UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DIEGO SOGARI

Análise Comparativa de Métodos de Detecção Automática de Isquemias Cardíacas

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Prof. Dr. Valter Roesler
Orientador

Prof. Guilherme Lazzarotto de Lima
Coorientador

Porto Alegre, 11 de julho de 2014

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ECG	Eletrocardiograma
A/D	Conversão analógico-digital
SA	Sino-atrial
AV	Atrioventricular
DI ou D1	Primeira derivação de ECG dos membros
DII ou D2	Segunda derivação de ECG dos membros
DIII ou D3	Terceira derivação de ECG dos membros
MLI	Primeira derivação de ECG dos membros (sigla alternativa)
MLIII	Terceira derivação de ECG dos membros (sigla alternativa)
aVR	Derivação de ECG aumentada lateral direita
aVL	Derivação de ECG aumentada lateral esquerda
aVF	Derivação de ECG aumentada frontal
V1, . . . , V6	Derivações de ECG precordiiais
QRS	Complexo de ondas Q, R e S no ciclo cardíaco
ST	Tempo compreendido entre o fim do complexo QRS e o início da onda T
DTFT	*Discrete-time Fourier transform*
LIT	Linear e invariante no tempo
FIR	*Finite impulse response*
IIR	*Infinite impulse response*
BIBO	*Bounded input, bounded output*
MATLAB	*Matrix Laboratory*
SE	Sensibilidade
ES	Especificidade

PP	Preditividade positiva
PN	Preditividade negativa
AC	Acurácia
TF	Taxa de falha
VP	Verdadeiro positivo
VN	Verdadeiro negativo
FP	Falso positivo
FN	Falso negativo
PVC	*Premature ventricular contraction*
DFT	*Discrete Fourier trasnform*
RR	Intervalo entre dois batimentos, ou R-R
RMS	*Root Mean Square*
UFRGS	Universidade Federal do Rio Grande do Sul
FP	*Fiducial points*
FFT	*Fast Fourier transform*
API	*Application Programming Interface*
ST-T	União do segmento ST com a onda T
WFDB	*Waveform Database*

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

RESUMO

Esta monografia apresenta um estudo comparativo prático de três algoritmos de detecção de isquemia cardíaca em eletrocardiogramas. Todas as etapas de cada algoritmo – pré-processamento, extração de características e classificação –, são implementadas em MATLAB e nas linguagens C e C++.

O trabalho se propõe a atingir os seguintes objetivos: primeiramente inspirar-se nos algoritmos originais para criar uma implementação própria; em segundo lugar, pensa-se em unificar a implementação de modo que apenas a parte de extração (aquela que realmente define cada método) seja diferente; em terceiro lugar, quer-se avaliar o desempenho de cada método através de métricas de confiabilidade, como a sensibilidade e a preditividade positiva; por fim, deseja-se eleger o método que se sobressai como o mais confiável em termos destas métricas.

A razão de selecionar o método mais confiável é que, num momento futuro, poder-se-ia implementar o método de verdade num dispositivo do tipo *smartphone* ou mesmo num sistema de monitoramento cardíaco com suporte à tomada de decisão médica.

Este trabalho foi realizado em conjunto com o mestrado de Guilherme Lazarotto de Lima, Mestrando em Computação pela UFRGS.

Palavras-chaves: biomedicina. processamento de sinais. eletrocardiograma. isquemia cardíaca. redes neurais. monitoramento remoto.

ABSTRACT

This monograph presents a practical, comparative study of three algorithms for detection of myocardial ischemia in electrocardiograms. All stages in each of the algorithms – preprocessing, features extraction and classification –, are implemented in MATLAB and in the C and C++ programming languages.

The purpose of this work is to achieve the following goals: firstly, to draw on the original algorithms to create a similar, but own implementation; secondly, to unify the implementation in a manner such that only the extraction phase (the one that really defines each method) be different; then, it wishes to evaluate the performance of the methods through a reliability analysis, taking into account metrics like sensitivity and positive predictivity; ultimately, it is desired that the method which out-tops the others in terms of these metrics be elected as the most reliable.

The reason to select the most reliable method is that, in a future time, it could be effectively implemented in a *smartphone* or even in a cardiac monitoring system with support for medical decision-making.

This work was done in cooperation with Guilherme Lazarotto de Lima, in his research for the Master's degree in Computer Science at UFRGS.

Key-words: biomedicine. signal processing. electrocardiogram. myocardial ischemia. neural networks. telehomecare.

1 INTRODUÇÃO

O eletrocardiograma fornece aos especialistas da área de cardiologia informações de extrema relevância para o diagnóstico de doenças cardíacas como a isquemia. Ainda, por mais acurado que seja o diagnóstico do cardiologista, há situações em que se necessita de uma análise automatizada do eletrocardiograma.

Por exemplo, se um paciente está sob monitoramento contínuo através de um equipamento Holter, ele permanecerá com o equipamento por no mínimo 24 horas. Em tal situação, a duração do registro eletrocardiográfico é tamanha que o especialista dificilmente terá condições de analisá-lo na íntegra. Um computador deverá então processar o registro previamente e destacar áreas de interesse que possam ser inspecionadas com mais minúcia pelo cardiologista.

Outra situação em que se poderia fazer uso da tecnologia para auxiliar a tomada de decisão médica é no caso de competições esportivas. Frequentemente, atletas estão dispostos a ultrapassar seus limites na tentativa de quebrar um recorde, ou vencer uma competição. Nestes casos, é possível instalar nos atletas um dispositivo que registre a atividade elétrica do seu coração e envie os dados a uma central de monitoramento. Nesta central, um ou mais especialistas terão que analisar um volume enorme de eletrocardiogramas em tempo real, o que pode não ser factível. Aqui a automatização do processo de análise facilita a identificação de atletas que estejam à beira da exaustão e, sobretudo, de uma complicação cardíaca grave.

Estes são apenas alguns exemplos que ilustram a necessidade de automatizar a análise de eletrocardiogramas e levam, então, a que se estude métodos de detecção automática de patologias cardíacas. Notoriamente, entre estas patologias, está a isquemia.

Utilizando-se unicamente do eletrocardiograma como objeto sob análise e valendo-se de ferramentas matemáticas e computacionais capazes de extrair-lhe informações, diversos métodos foram propostos na literatura biomédica com vistas à detecção automática da isquemia cardíaca (ROCHA et al., 2010; MOHEBBI; MOGHADAM, 2007; GOPALAKRISHNAN; ACHARYA; MUGLER, 2004; PAPALOUKAS et al., 2001; GARCIA et al., 2000; GOLETSIS et al., 2004; AFSAR; ARIF, 2007; ANDREAO et al., 2004; BADILINI et al., 1992; MILOSAVLJEVIC; PETROVIC, 2006; PANG et al., 2005; STAMKOPOULOS et al., 1997). Dada uma especificação de sistema para realizar o procedimento de detecção, é importante escolher um método que se adeque às características do sistema e que apresente um bom desempenho uma vez implantado.

Neste capítulo será apresentada a motivação para o estudo desses métodos. Serão abordados conceitos relativos à isquemia cardíaca e o porquê de seu diagnóstico e tratamento serem tão importantes para a manutenção da saúde humana. Consecutivamente serão introduzidos os objetivos desta monografia e, finalmente, a sua estrutura.

1.1 Motivação

A isquemia cardíaca manifesta-se como consequência direta da doença coronariana, que é caracterizada pelo estreitamento das artérias coronárias. A causa principal desse estreitamento é a aterosclerose (doença inflamatória crônica). A palavra *isquemia* origina-se do grego e significa "constrição sanguínea". Em outras palavras, uma obstrução em uma artéria coronária causa falta de suprimento sanguíneo ao músculo do coração (miocárdio).

A falta de sangue no tecido cardíaco provoca má oxigenação do tecido e pode levar à morte das células (necrose) no local (DUBIN, 2000). Ao processo de necrose de parte do músculo cardíaco dá-se o nome de infarto do miocárdio (conhecido também como ataque cardíaco). Quando parte do tecido do miocárdio está sob a condição de infarto, ela deixa de responder aos estímulos elétricos do coração, assim como deixa de transmitir esses estímulos às áreas não afetadas. Dessa maneira, há um grande risco de parada cardíaca e morte nas pessoas acometidas de isquemia cardíaca.

Essa manifestação patológica pode ser silenciosa, sem evidência de sintomas, ou pode ainda causar dor no peito (conhecida como *angina pectoris*). Em algumas situações, isquemias desencadeiam ritmos cardíacos anormais (arritmias), que por sua vez podem levar a desmaios e até à morte súbita. Considerando os perigos apresentados pela isquemia cardíaca, seu diagnóstico precoce e o tratamento da doença coronariana são fundamentais para evitar consequências graves na saúde do paciente.

Monitorar constantemente o coração do paciente quando há suspeita de doença coronariana ou existência de isquemia cardíaca é uma medida muito importante a ser tomada para evitar complicações de maior gravidade. Geralmente o monitoramento é feito por um eletrocardiógrafo, em um procedimento chamado de eletrocardiograma. O eletrocardiograma, referido doravante como ECG, é um procedimento não-invasivo que registra a atividade elétrica do coração. Há mais de 80 anos o ECG é utilizado como base de diagnóstico de cardiopatias. O procedimento de aquisição do ECG é simples, barato e pode ser aplicado constantemente. O ECG pode ser utilizado tanto no diagnóstico de arritmias e isquemias, quanto no de outras doenças que afetam o coração direta ou indiretamente (FISCH, 2000), o que o torna extremamente versátil.

A saúde pública brasileira sofre de baixa disponibilidade de leitos e alta demanda de pacientes. O Ministério da Saúde estabelece que deva haver 2,5 leitos disponíveis ao SUS para cada 1000 habitantes (OLIVEIRA; CAVARARO, 2009), (BRASIL, 2002). Entretanto, este indicador (número de leitos por mil habitantes) apresenta um índice de 1,6 no Brasil, sendo 1,5 para a região Norte do país e um máximo de 1,9 para a região Sul. O que significa que existem muitas pessoas necessitando de atendimento, entretanto a rede pública não é capaz de atender a todos.

Este quadro apresenta inúmeras implicações, como as já conhecidas filas de espera grandes e atendimentos realizados em espaços inapropriados. Além disso, a baixa disponibilidade de leitos e alta demanda de pacientes implicam, comumente, na saída precoce de pacientes dos

hospitais. Nesta situação, o paciente acaba retornando às suas atividades sem estar completamente restabelecido, de modo que outro paciente, aparentemente necessitando de maiores cuidados médicos, possa usufruir de seu antigo leito.

Uma forma de resolver este problema é o uso do *homecare*, palavra oriunda da língua inglesa, significando "cuidado em domicílio". O homecare é uma alternativa ao tradicional atendimento em hospitais. Ao invés de ser internado em um hospital, o paciente recebe cuidados em casa, onde possui leito à sua disposição. Ademais, essa forma de atendimento reduz gastos públicos com internação e reduz a lotação de leitos hospitalares. Os avanços tecnológicos atuais, especialmente no campo das telecomunicações, permitem que se faça ainda um monitoramento remoto do paciente, ou seja, que os sinais do paciente possam ser analisados a partir de um local que não o do próprio paciente. O uso desse tipo de tecnologia na medicina, aliado ao homecare, caracteriza o *telehomecare* (AUGUSTYNIAK; TADEUSIEWICZ, 2009).

