = legend('Derivate of signal','Threshold = Max(Derivate)/2');
    set(h,'interpreter','none')

% Count/detect the number of peaks
    SigWdn = cell(length(ecgit),1);      % Storage windowed signals(for each
beat)
    StepWindow = 0.35;                   % trim signal over +/- 700ms around
peak
    bts = 0;                             % count beat
    FlagUp = 0;                          % Flag that counts if signal is
above the threshold
    posBeat = zeros(length(ecgDiff),1);  % Position where occur beat

    for i=1400:1:length(ecgDiffAll)-2100        % Start count after 1400
samples
```

```
        if((ecgDiffAll(i))>th)&&(FlagUp == 0)

            % Counter number of beats
            bts = bts+1;
            FlagUp = 1;

            % Create vector of positions which indicate where occur beat
            posBeat(i) = max(ecgit)-th;

            % Find when occur beat, then cut signal over 2800 samples between
the peak
            % Cut signal in windows
            % 4000Hz*700ms = 2800 samples
            SigWdn{i} = (ecgit(((i-700)):((i+2100)),1));

        end
        if((ecgDiffAll(i))<th)&&(FlagUp == 1)
            FlagUp = 0;
        end
    end

    % Get where occur beats
    pos = find(posBeat>0);


% Get positions where occur beat, then plot windowed signals
    figure;plot(SigWdn{pos(3),1},'k--');
    hold on;plot(SigWdn{pos(4),1},'b--');
    hold on;plot(SigWdn{pos(5),1},'y--');
%     hold on;plot(SigWdn{pos(6),1},'r');
%     hold on;plot(SigWdn{pos(7),1},'g');
%     hold on;plot(SigWdn{pos(8),1},'k');
    title('Sample of separated complex PRSQT signals')
    xlabel('Samples')
    ylabel('Amplitude')
```

7.3. Função *Promed.m*. Responsável pela promediação dos complexos PQRST

```
function [SignalPromediated] = Promed(posBeats,SigWdn)
% [SignalPromediated] = Promedation(posBeats,ECGSignal)
% Input:
% posBeats:              Identificated position of each beat
% SigWdn:                ECG separated signals (cell format)
% Output:
% SignalPromediated:     Promediated Signal

    pos = posBeats;

    SignalPromediates = cell(length(pos),length(pos));
```

```
        for i=1:length(pos)
            for j=1:length(pos)
                [COR,lagsCOR]=xcorr(SigWdn{pos(i),1},SigWdn{pos(j),1});
                COR = COR./max(COR);
                [posMaxCor, LAG] = max(abs(COR));
                if(posMaxCor > 0.995) % If correlation in 0 higher than 95%
                    % Coordinates where occur more correlation
                    SignalPromediates{i,j} = circshift(SigWdn{pos(i),1},
lagsCOR(LAG));
                end
            end
        end

% Promediate correlated signals
SigPromed = zeros(length(SigWdn{pos(1),1}),1);
SignalsPrm = cell(length(pos),1);
N = 0;
    for i=1:length(pos)
        for j=1:length(pos)
            if (isempty(SignalPromediates{i,j}) == 0) && (i~=j)
                SigPromed = SigPromed + SignalPromediates{i,j};
                SignalsPrm{i} = SignalPromediates{i,j};
                N = N+1;  % Number of signals used in promediation
            end
        end
    end

% Divide by the number of signals
    SigPromed = SigPromed./N;
    SignalPromediated = SigPromed;

% Promediate Signal
    figure;plot(SigPromed,'k')
    title('Signal obtained from Promed.')
    xlabel('Samples')
    ylabel('Amplitude')
    grid on
    hold on
    axis([-200 3001 -.8 1.05])
```

7.4. Função *QTdetect.m*. Responsável pela identificação das ondas Q e T

```
function [R_R, mean_RR_interval, QTavg, QTcBazett,
QTcFramingham]=QTdetect(ecg,fs)
% function [R_i,R_amp,S_i,S_amp,T_i,T_amp]=peakdetect(ecg,fs,view)
% QRS detection
% Detects Q , R and S waves,T Waves
% Uses the state-machine logic to determine peaks in ECG
% Inputs
% ecg : raw ecg vector
% fs : sampling frequency
% view : display span of the signal e.g. 8 seconds (8 seconds is the default)
% Outputs
```

```
% R_R: RR distance
% mean_RR_interval
% QTavg: QT interval (s)
% QT: Bazett method
% QT: Framingham method
% [R_R, mean_R_interval, QTavg, QTcBazett, QTcFramingham]=QTdetect(ecg,fs)