Finalmente, o contexto da saúde pública brasileira e o da saúde mundial demonstram que necessita-se cada vez mais de alternativas de atendimento e, além disso, de prevenção da doença coronariana. Uma boa alternativa é a utilização do *telehomecare*, para no caso das cardiopatias monitorar os sinais cardíacos de um paciente via ECG. Aliado a isso, métodos de análise automática de ECG podem acelerar e facilitar a tomada de decisão dos médicos e especialistas.

1.2 Objetivos

Dados três métodos de detecção de isquemia cardíaca propostos em artigos científicos, os objetivos deste trabalho são:

1. inspirar-se nas propostas dos autores para criar uma implementação própria

2. criar uma implementação unificada que sirva para teste dos três métodos

3. verificar o desempenho obtido para cada método em termos de métricas de confiabilidade, como sensibilidade e preditividade positiva

4. comparar os métodos entre si, e determinar aquele que se destaca como mais confiável

O fato de não se desejar reproduzir os experimentos originais de igual para igual vem do fato de que muitos detalhes de implementação de cada método não são expostos pelos autores. Assim, uma implementação própria – diferente da dos autores – é conveniente para avaliar a viabilidade dos métodos num sistema real.

1.3 Estrutura

Este trabalho está dividido logicamente em três partes: a primeira parte trata dos fundamentos necessários para a compreensão do conteúdo do trabalho e comporta os capítulos 2 ao 4; a segunda parte aborda os métodos de detecção de isquemia cardíaca e a implementação dos métodos, sendo composta dos capítulos 5 e 6; a terceira e última parte contém os capítulos 7 e 8, em que se discute os resultados dos testes e dá-se uma conclusão geral.

Primeiramente no capítulo 2 serão discutidos conceitos básicos de cardiologia necessários para a compreensão da tarefa de análise de ECGs. Em seguida, no capítulo 3, serão vistos tópicos sobre a representação e decomposição de sinais discretos, assim como a questão do projeto de filtros digitais, necessária para a realização da parte de pré-processamento dos métodos. O capítulo 4 introduz as métricas de avaliação utilizadas na comparação dos métodos.

O capítulo 5 dá uma visão geral sobre os métodos e procura categorizá-los de acordo com suas características comuns. O capítulo 6 fala sobre o projeto e a implementação de cada etapa dos métodos estudados. No capítulo 8 os resultados são apresentados, enquanto no capítulo **??** é dada conclusão.

2 NOÇÕES DE CARDIOLOGIA

Neste capítulo são apresentados alguns conceitos fundamentais de cardiologia para a interpretação de ECGs. Também são abordados os padrões do sinal de ECG que correspondem a uma determinada cardiopatia. O livro (DUBIN, 2000) foi utilizado como base para os estudos dos conceitos de cardiologia.

2.1 O Ciclo Cardíaco

O coração é o órgão principal do sistema cardiovascular humano, sendo que o ciclo cardíaco (também chamado batimento cardíaco), é o procedimento pelo qual o coração exerce a sua função de bombeamento de sangue neste sistema. O funcionamento do coração se deve a estímulos elétricos que atuam sobre as células do músculo cardíaco e fazem este se contrair. É através desta atividade elétrica que o sangue é impulsionado do coração para os pulmões, onde ocorre a troca entre gás carbônico e oxigênio, e para todo o resto do corpo, que necessita de oxigênio.

A atividade cardíaca pode ser resumida da seguinte forma. Primeiramente o coração encontra-se em um estado de repouso. Neste estado, as células do miocárdio têm seu interior carregado negativamente e diz-se que elas estão polarizadas (polaridade "-"). Quando estimuladas por um impulso elétrico, as células despolarizam-se (polaridade "+") e ficam então carregadas positivamente. O processo de despolarização faz as células e, por consequência, o miocárdio, se contraírem. Assim, a despolarização pode ser considerada como a progressão de uma onda de cargas positivas dentro das células, estimulando-as a se contrair.

Localizado no átrio direito, encontra-se o Nódulo Sino-Atrial (ou Nódulo SA) que é responsável pela produção dos estímulos elétricos de despolarização. Ele desencadeia uma propagação de cargas positivas ao longo do músculo cardíaco, provocando uma contração quase simultânea dos átrios direito e esquerdo. Esta fase é conhecida como **sístole atrial**. A sístole atrial impulsiona o sangue na direção dos ventrículos, que estão abaixo no eixo normal do coração. Quando, após um intervalo de aproximadamente 100 ms, a onda de despolarização atinge o Nódulo Atrioventricular (ou Nódulo AV), situado entre os átrios, este transmite o estímulo elétrico aos ventrículos por meio do Feixe Atrioventricular (ou Feixe de His). A pausa entre a emissão do estímulo pelo Nódulo SA e o seu recebimento pelo Nódulo AV é necessária para que o sangue entre nos ventrículos antes deste se contrair.

O Feixe de His é divido em dois ramos, Ramo Direito e Ramo Esquerdo. O Ramo Direito se estende por todo o Ventrículo Direito enquanto o Ramo Esquerdo, pelo Ventrículo Esquerdo. Ambos possuem ramificações chamadas fibras de Purkinje, que são os verdadeiros condutores do estímulo elétrico despolarizante. Tão logo a despolarização seja transmitida às células mio-

cárdicas, os ventrículos se contraem, caracterizando a fase conhecida como **sístole ventricular**. Após uma certa pausa, os ventrículos repolarizam-se, adquirindo novamente cargas negativas. A fase de repolarização é conhecida como **repouso entre batimentos**. A figura 2.1 ilustra este ciclo dentro da morfologia de um coração humano.

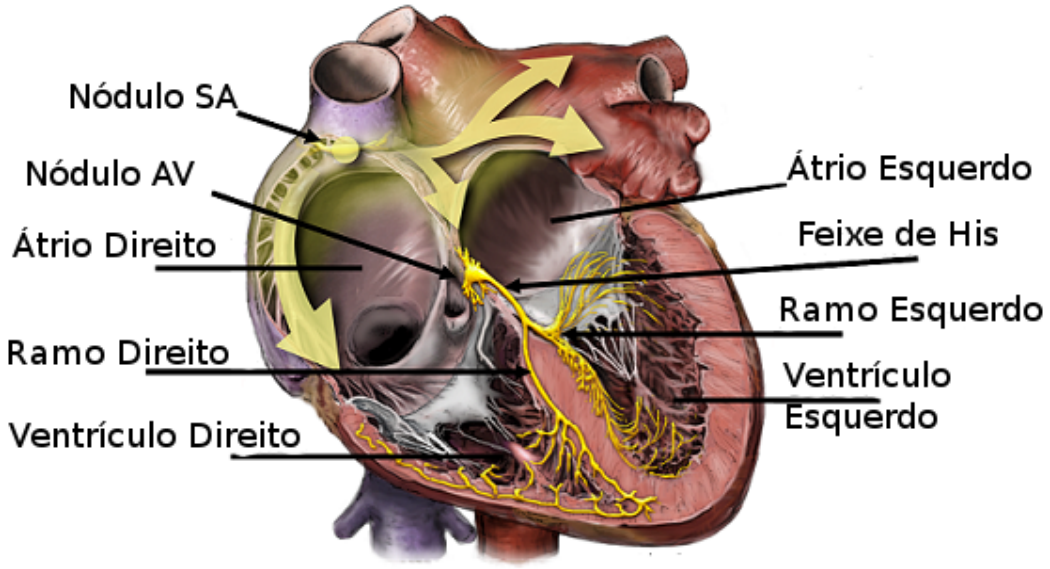


Figura 2.1 – Coração e sentido da despolarização. Adaptada de (CRILEY,).

2.2 Aquisição do eletrocardiograma

O ECG é um exame médico que registra a atividade elétrica do coração. As fases do ciclo cardíaco são captadas por um eletrocardiógrafo com o auxílio de eletrodos (ou sensores) cutâneos. Os eletrodos são colocados aos pares no corpo do paciente, de forma que cada configuração de posicionamento dos eletrodos representa uma **derivação**. Um ECG padrão possui doze derivações, seis delas precordiais (tórax) e seis periféricas (membros). Uma derivação é formada por um eletrodo positivo e um eletrodo negativo, sendo que o ponto de contato do eletrodo positivo define a derivação utilizada.

Quando uma onda despolarização (cargas positivas) se move na direção de um eletrodo positivo instalado na pele do paciente, registra-se uma deflexão positiva no ECG. No caso de um sistema de aquisição com papel milimetrado, a ponteira do eletrocardiógrafo se moverá para cima no traçado. Num sistema de aquisição digital, as amplitudes do sinal na saída do conversor analógico-digital (A/D) se afastarão do valor de referência do conversor no sentido positivo.

Para obter as derivações periféricas, coloca-se eletrodos nos braço direito e esquerdo e na perna esquerda, disposição esta que forma um triângulo, como pode ser verificado a partir da figura 2.2. A derivação I (DI) é composta por um par de eletrodos dispostos horizontalmente, com o pólo negativo colocado no braço direito e o positivo no braço esquerdo. Na derivação II (DII)

o pólo negativo também é colocado no braço direito, porém o positivo vai na perna esquerda. E, finalmente, tem-se a derivação III (DIII) em que o pólo positivo vai na perna esquerda e o negativo no braço esquerdo.

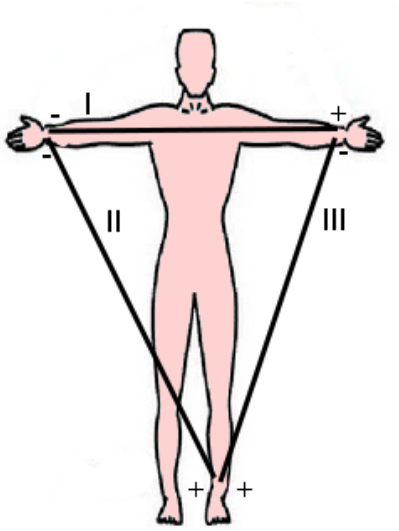


Figura 2.2 – Derivações periféricas: I, II e III.

As outras três derivações periféricas são as chamadas derivações aumentadas, quais sejam: aVR, aVF e aVL. A derivação aVR utiliza um eletrodo positivo no braço direito e um negativo comum (terra) dividido em três pontos de contato: um no braço esquerdo, um no pé esquerdo e outro no pé direito. A aVL é similar, porém com o pólo positivo no braço esquerdo e o negativo comum no braço direito, pé esquerdo e pé direito. Já a aVF é obtida com o pólo positivo no pé esquerdo e o negativo comum no braço esquerdo, braço direito e pé direito. Estas configurações podem ser visualizadas mais facilmente na figura 2.3.

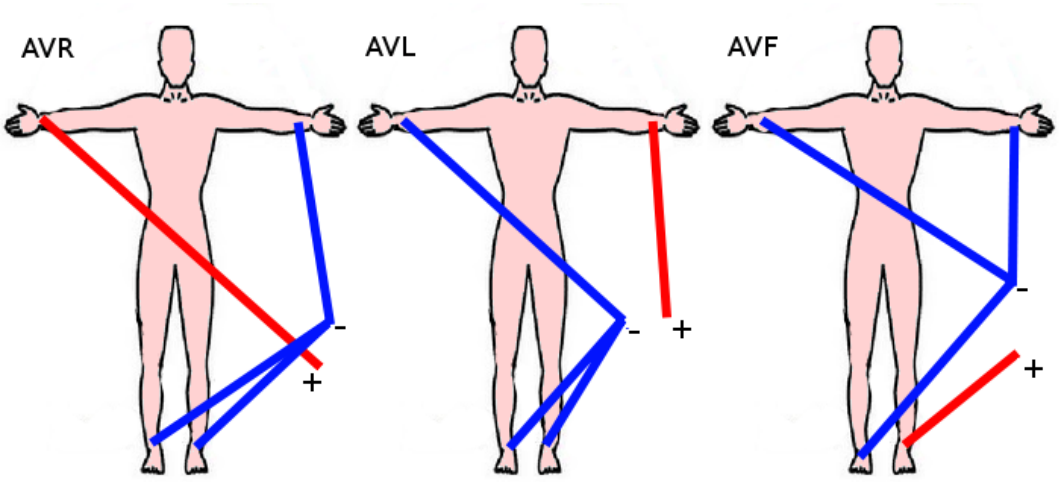


Figura 2.3 – Derivações periféricas aumentadas: aVR, aVL e aVF.

As derivações precordiais ou torácicas são obtidas de maneira diferente. Coloca-se eletrodos positivos na parte frontal do tórax, conforme ilustrado na figura 2.4. O dorso do paciente então é

considerado como o pólo negativo da derivação. Na figura 2.3 é possível ver que as derivações V1 e V2 estão localizadas no lado direito do coração; V3 e V4 localizam-se sobre o septo interventricular; por fim, V5 e V6 ficam sobre o lado esquerdo do coração. O intuito dessas derivações é de cobrir totalmente o coração em sua posição dentro da caixa torácica.

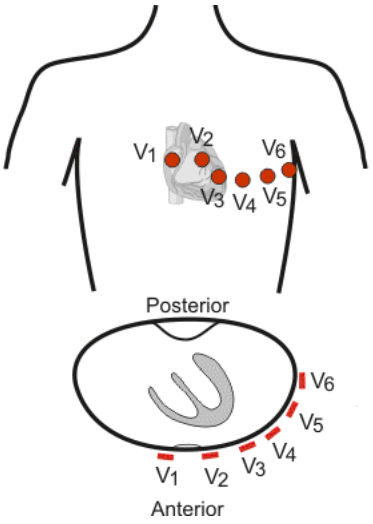


Figura 2.4 – Derivações precordiais: V_1, V_2, V_3, V_4, V_5 e V_6. Extraída de (KLABUNDE, 2008)

Na figura 2.5 observa-se um batimento típico registrado em uma derivação de eletrocardiograma. Cinco ondas se distinguem na figura, denominadas pelas letras P, Q, R, S e T. A onda P representa a deflexão causada pela contração atrial (despolarização dos átrios). Já as ondas Q, R e S formam o chamado **complexo QRS** e representam a despolarização ventricular. Por último, a onda T representa a repolarização ventricular. A repolarização atrial, contudo, dificilmente pode ser visualizada no ECG porque se mistura com o complexo QRS.

Ainda na figura 2.5, pode-se ver o chamado segmento PR, entre o fim da onda P e o início da onda Q, representando a pausa que existe entre a emissão do estímulo do Nódulo SA e o recebimento deste pelo Nódulo AV. O segmento ST, por sua vez, representa a pausa existente entre a contração ventricular e a repolarização ventricular. Há ainda o intervalo PR, que indica o tempo decorrido desde o início da despolarização atrial até o seu término. Finalmente, o intervalo QT indica o tempo entre o início da contração ventricular e o fim da repolarização ventricular.

2.3 Interpretação do eletrocardiograma

O ECG nos fornece informações valiosas a respeito da saúde física de um paciente. Em condições normais, um adulto apresenta morfologia de batimento cardíaco nos moldes da figura 2.5. Este padrão varia ligeiramente de acordo com a derivação escolhida, mas pode-se tomá-lo como referência na discussão que segue. Assim sendo, verá-se como uma doença cardíaca acarreta alterações no formato de ondas do ciclo cardíaco, quando analisado por meio do ECG.

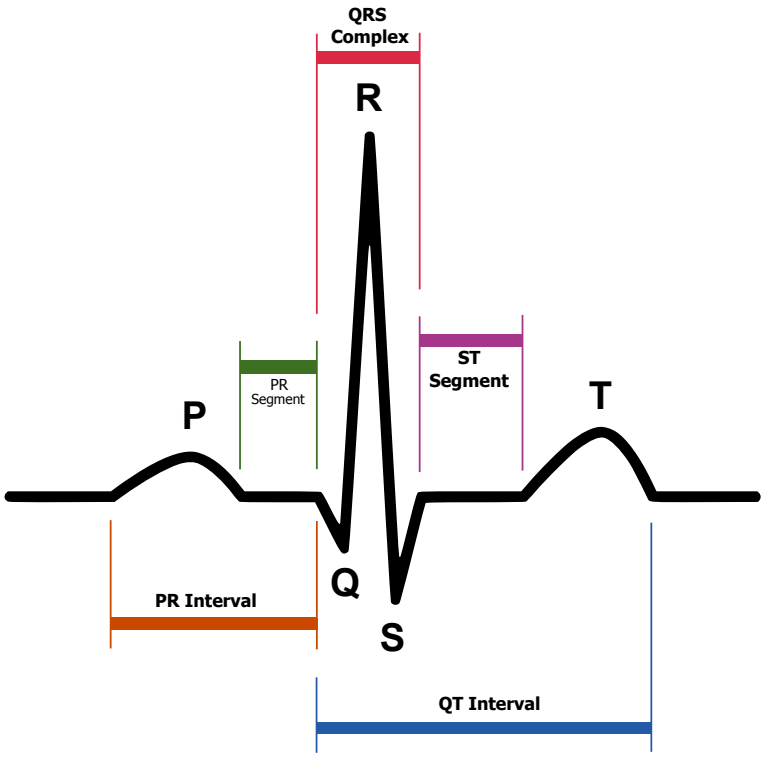


Figura 2.5 – Ciclo cardíaco típico em um eletrocardiograma. Extraída de (ATKIELSKI, 2007)

Ademais, informações como **frequência**, **ritmo**, **eixo**, **hipertrofia** e **infarto** podem ser avaliadas a partir do ECG.

Determina-se a frequência com que os batimentos ocorrem pela distância entre os mesmos no eixo temporal; pode-se então avaliar o ritmo dos batimentos como uma medida qualitativa da variância da frequência cardíaca. Diz-se que o ritmo é normal se a frequência se mantém constante ou quase constante ao longo do intervalo de observação. Caso contrário, diz-se que o ritmo é irregular e tem-se então as chamadas arritmias.

A direção com que se propaga a onda de despolarização do coração é chamada de eixo e é representada por um vetor, o vetor do estímulo elétrico. A partir da análise de várias derivações em conjunto pode-se identificar o eixo, bem como possíveis anormalidades a seu respeito. Também a existência de hipertrofias cardíacas, ou seja, aumento da massa muscular do coração, pode ser observada. Finalmente, é possível averiguar ocorrência de infarto, isto é, morte parcial do músculo cardíaco resultante da oclusão de uma artéria coronária.

Os métodos estudados neste trabalho detectam apenas isquemia cardíaca. Para entender como a isquemia se manifesta no ECG, é necessário conhecer um pouco do sistema cardiovascular e saber que o coração recebe sangue pelas artérias coronárias (figura 2.6). Quando, por algum motivo, um ramo da artéria coronária se estreita acentuadamente ou fica obstruído, a zona do miocárdio servida por esse ramo deixa de ter irrigação sanguínea adequada.

O ventrículo esquerdo é a cavidade mais espessa do coração, necessitando portanto de maior

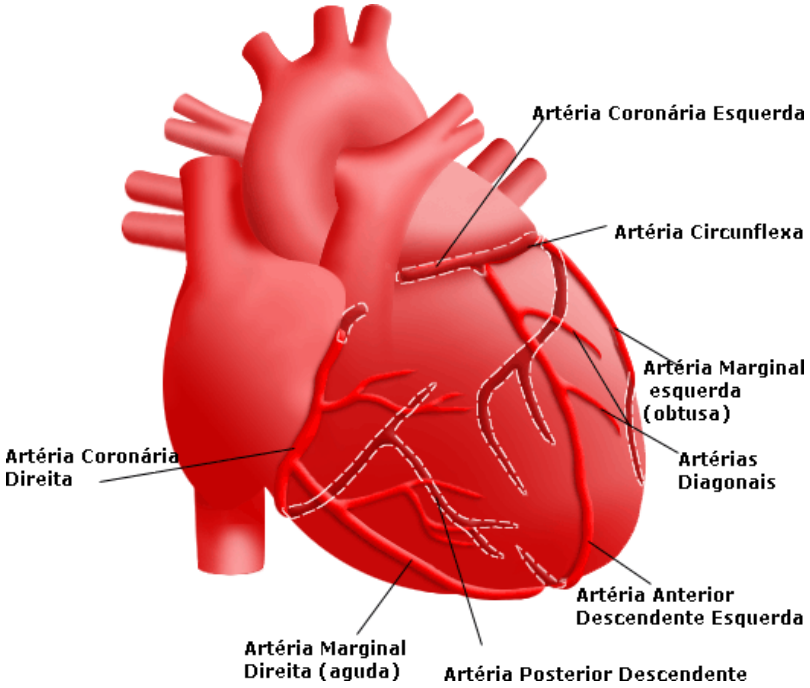


Figura 2.6 – Artérias coronárias. Extraído de (GROUP,)

quantidade de sangue. Assim, no caso de estreitamento das artérias coronárias, o ventrículo esquerdo é o primeiro a sofrer com a redução desse suprimento. É importante saber que o ventrículo esquerdo é em grande parte o responsável por enviar sangue para todas as partes do corpo. Quando parte do ventrículo esquerdo fica infartada, ela se torna eletricamente morta e não responde à despolarização. Consequentemente, ela não se contrai e acaba por prejudicar o bombeamento de sangue do coração. A figura 2.7 ilustra uma possível situação de infarto.

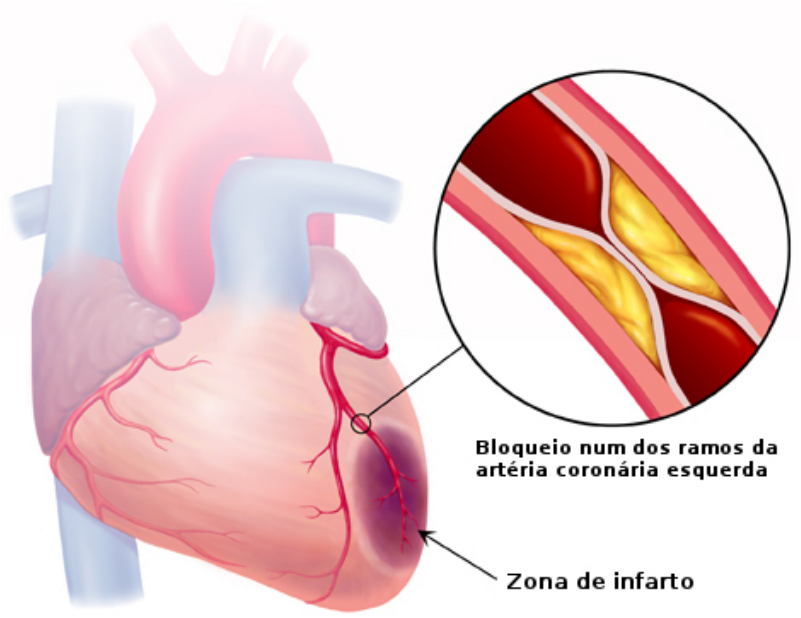


Figura 2.7 – Região de infarto do miocárdio. Extraído de (ANTIPUESTO, 2014)

A isquemia cardíaca é a redução do suprimento de sangue ou suprimento abaixo do normal.