%% initialize - Inicialização
R_i = [];%index da onda R
R_amp = []; %Amplitude onda R
S_i = [];%save index of S wave
S_amp = []; %save amp of S wave
T_i = [];%save index of T wave
T_amp = [];%save amp of T wave
thres_p =[]; % Threshold adaptative
buffer_plot =[];
buffer_long=[]; % buffer for online processing
state = 0 ; % determines the state of the machine in the algorithm
c = 0; % counter to determine that the state-machine doesnt get stock in T
wave detection wave
T_on = 0; % counter showing for how many samples the signal stayed above T
wave threshold
T_on1=0; % counter to make sure its the real onset of T wave
S_on = 0; % counter to make sure its the real onset of S wave
sleep = 0; % counter that avoids the detection of several R waves in a short
time
S_amp1 = []; % buffer to set the adaptive T wave onset
buffer_base=[]; % Move average signal
dum = 0; % counter for detecting the exact R wave (CONTADOR DE PICOS R)
window = round(fs/25); % Tamanho medio da janela
weight = 1.8; %initial value of the weigth
co = 0; % T wave counter to come out of state after a certain time
thres2_p = []; % Threshold: wave T
thres_p_i = []; % Index: wave T
S_amp1_i = []; % Index: wave S
thres2_p_i = []; % Index: Threshold wave T
Q_i = []; % Index: wave Q
Q_amp =[]; % Vectors Q wave amplitude
%% preprocess (Filter IIR)
ecg = ecg (:); % make sure its a vector
ecg_raw =ecg; %take the raw signal for plotting later
time_scale = length(ecg_raw)/fs; % total time;
% Noise cancelation(Filtering)
f1=0.5; %cuttoff low frequency to get rid of baseline wander
f2=45; %cuttoff frequency to discard high frequency noise
Wn=[f1 f2]*2/fs; % cutt off based on fs
N = 3; % order of 3 less processing
[a,b] = butter(N,Wn); %bandpass filtering
ecg = filtfilt(a,b,ecg);

%% define two buffers
%%% Threshold of DC component
buffer_mean=mean(abs(ecg(1:2*fs)-mean(ecg(1:2*fs)))); % adaptive threshold DC
corrected (baseline removed)
%%% Threshold of T wave
buffer_T = mean(ecg(1:2*fs)); % second adaptive threshold to be used for T
```

```
wave detection

%% start online inference (Assuming the signal is being acquired online)
for i = 1 : length(ecg)

buffer_long = [buffer_long ecg(i)] ; % save the upcoming new samples
buffer_base = [buffer_base ecg(i)] ; % save the baseline samples

%% Renew the mean and adapt it to the signal after 1 second of processing
if length(buffer_base) >= 2*fs
    buffer_mean = mean(abs(buffer_base(1:2*fs)-mean(buffer_base(1:2*fs))));
    buffer_T = mean(buffer_base(1:2*fs));
    buffer_base =[];
end

%% smooth the signal by taking the average of 15 samples and add the new
upcoming samples
  if length(buffer_long)>= window % take a window with length 15 samples for
averaging
      mean_online = mean(buffer_long);  % take the mean
      buffer_plot =[buffer_plot mean_online]; % save the processed signal


    %% Enter state 1(putative R wave) as soon as that the mean exceeds the
double time of threshold
    if state == 0
     if length(buffer_plot) >= 3   %added to handle bugg for now
      if mean_online > buffer_mean*weight && buffer_plot(i-1-window) >
buffer_plot(i-window)    %2.4*buffer_mean
          state = 1; % entered R peak detection mode
          currentmax = buffer_plot(i-1-window);
          ind = i-1-window;
          thres_p = [thres_p buffer_mean*weight];
          thres_p_i = [thres_p_i ind];
      else
          state = 0;
      end
     end
    end

    %% Locate the R wave location by finding the highest local maxima
      if state == 1 % look for the highest peak

            if  currentmax > buffer_plot(i-window)
                dum = dum + 1; %%% Contador de picos (R)
                if dum > 4
                R_i = [R_i ind];%save index
                R_amp = [R_amp buffer_plot(ind)]; %save index

                % Locate Q wave (Localized in the minimum peak 40ms before
                % R peak
                [Q_tamp Q_ti] = min(buffer_plot(ind-round(0.040*fs):(ind)));
                Q_ti = ind-round(0.040*fs) + Q_ti -1;
                Q_i = [Q_i Q_ti];
                Q_amp = [Q_amp Q_tamp];
```

```
                if length(R_amp) > 8
                weight = 0.30*mean(R_amp(end-7:end)); %calculate the 35% of
the last 8 R waves
                weight = weight/buffer_mean;
                end
                state = 2; % enter S detection mode state 2
                dum = 0;
                end
            else
                dum = 0;
                state = 0;
            end

      end

    %% check weather the signal drops below the threshold to look for S wave
      if state == 2
        if  mean_online <= buffer_mean     % check the threshold
             state = 3;   %enter S detection
        end
      end