Ela precede um possível infarto e pode ser identificada no ECG principalmente através de duas características (ou uma das duas): inversão da onda T, podendo variar desde uma onda achatada ou deprimida até uma inversão profunda; e desnivelamento do segmento ST, podendo ser um supradesnivelamento ou um infradesnivelamento.

Nem toda diminuição do suprimento sanguíneo produz um infarto. Ondas T invertidas podem indicar a existência de isquemia sem infarto do miocárdio. A onda T tipicamente isquêmica é simetricamente invertida. Como as derivações precordiais registram a atividade cardíaca em maior proximidade dos ventrículos, as alterações na onda T serão mais evidentes nessas derivações (V_1 a V_6). A figura 2.8 mostra possíveis variações morfológicas da onda T.

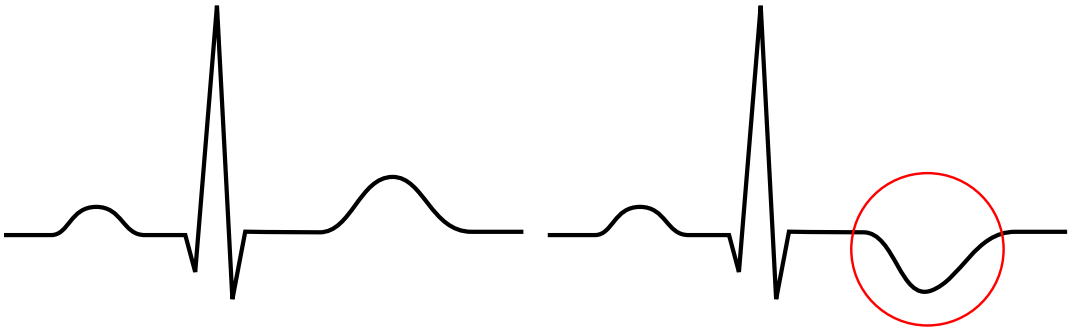


Figura 2.8 – Onda T normal e Onda T isquêmica.

A isquemia pode causar um lesão no tecido do miocárdio, caracterizando infarto agudo (recente). Para identificar a existência de lesão, analisa-se alteração no segmento ST. A elevação (supradesnivelamento) do segmento ST indica uma lesão e nos dá a certeza de que um infarto é agudo. Em certas situações, como Infarto Subendocárdico e Digitalis, o segmento ST apresenta uma depressão (infradesnivelamento). A figura 2.9 mostra as três variedades de nivelamento do segmento ST.

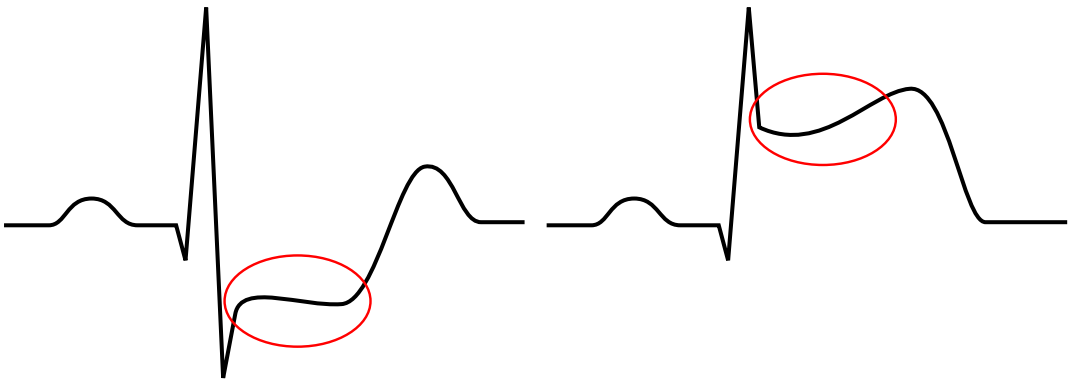


Figura 2.9 – Segmento ST infradesnivelado (1) e supradesnivelado (2).

2.4 Resumo

O ciclo cardíaco é um fenômeno fisiológico vital para os seres humanos. Essencialmente, tem-se que o Nódulo Sino-Atrial origina uma onda de despolarização (cargas positivas), consequente sístole atrial e, por meio do feixe de His, sístole ventricular. Após um período de repouso, átrios e ventrículos repolarizam-se (adquirem cargas negativas) e tornam-se capazes de iniciar um novo ciclo. É através desta atividade que o sangue circula em direção aos pulmões e para todo o resto do corpo, voltando ao coração após a troca gasosa e de nutrientes.

O exame de ECG registra as etapas do ciclo cardíaco por meio de derivações precordiais e/ou periféricas, formadas por um par de eletrodos cutâneos com cargas elétricas opostas. As derivações registram deflexões que representam cada uma das fases do ciclo, quais sejam: onda P, que é a deflexão causada pela contração atrial; as ondas Q, R e S, que formam o chamado complexo QRS e representam a contração ventricular; e a onda T, que representa a repolarização ventricular. A partir desse registro, é possível identificar informações como frequência cardíaca, ritmo, eixo, hipertrofia e infarto.

Especialmente, estamos interessados em identificar isquemia cardíaca, que é a redução do suprimento de sangue ao coração. A falta de irrigação sanguínea adequada ao coração, sobretudo ao ventrículo esquerdo, antecede um possível infarto e pode ser identificada pela ocorrência de inversão da onda T ou desnivelamento do segmento ST. É com base nessas informações que se consegue diagnosticar a isquemia a partir do ECG.

3 NOÇÕES DE PROCESSAMENTO DE SINAIS

Um pequeno ferramental matemático necessário para entender o tratamento de sinais digitais será apresentado neste capítulo. Num primeiro momento serão vistos conceitos como espaços, bases ortonormais e aproximação de vetores. Depois, a transformada de Fourier de tempo discreto e sua generalização, a transformada z, serão apresentadas. Em seguida, será feita uma breve discussão sobre a questão do projeto de filtros digitais. Por fim, serão discutidos os polinômios de Hermite e a expansão em funções de Hermite. Isto visa permitir um melhor entendimento dos próximos capítulos. Contudo, o leitor, se desejar, pode se referir apenas ao resumo no final do capítulo e prosseguir com a leitura do trabalho.

3.1 Representação de sinais discretos

Um sinal discreto $x[n]$ é uma sequência de valores em $\mathbb{R}$ ou $\mathbb{C}$, alocados em instantes de tempo discreto n. Normalmente, esta sequência representa as amplitudes geradas por um instrumento de medição sobre um processo físico qualquer, depois de amostradas e quantizadas por um conversor analógico-digital (A/D). Em particular, estamos interessados em processos reais (sem parte imaginária) e estocásticos, isto é, que não possuem comportamento previsível mas possuem propriedades estatísticas que os descrevem. Nas próximas subseções, verá-se como um sinal deste tipo pode ser decomposto em funções mais simples e conhecidas e também como se pode utilizar um conjunto finito e reduzido destas funções para aproximá-lo. Para um tratamento mais completo do assunto, sugere-se referir ao capítulo 2 do livro (VETTERLI; KOVACEVIC, 1995).

Espaços, bases e ortogonalidade

Na álgebra linear, costumamos tratar espaços vetoriais de dimensões finitas, como $\mathbb{R}^n$ ou $\mathbb{C}^n$. Neste contexto, dado um conjunto de vetores v_k, desejamos responder a perguntas como:

- o espaço em questão é *gerado* (*spanned*) pelo conjunto v_k? Em outras palavras, qualquer vetor no espaço pode ser representado por uma combinação linear de vetores de v_k?
- os vetores são linearmente independentes? Isto é, a afirmação de que um vetor em v_k pode ser escrito como combinação linear dos demais é falsa?
- como é possível encontrar bases e, em particular, bases ortonormais, que geram o espaço?
- por fim, dado um subespaço e um vetor qualquer, como podemos encontrar uma aproximação para esse vetor no subespaço?

Para respondê-las, utilizamos as noções de **norma** (ou tamanho) e **ortogonalidade** de vetores. Dados dois vetores x e y no espaço $\mathbb{R}^n$, temos:

- a norma quadrada: $\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$
- o produto escalar: $\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$
- a propriedade de ortogonalidade: $\langle x, y \rangle = 0$

Quando se trabalha com espaços infinitamente dimensionais, fala-se de uma extensão da álgebra linear, que são os espaços de Hilbert. Um espaço de Hilbert é um espaço vetorial de dimensões possivelmente infinitas, equipado com um operador **produto interno**. Podemos escrever o produto interno $\langle \cdot, \cdot \rangle$ para o caso de sequências reais como:

$$\langle x, y \rangle = \sum_{n=-\infty}^{\infty} x[n] y[n] \tag{3.1}$$

Este operador é linear, pois satisfaz as propriedades de aditividade $\langle x+y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$ e homogeneidade $\langle x, \alpha y \rangle = \langle \alpha x, y \rangle = \alpha \langle x, y \rangle$, para $\alpha \in \mathbb{R}$. Também satisfaz $\langle x, x \rangle \geq 0$, e $\langle x, x \rangle = 0$ se $x = 0$. Podemos usar o produto interno para definir normas e ortogonalidade. A norma de um vetor é escrita como $\|x\| = \sqrt{\langle x, x \rangle}$ e a distância entre dois vetores x e y, como a norma de sua diferença $\|x - y\|$. Podemos ainda dizer que x e y são ortogonais, $x \perp y$, se $\langle x, y \rangle = 0$.

Precisamos também definir o conceito de base. Um subconjunto $S = \{x_1, \ldots, x_n\}$ de um espaço vetorial E forma uma **base** para E se o espaço é gerado por S e se $x_i, \ldots, x_n$ são linearmente independentes. No caso de um espaço de dimensões infinitas, S deve conter infinitos vetores linearmente independentes. Por exemplo, o conjunto $\{\delta[n-k]\}_{k \in \mathbb{Z}}$, em que $\delta[n]$ é a função impulso discreto unitário, forma uma base para o espaço de sequências infinitas, pois possui infinitos vetores linearmente independentes que geram este espaço.

Um caso especial é quando os vetores da base são ortogonais entre si e possuem norma unitária. Diz-se então que a base é ortonormal. O procedimento para se encontrar uma base ortonormal a partir de um conjunto de vetores linearmente independentes (que geram o espaço de interesse) é chamado de ortogonalização de Gram-Schmidt. Essencialmente, tem-se

$$y_k = \frac{x_k - v_k}{\|x_k - v_k\|}, \tag{3.2}$$

onde $v_1 = 0$, e

$$v_k = \sum_{i=1}^{k-1} \langle y_i, x_k \rangle y_i, \quad k = 2, 3, \ldots \tag{3.3}$$

são as projeções sucessivas de um vetor do conjunto no subespaço gerado pelos vetores previamente ortogonalizados. Como será visto a seguir, esta expressão dá a melhor aproximação de um vetor no subespaço.

Aproximação por mínimos quadrados

Dado um vetor y num espaço E, deseja-se encontrar uma aproximação para este vetor que esteja num subespaço $S \subset E$. Assumindo que S possua uma base ortonormal $\{x_1, x_2, \ldots\}$, a projeção ortogonal de $y \in E$ em S é dada por

$$\hat{y} = \sum_i \langle x_i, y \rangle x_i. \tag{3.4}$$

A diferença $d = y - \hat{y}$ está no complemento ortogonal de S e, em particular, d é ortogonal a $\hat{y}$. Esta relação pode ser facilmente visualizada no espaço Euclideano tridimensional, conforme mostra a figura 3.1. Pela mesma figura, é possível ver que

$$\|y\|^2 = \|\hat{y}\|^2 + \|d\|^2. \tag{3.5}$$

A aproximação é ótima no sentido de mínimos quadrados, isto é, o valor $min\|y - x\|$ para x em S e $y = \sum_i \alpha_i x_i$ é obtido com

$$\alpha_i = \langle x_i, y \rangle. \tag{3.6}$$

Os α_i são chamados coeficientes da expansão de y em termos de x_i.

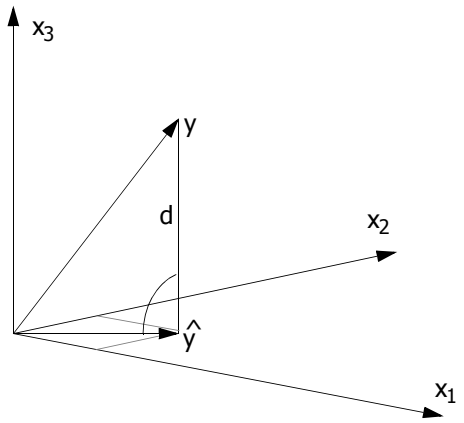


Figura 3.1 – Projeção no espaço tridimensional. Extraída de (VETTERLI; KOVACEVIC, 1995).

A seguir será introduzida uma metodologia de representação de sinais discretos que usa como base a sequência de polinômios de Hermite. Esta técnica será utilizada em dois dos métodos de detecção de isquemia, conforme será visto nos capítulos 5 e 6.

Polinômios de Hermite

Os polinômios de Hermite podem ser obtidos pela seguinte relação de recorrência:

$$\begin{aligned} &H_0(x) = 1 \\ &H_1(x) = 2x \\ &H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x), \quad n \geq 1 \end{aligned} \quad . \tag{3.7}$$

Ou podem ser expressos de forma explícita (não recursiva) por

$$H_n(x) = n! \sum_{m=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^m}{m!\,(n-2m)!} (2x)^{n-2m}. \tag{3.8}$$

Os polinômios de Hermite são ortogonais entre si com respeito à função e^{-x^2}. Tomando o produto interno por esta métrica, tem-se:

$$\int_{-\infty}^{\infty} H_m(x) H_n(x) e^{-x^2} dx = 0, \quad m \neq n \tag{3.9}$$

Note aqui o uso da integral em vez do somatório (ver equação 3.1), por conta da variável x ser contínua. Os primeiros seis polinômios de Hermite são descritos abaixo e têm seu gráfico mostrado na figura 3.2.

$$\begin{aligned} H_0(x) &= 1 \\ H_1(x) &= 2x \\ H_2(x) &= 4x^2 - 2 \\ H_3(x) &= 8x^3 - 12x \\ H_4(x) &= 16x^4 - 48x^2 + 12 \\ H_5(x) &= 32x^5 - 160x^3 + 120x \end{aligned} \tag{3.10}$$

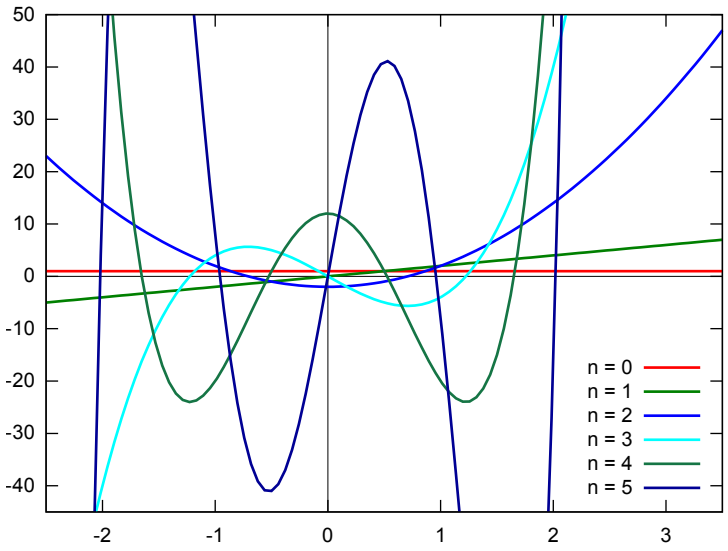


Figura 3.2 – Gráfico dos primeiros seis polinômios de Hermite. Extraído de (DAMATO, 2009).

Expansão em funções de Hermite

As funções de Hermite são definidas em termos dos polinômios de Hermite, de acordo com a equação 3.11 abaixo. A figura 3.3 mostra o gráfico das primeiras seis funções.

$$\psi_n(x) = \left(\frac{e^{-x^2}}{n! 2^n \sqrt{\pi}} \right)^{\frac{1}{2}} H_n(x) \tag{3.11}$$

De acordo com a teoria descrita em (ABRAMOWITZ, 1974), seguindo também o raciocínio introduzido para os polinômios de Hermite, essas funções são ortogonais entre si. Tomando o produto interno, tem-se:

$$\int_{-\infty}^{\infty} \psi_m(x)\,\psi_n(x)dx = 0, \quad m \neq n. \tag{3.12}$$

As funções de Hermite formam uma base ortonormal para o espaço de funções integráveis ao quadrado, $L^2(\mathbb{R})$. Portanto, têm aplicação direta na expansão de sinais desse tipo.

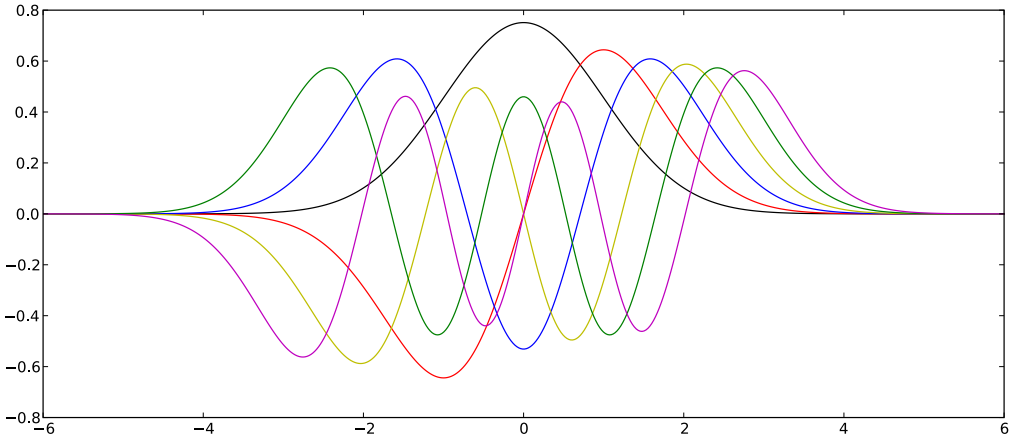


Figura 3.3 – Gráfico das primeiras seis funções de Hermite. Extraído de (DEMS, 2009).

No caso discreto, precisamos adotar uma abordagem um tanto diferente de cálculo dessas funções, que advém do fato delas serem autofunções da transformada de Fourier. Segundo a técnica descrita em (MUGLER; CLARY; WU, 2002), as funções discretas de Hermite são autovetores de uma matriz tridiagonal e simétrica T_b. Esta matriz é completamente especificada por sua diagonal principal e diagonal secundária, cujos elementos são dados por

$$d_{0,i} = -2\cos(N\pi\tau)\sin(i\pi\tau)\sin((N-i-1)\pi\tau), \quad 0 \leq i \leq N-1 \tag{3.13}$$

e

$$d_{1,k} = \sin(k\pi\tau)\sin((N-k)\pi\tau), \quad 1 \leq k \leq N-1, \tag{3.14}$$

onde $\tau = \frac{1}{Nb^2}$, b é um parâmetro de dilatação e N é o tamanho do sinal. O efeito do parâmetro b é um alargamento das funções de Hermite no eixo temporal, à medida que b aumenta.

Os autores advogam que o maior autovalor da matriz T_b (num ordenamento do maior para o menor), corresponde à primeira função discreta de Hermite; o segundo maior autovalor, à segunda função de Hermite e assim por diante. Logo, o algoritmo de cálculo das funções discretas de Hermite consiste em construir a matriz T_b conforme as equações 3.13 e 3.14, extrair-lhe os autovetores e autovalores e, finalmente, enumerar os autovetores de acordo com seus respectivos autovalores em ordem decrescente. Uma vez normalizados (norma unitária), estes vetores formam uma base ortonormal para o espaço de sequências somáveis ao quadrado, $L^2(\mathbb{Z})$.

A expansão de $x[n]$, com $0 \leq n \leq N-1$, por funções de Hermite se dá pelo produto interno de x com cada um dos vetores previamente calculados. Podemos ainda construir uma matriz U tendo esses vetores como linhas. Então a expansão se resume a um produto matricial Ux. Note que x, assim como U, possui dimensões finitas e, portanto, esta operação é na verdade uma aproximação. Em outras palavras, ela é a projeção de x no subespaço gerado pelas linhas de U. A figura 3.4 ilustra em linha pontilhada a aproximação de um sinal discreto (representado, por conveniência, usando linha contínua) usando as 50 primeiras funções discretas de Hermite.

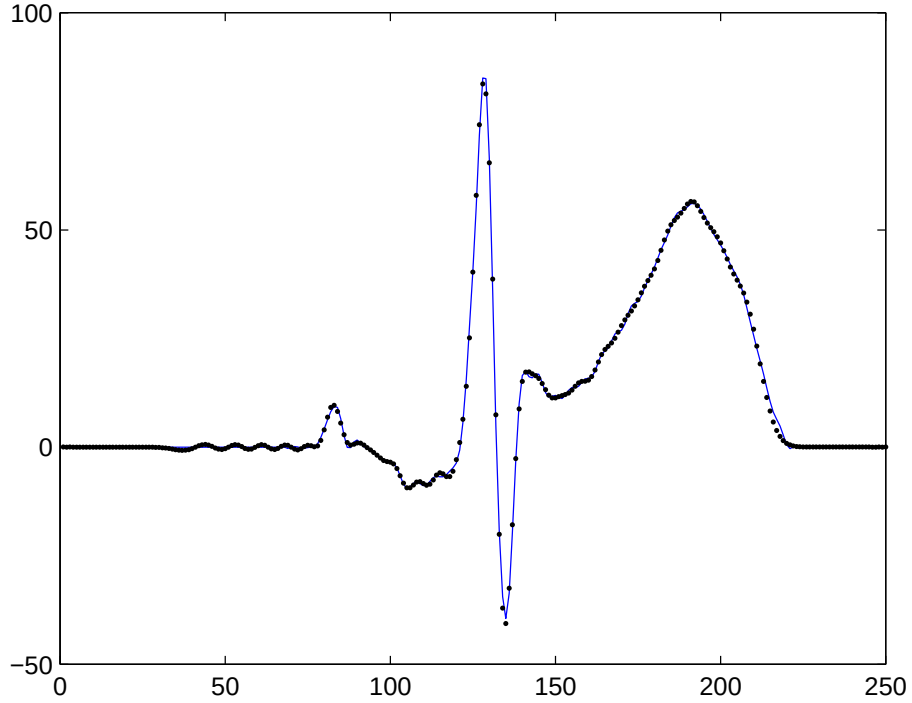


Figura 3.4 – Gráfico da expansão usando 50 funções de Hermite. Produzido no MATLAB.