      %% Enter S wave detection state3 (S detection)
          if state == 3
            co = co + 1;

          if co < round(0.200*fs)
            if buffer_plot(i-window-1) <= buffer_plot(i-window) % see when
the slope changes
              S_on = S_on + 1; % set a counter to see if its a real change or
just noise
              if S_on >= round(0.0120*fs)
              S_i = [S_i i-window-4];%save index of S wave
              S_amp = [S_amp buffer_plot(i-window-4)];%save index
              S_amp1 = [S_amp1  buffer_plot(i-window-4)]; %ecg(i-4)
              S_amp1_i = [S_amp1_i ind]; %index of S_amp1_i
              state = 4; % enter T detection mode
              S_on = 0;
              co = 0;
              end
            end
          else
              state = 4;
              co = 0;
          end
          end

       %% enter state 4 possible T wave detection
       if state == 4
         if mean_online < buffer_mean % see if the signal drops below mean
           state = 6; % confirm
         end
       end
```

```
        %% Enter state 6 which is T wave possible detection
        if state ==6
          c = c + 1; % set a counter to exit the state if no T wave detected
after 0.3 second
          if c <= 0.7*fs

              % set a double threshold based on the last detected S wave and
              % baseline of the signal and look for T wave in between these
              % two threshold
              thres2 = ((abs(abs(buffer_T)-abs(S_amp1(end))))*3/4 +
S_amp1(end));
              thres2_p =[thres2_p thres2];
              thres2_p_i =[thres2_p_i ind];
              if mean_online > thres2
               T_on = T_on +1; % make sure it stays on for at least 3 samples
               if T_on >= round(0.0120*fs)
                if buffer_plot(i-window-1)>= buffer_plot(i-window)
                    T_on1 = T_on1+1; % make sure its a real slope change
                   if T_on1 > round(0.0320*fs)
                    T_i = [T_i i-window-11];%save index of T wave
                    T_amp = [T_amp  buffer_plot(i-window-11)];%save index
                    state = 5; % enter sleep mode
                    T_on = 0;
                    T_on1 = 0;
                   end

                end
               end
              end



          else
              state= 5; % enter Sleep mode
          end

        end


        %% this state is for avoiding the detection of a highly variate noise
or another peak
        % this avoids detection of two peaks R waves less than half a second
        if state==5
            sleep =sleep+c+1;
            c = 0;
            if sleep/fs >= 0.400
                state = 0;
                sleep = 0;%look for the next peak
            end
        end

       % update the online buffer by removing the oldest sample
       buffer_long(1)=[];
```

```
  end

end

%% conditions
R_R = diff(R_i); % calculate the distance between each R wave
heart_rate=length(R_i)/(time_scale/60); % calculate heart rate
% compute the min, max and average R-R wave
max_R_interval = max(R_R);
min_R_interval = min(R_R);
mean_RR_interval = (max_R_interval + min_R_interval)/2

%% QT difference
    %%% QT medio
    QTavg = 0;
    for i=1:length(T_i)
        QT = (T_i(1,i) - Q_i(1,i))/fs;
        QTavg = QTavg + QT;
    end
    %%% QT mean
    QTavg = QTavg/length(T_i)
    %%% QTc = QT/sqrt(RR) - Correcao Bazett
    QTcBazett = QTavg/sqrt(60/heart_rate)
    %%% QTc = QT/sqrt(RR) - Correcao Framingham
    QTcFramingham = QTavg + 0.154*(1-60/heart_rate)
```

7.5. Função *Moving_Average_FIlter.m*. Responsável pela realização do filtro de média móvel

```
function filteredSignal = Moving_Avarage_Filter(input, terms)
%%
%Essa funcao eh um filtro de media movel,
%Depende do sinal de entrada (input)
%Depende do numero de termos do filtro (terms)
%This function implements a move avarage filter
%input -> signal to be processed
%terms -> number of samples that are used

%%Creates the filter coeficients
filter_coef = ones(terms, 1) * (1/terms);

filteredSignal = filter(filter_coef, 1, input);

end
```

7.6. Função *onemininterval.m*. Responsável pela separação em trechos de 60s

```
function output = onemininterval(frequency, input, time)
%% Function to part a signal in time (seconds) domain into
%  60 seconds slots
%
% frequency -> signal sample rate
% input -> Signal to be processed
% time -> time in seconds that will be slotted

%inicialize output vector
    output = zeros(time/60, frequency*60);


%we have time/60 minutes, so we need i = time/60 lines
%frequency samples/second * seconds = sample
%each line receive its respectively sample sequency
    for i = 1:1:(time/60)
        for j = 1:1:(frequency*60)
            output(i,j) = input(j+((i-1)*frequency*60));
        end
    end
end
```