3.2 Transformadas de tempo discreto

A transformada de Fourier é uma ferramenta versátil de análise de sinais, pois evidencia componentes de frequência existentes num sinal e permite manipulá-lo por meio dessas componentes. Nesta subseção serão introduzidas a transformada de Fourier de tempo discreto e sua generalização, a transformada z. Ambas constituem o fundamento para a análise de sinais discretos no domínio frequencial, inclusive no que tange ao projeto de filtros digitais, assunto da seção consecutiva. A base para a discussão que segue, tanto nesta seção como na 3.3, vem do livro (OPPENHEIM; SCHAFER, 2009).

Transformada de Fourier de tempo discreto

A transformada de Fourier de tempo discreto (DTFT) é definida como

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}. \tag{3.15}$$

Sua inversa, ou seja, a operação que fornece $x[n]$ a partir de $X(e^{j\omega})$ é

$$x[n] = \frac{1}{2\pi} \int_{(2\pi)} X(e^{j\omega})e^{j\omega n}. \tag{3.16}$$

A equação 3.15 pode ser entendida como a relação que dá os coeficientes da expansão de $x[n]$ em funções exponenciais complexas, sendo estas as componentes de uma base para o espaço de sequências infinitas (e somáveis ao quadrado). A variável ω tem unidade em radianos.

Transformada z

A transformada z é definida como

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}, \tag{3.17}$$

onde z é, em geral, um número complexo: $z = Ae^{j\phi}$.

Por vezes, estamos interessados na representação de um sistema linear e invariante no tempo (LIT) através da transformada z, em cujo caso expressamos a equação de diferenças do sistema,

$$\sum_{k=0}^{N} a_k y[n] = \sum_{k=0}^{M} b_k x[n], \tag{3.18}$$

na forma (usando as propriedades de linearidade e deslocamento no tempo da transformada):

$$Y(z) \sum_{k=0}^{N} a_k z^{-k} = X(z) \sum_{k=0}^{M} b_k z^{-k}. \tag{3.19}$$

Podemos ainda escrever a função de transferência (ou resposta em frequência) do sistema como

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=0}^{N} a_k z^{-k}}. \tag{3.20}$$

A equação 3.20 também pode ser rearranjada para evidenciar as raízes do polinômio numerador e do polinômio denominador, chamadas de zeros e pólos, respectivamente. Os zeros e pólos indicam os pontos onde a resposta em frequência do sistema cai a zero ou tende ao infinito. Eles são mais facilmente visualizados no **plano z**, conforme mostra a figura 3.5. O plano z é uma ferramenta indispensável na análise de estabilidade e causalidade de sistemas discretos, bem como no projeto de filtros digitais, conforme será discutido na próxima seção.

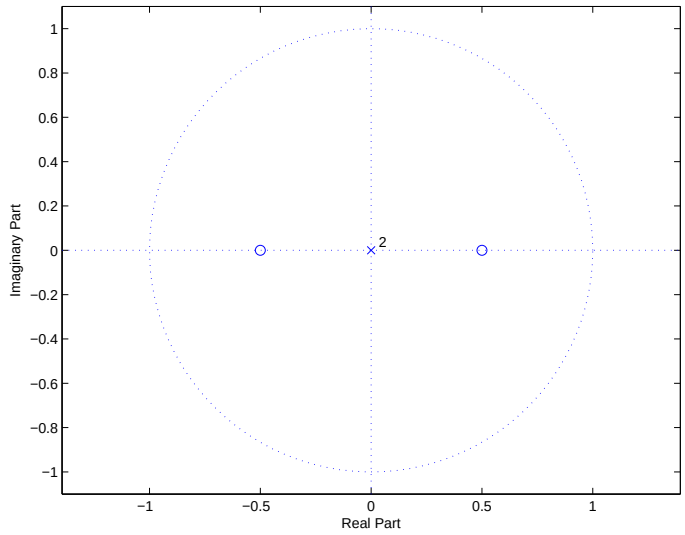


Figura 3.5 – Plano z. Pólos duplos em $z = 0$ e zeros em $z = \pm 0.5$.

3.3 Projeto de filtros digitais

Filtros digitais têm uma grande variedade de aplicações, tanto na área biomédica quanto em qualquer outra que requeira processamento de sinais digitais. Neste trabalho, deseja-se manipular um sinal de ECG com vistas à detecção de batimentos cardíacos e também filtrá-lo para remover ruído e interferência da rede elétrica (50 ou 60 Hz). Esta seção apresenta alguns conceitos sobre o projeto de filtros digitais que auxiliarão o leitor no entendimento dos próximos capítulos, sobretudo no que tange à etapa de pré-processamento dos métodos de detecção de isquemia.

Tipos de filtros

Há duas categorias básicas de filtros: o de resposta impulsiva finita (FIR) e o de resposta impulsiva infinita (IIR). O primeiro diz respeito àqueles filtros cuja resposta depende apenas da entrada em diferentes instantes de tempo. Em outras palavras, a equação de diferenças que descreve o comportamento do filtro não possui termos em $y[n-k], k \neq 0$. Isto também equivale a dizer que $H(z)$, a função de transferência do filtro, não possui pólos que não estejam localizados na origem do plano complexo z. A figura 3.6 mostra a estrutura genérica de um filtro FIR.

O segundo tipo de filtro diz respeito àqueles cuja resposta depende não apenas dos valores da entrada mas também dos valores da própria saída. Em outras palavras, há um laço de *feedback* que conecta a saída a uma parte da entrada do filtro, realimentando-o. Isto implica dizer que sua resposta ao impulso será potencialmente infinita. Por esse motivo aliás, o filtro pode ser instável no sentido BIBO (*bounded input, bounded output*), isto é, pode ter saída de valor ilimitado para uma entrada de valor limitado. A figura 3.7 mostra a estrutura genérica de um filtro IIR.

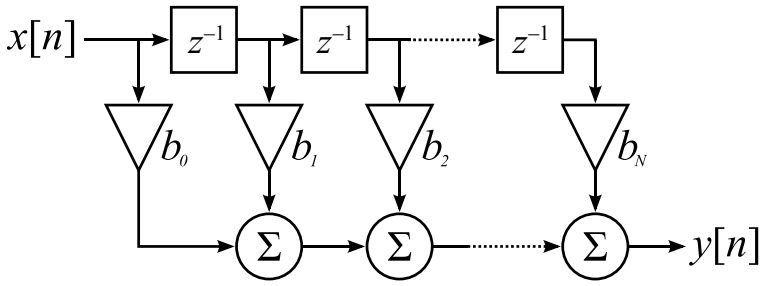


Figura 3.6 – Estrutura genérica de um filtro FIR. Extraída de (BLANCHARD, 2008).

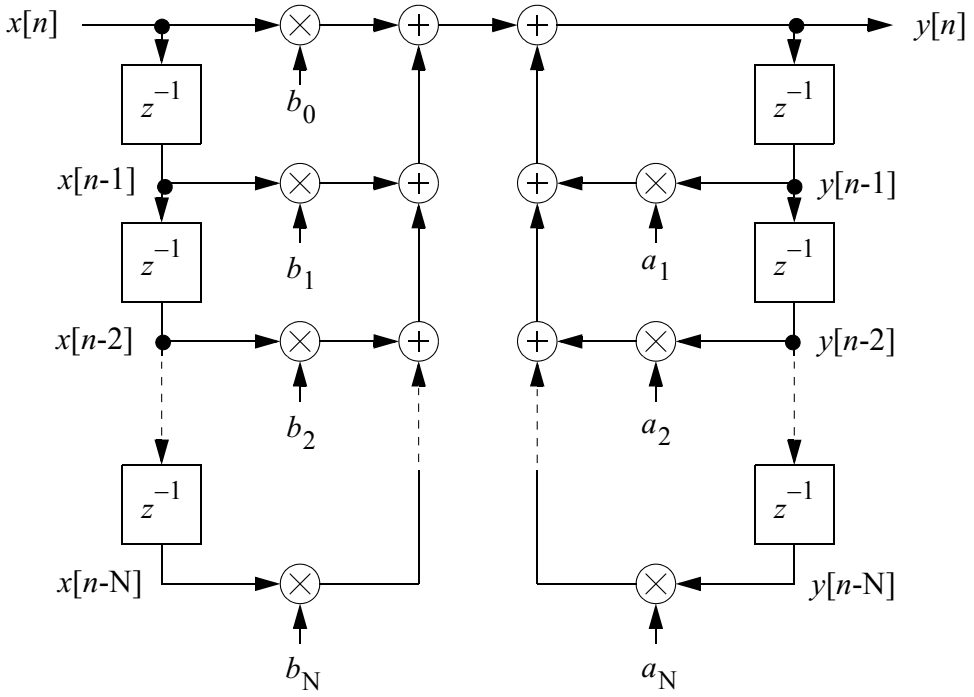


Figura 3.7 – Estrutura genérica de um filtro IIR.

Não se pretende aqui discutir em detalhe os critérios de estabilidade e causalidade de um filtro, questões estas que estão diretamente relacionadas com a região de convergência escolhida para a transformada z do sistema. Entretanto, vale mencionar que um filtro será realizável se ele for causal (saída zero para tempos menores que zero) e estável (saída limitada para entrada limitada). Todos os filtros projetados neste trabalho são realizáveis, de modo que podemos nos concentrar no projeto em si e não abordar estes princípios explicitamente.

Filtros FIR podem ser projetados de várias maneiras, sendo algumas delas: truncagem da resposta ideal, alocação de zeros no plano z e amostragem em frequência. Num primeiro momento, apresentaremos uma metodologia especial desenvolvida por Lynn ((LYNN, 1971), (LYNN, 1977)), em que são projetados filtros com coeficientes inteiros. Consecutivamente, será vista a técnica de truncagem da resposta ideal por janelas. Esta última é necessária para

o algoritmo de reamostragem que será visto no capítulo 6. Ao final da seção, será abordada questão de projeto de filtros IIR, necessária para o condicionamento do sinal de ECG (remoção de ruido/interferência).

Projeto de filtros FIR com coeficientes inteiros

Lynn (LYNN, 1977) propõe uma técnica de projeto de filtros FIR que proporciona vantagens sobre outras técnicas em termos de eficiência computacional, embora seja um tanto restritiva em termos de qualidade da resposta desejada. A principal vantagem desses filtros é que sua equação de diferenças possui apenas coeficientes inteiros, de modo que a filtragem possa ser realizada por multiplicações e adições inteiras, em vez de por operação em ponto-flutuante. Dependendo do caso, pode-se até realizar as operações de multiplicação por meio de deslocamentos de bits, tornando a filtragem ainda mais rápida.

A função de transferência base para um filtro passa-baixas nesta técnica é da seguinte forma:

$$H_{lp}(z) = \frac{1 - z^{-m}}{1 - z^{-1}}. \tag{3.21}$$

O termo denominador indica que há um pólo em $z = 1$ (ou frequência $\omega = 0$ rad), enquanto o termo numerador indica que existem m zeros uniformemente distribuídos ao longo do círculo unitário, com ângulo de separação igual a $(2\pi/m)$ radianos. O zero em $z = 1$ cancela o pólo no mesmo local, de modo que o filtro possui comportamento FIR. Não obstante, o pólo em $z = 1$ introduz um termo em $y[n-1]$ na equação de diferenças, o que é claramente uma característica de filtros IIR. Esta também é uma vantagem, pois normalmente a equação de diferenças da versão IIR de um filtro FIR, quando ela existe, possui menos termos e, por consequência, necessita de menos operações (isto faz sentido porque o laço de realimentação reduz a dependência explícita da saída em termos da entrada).

A primeira coluna da figura 3.8 ilustra a disposição de zeros da equação 3.21 no plano z para alguns valores de m. Como o zero em $z = 1$ é cancelado pelo pólo coincidente, a resposta em frequência do filtro tem seu pico neste ponto, em vez de cair a zero. A figura 3.9 mostra o módulo de $H_{lp}(z)$ quando $m = 3$. A escala de frequências está normalizada no intervalo 0 a 1, correspondendo a 0 e π radianos. Nota-se que a frequência onde o módulo cai a zero é igual a $(2\pi/3)$, ou, na escala da figura, $0,\overline{6}$. Nota-se também que o ganho do filtro na banda passante é o próprio valor de m, neste caso igual a 3. Isto vale para qualquer filtro no formato da Eq. 3.21.

A função de transferência do passa-altas é similar, mas apresenta duas formas:

$$H_{hp_1}(z) = \frac{1 - z^{-m}}{1 + z^{-1}}, \qquad H_{hp_2}(z) = \frac{1 + z^{-m}}{1 + z^{-1}}. \tag{3.22}$$

Aqui o denominador indica que o pólo está em $z = -1$ (ou frequência $\omega = \pi$ rad). Na segunda forma, o numerador implica uma rotação na disposição de zeros no círculo unitário, em relação à primeira forma, conforme ilustra a segunda coluna da figura 3.8. Isto significa que, para m par,

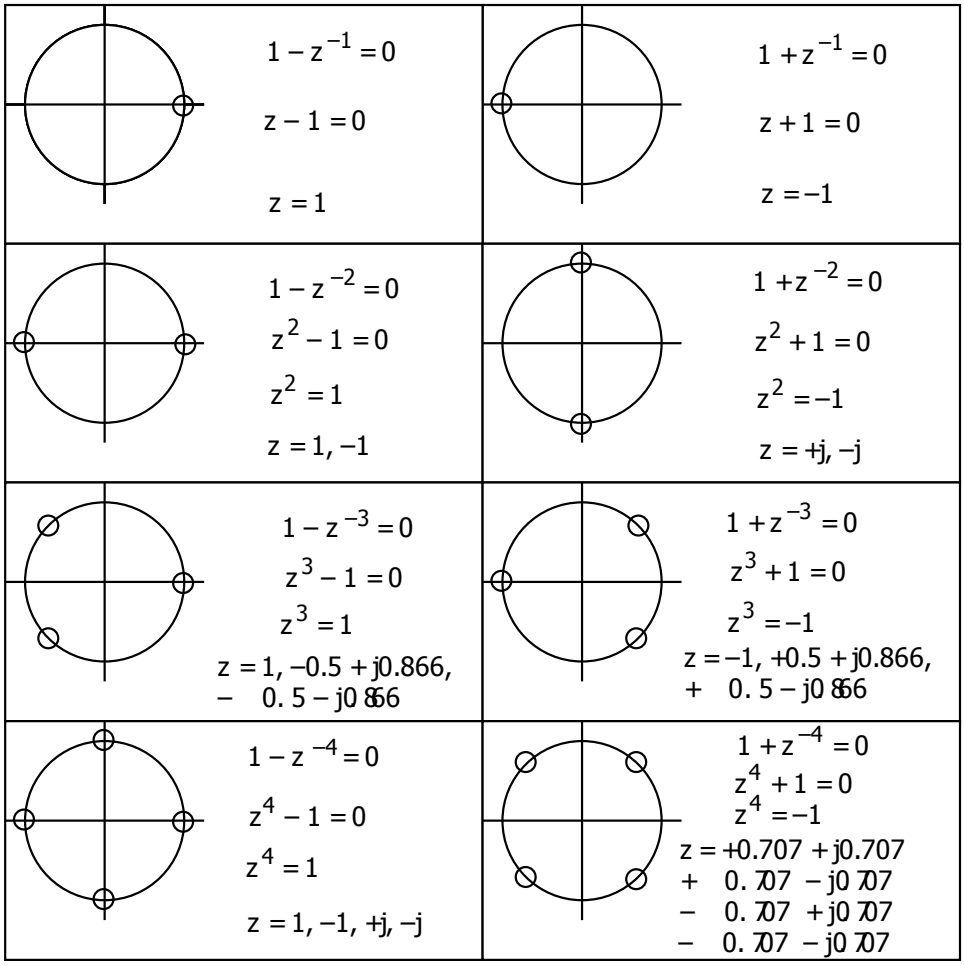


Figura 3.8 – Raízes de $1 - z^{-m}$ (primeira coluna) e de $1 + z^{-m}$ (segunda coluna). Extraída de (TOMPKINS, 1993)

deve-se usar $H_{hp_1}(z)$ no projeto do passa-altas e usar $H_{hp_2}(z)$ somente quando m for ímpar. A figura 3.10 mostra o módulo de $H_{hp_2}(z)$ quando $m = 3$.

O parâmetro m deve ser cuidadosamente escolhido para atender a uma determinada especificação. Nesta técnica de projeto, não há como satisfazer uma especificação arbitrária, pois a localização de pólos e zeros no círculo unitário é bastante restritiva. Contudo, dadas uma frequência de corte ω_c e a magnitude de $H(e^{j\omega})$ nesta frequência, é possível encontrar um valor de m que atenda a essas restrições, mesmo que outras características da resposta em frequência permaneçam irrestritas. Assim sendo, deve-se resolver a equação

$$|H(e^{j\omega})|\big|_{\omega=\omega_c} = mC, \tag{3.23}$$

onde C é a magnitude desejada (com respeito a um ganho unitário) em ω_c e m constitui o ganho do filtro na banda passante. De maneira similar, foi realizada a análise da função de transferência

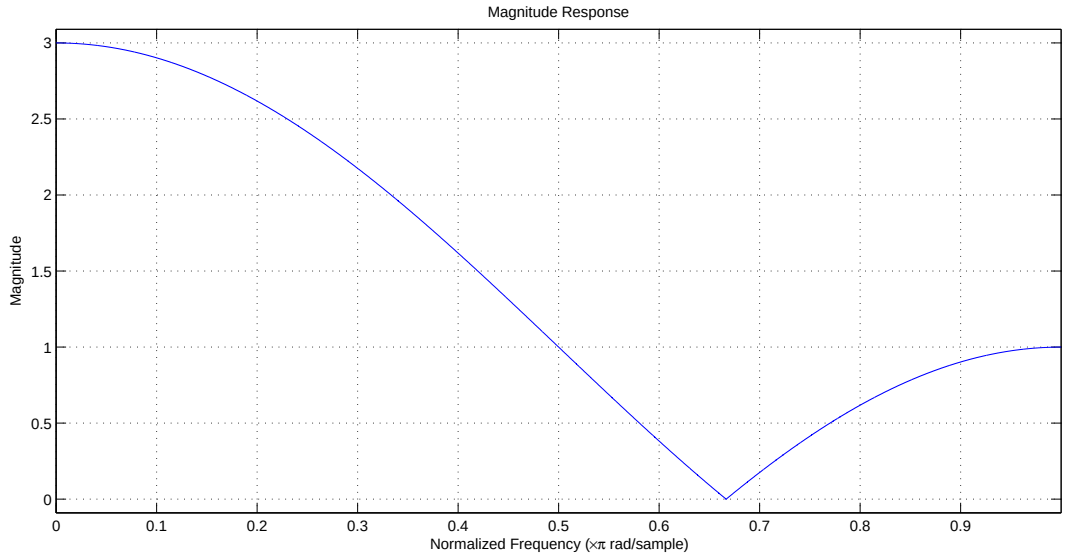


Figura 3.9 – Resposta em frequência do filtro passa-baixas para $m = 3$. Produzida no MATLAB

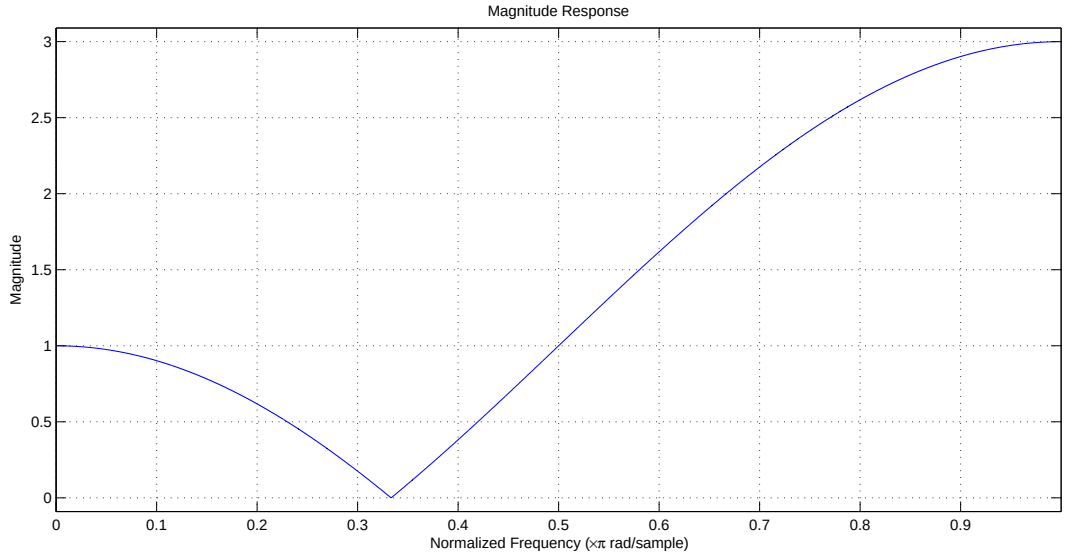


Figura 3.10 – Resposta em frequência do filtro passa-altas para $m = 3$. Produzida no MATLAB

de ordem N, que nada mais é do que a composição de N filtros iguais em cascata:

$$H_{lp}^{N}(z) = \left(\frac{1-z^{-m}}{1-z^{-1}}\right)^{N} \tag{3.24}$$

e

$$H_{hp_1}^{N}(z) = \left(\frac{1-z^{-m}}{1+z^{-1}}\right)^{N}, \quad H_{hp_2}^{N}(z) = \left(\frac{1+z^{-m}}{1+z^{-1}}\right)^{N}. \tag{3.25}$$

A equação 3.23 pode ser expressa como uma equação trigonométrica, usando a função seno, mas não possui solução analítica simples. Portanto, neste trabalho utilizou-se o resolvedor numérico do MATLAB (função `vpasolve`) e encontrou-se valores de ω_c para diversas combinações

de m e C. Variou-se m no intervalo $[2,200]$ e C possuía um de três valores, correspondendo às atenuações de -3dB, -6dB e -24dB, relativas ao ganho m do filtro. A equação trigonométrica fornecida ao resolvedor vem da substituição $z = e^{j\omega}$ em $H_{lp}^{N}(z)$ e é dada por

$$\left| \frac{\sin\left(m\frac{\omega}{2}\right)}{m\sin\left(\frac{\omega}{2}\right)} \right|^{N} = C. \tag{3.26}$$

Com os valores de m e ω_c, para cada atenuação desejada e ordem N do filtro, construiu-se um modelo de regressão que nos dá m em função de ω_c e estimou-se os parâmetros do modelo usando a função `fitnlm` do MATLAB. Por fim, foi encontrado um modelo de regressão que expressa m em função de ω_c e N, para $1 \leq N \leq 10$. Abaixo estão as expressões encontrdas:

$$m = \begin{cases} \frac{0.91823\pi}{\omega_c\sqrt{N+0.072177}} & \text{aten. de -3dB} \\ \frac{1.2992\pi}{\omega_c\sqrt{N+0.15023}} & \text{aten. de -6dB} \\ \frac{1.732\pi}{\omega_c\sqrt{N+0.28356}} & \text{aten. de -24dB} \\ \frac{2\pi}{\omega_c} & \text{aten. de } -\infty \end{cases}. \tag{3.27}$$

É importante ressaltar que o mesmo modelo pode ser usado tanto no projeto do passa-baixas quanto no do passa-altas, pois o que muda é a rotação da disposição de zeros no plano z e não o número de zeros que dividem o círculo unitário. (Podemos pensar no passa-altas como um deslocamento de π radianos da resposta em frequência do passa-baixas.)

Projeto de filtros FIR por truncagem da resposta ideal

A segunda maneira de projetar filtros FIR usada neste trabalho advém da construção da resposta impulsiva do filtro com base numa resposta ideal. A resposta em frequência de um filtro passa-baixas ideal é dada por:

$$H_d(e^{j\omega}) = \begin{cases} e^{j\omega\alpha} & |\omega| \leq \omega_c \\ 0 & \omega_c \leq |\omega| \leq \pi \end{cases}, \tag{3.28}$$

periódica com período 2π. O parâmetro α é um fator de fase e garante, no domínio tempo, o atraso necessário para tornar o filtro causal. A figura 3.11 mostra um esboço da magnitude de $H_d(e^{j\omega})$. A resposta impulsiva do filtro é dada pela aplicação da transformada inversa de Fourier de tempo discreto:

$$h_d[n] = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{j\omega\alpha}e^{j\omega n} = \frac{\sin\left(\omega_c\left(n-\alpha\right)\right)}{\pi\left(n-\alpha\right)}. \tag{3.29}$$

Desejamos truncar $h_d[n]$ num intervalo de tamanho M, com $\alpha = \frac{M}{2}$. Para tanto, utilizamos uma "janela" delimitadora, $w[n]$. Assim, tem-se

$$h[n] = h_d[n]w[n], \quad 0 \leq n \leq M, \tag{3.30}$$

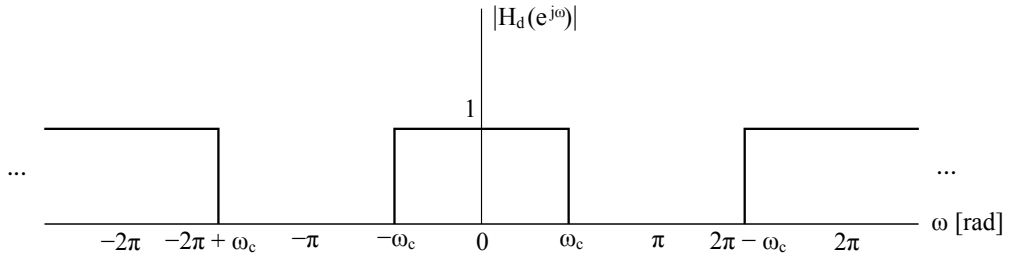


Figura 3.11 – Resposta em frequência do filtro passa-baixas ideal.

que no domínio frequência equivale a uma convolução de $H_d(e^{j\omega})$ com $W(e^{j\omega})$, e implica num "espalhamento" de energia da resposta ideal. A figura 3.12 mostra um esboço da magnitude de $H(e^{j\omega})$ juntamente com a resposta ideal (em traço segmentado). A figura também revela os diversos parâmetros utilizados no projeto do filtro.

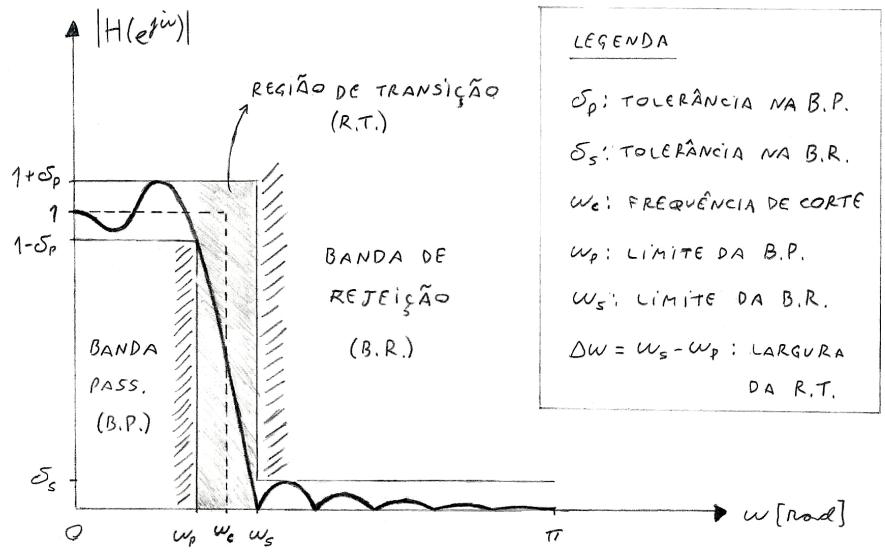


Figura 3.12 – Especificações de projeto do filtro passa-baixas realizável.

Visando reduzir o espalhamento de energia o tanto quanto possível, diversos tipos de janela foram propostos na literatura de processamento de sinais ((HARRIS, 1978), (KAISER; SCHAFER, 1980)), cada uma com suas características. As características mais importantes de uma janela, do ponto de vista de sua resposta em frequência, são a largura do lóbulo principal e a atenuação média da banda de rejeição. Alguns tipos comuns de janela utilizados são: retangular, Bartlett (ou triangular), Hanning, Hamming, Blackman e Kaiser.

Os passos para se obter a resposta desejada de um filtro passa-baixas digital são:

1. Tomar a menor tolerância entre as das bandas passante e de rejeição: $\delta = min(\delta_p, \delta_s)$;
2. Calcular a atenuação requerida em decibéis, com base na tolerância: $A = -20\log_{10}(\delta)$;

3. Escolher um tipo de janela que satisfaça a atenuação desejada;

4. Calcular a largura da região de transição: $\Delta\omega = |\omega_s - \omega_p|$;

5. Descobrir o tamanho da resposta com base em $\Delta\omega$ e na janela: $M = f_w(\Delta\omega)$; e[1]

6. Calcular $h[n]$ usando as Eqs. 3.29 e 3.30.

Conversão de filtros IIR analógicos

Historicamente, o projeto de filtros IIR digitais sempre teve como base as técnicas de projeto desenvolvidas no mundo contínuo, pois estas já são bastante avançadas e sofisticadas, de modo que é vantajoso reaproveitá-las no mundo discreto em vez de criar um novo ferramental "a partir do zero". A conversão da resposta impulsiva ou em frequência de um filtro de tempo contínuo para a resposta de um filtro de tempo discreto pode ser alcançada através de um dos seguintes procedimentos: invariância ao impulso (do inglês, *impulse invariance*) ou transformação bilinear.

Abaixo está um condensado da segunda técnica supracitada, já que é a mais comumente utilizada em ferramentas computacionais de projeto de filtros, como o MATLAB. Na verdade, o que segue é apenas uma elucidação do procedimento de projeto. Diferentemente do caso FIR, em que se havia uma fórmula simples para obtenção da resposta do filtro, o projeto dos filtros IIR empregados neste trabalho será realizado com auxílio do MATLAB, conforme será visto no capítulo 6.

A transformação bilinear é dada pela substituição

$$s = \frac{2}{T_d}\left(\frac{1-z^{-1}}{1+z^{-1}}\right), \tag{3.31}$$

onde T_d é um parâmetro associado à passagem do tempo contínuo para tempo discreto. Essa transformação mapeia o eixo $j\Omega$ do plano s no círculo unitário do plano z. Isto é, mapeia Ω no intervalo $(-\infty, +\infty)$ para ω no intervalo $[-\pi, +\pi]$, sendo portanto não-linear. Por este fato, ela provoca uma distorção das frequências, conforme evidencia a relação

$$\omega = \frac{2}{T_d}\arctan\left(\Omega\frac{T_d}{2}\right). \tag{3.32}$$

No projeto de um filtro IIR, é necessário converter as frequências especificadas (ω_p e ω_s) no mundo digital para o seu correspondente contínuo, usando a Eq. 3.32. As tolerâncias (δ_p e δ_s) não sofrem alteração. Com as especificações no mundo contínuo, determina-se a função de transferência do filtro analógico, que pode ser do tipo Butterworth, Chebyshev I, Chebyshev II ou elíptico, e finalmente faz-se a conversão da resposta para o mundo discreto usando a Eq. 3.31.

[1] a função f_w é apenas uma maneira de dizer que existe uma relação entre $\Delta\omega$ e M

3.4 Resumo

Este capítulo, que pode à primeira vista parecer extenso, assenta as peças necessárias ao entendimento dos assuntos vindouros. Aqui foram introduzidos os conceitos de norma, base e produto interno, no contexto de espaços vetoriais. Conceitos estes que dão lugar à questão da aproximação de sinais discretos. Uma interpretação foi apresentada no espaço Euclideano com auxílio da operação de projeção. Logo em seguida, viu-se que a sequência de polinômios de Hermite, assim como o conjunto de funções a ela associado, forma uma base ortonormal para o espaço de funções integráveis ao quadrado. No caso discreto, foi abordada uma técnica especial de cálculo das funções de Hermite que permite aproximação de um sinal discreto por meio de um simples produto matricial.

Depois, algumas transformadas de tempo discreto foram vistas, notadamente a transformada de Fourier de tempo discreto (DTFT) e a transformada z. Viu-se como é possível representar a equação de diferenças de um sitema linear e invariante no tempo (LIT) por meio da transformada z e que esta representação permite fácil identificação dos pólos e zeros do sistema. Ainda, o plano z oferece uma ferramenta imprescindível de análise de estabilidade e causalidade desses sistemas, bem como facilita o projeto de filtros, conforme visto na seção subsequente.

Existem diversas maneiras de se projetar filtros FIR, duas das quais foram discutidas na seção 3.3. A primeira, tocante ao projeto de filtros com coeficientes inteiros, forneceu um algoritmo para determinação da função de transferência de filtros passa-baixa e passa-alta com coeficientes inteiros. Depois, foi visto o projeto de um filtro passa-baixa com base na truncagem de sua resposta ideal no domínio tempo. Essa truncagem se dá pelo uso de janelas delimitadoras e resulta em um algoritmo para o cálculo da resposta impulsiva do filtro. Ainda nesta seção teve lugar um breve discurso sobre a conversão de filtros IIR analógicos para o caso discreto, com ênfase na transformação bilinear.

4 AVALIAÇÃO DE DESEMPENHO

A fim de avaliar os métodos de detecção de isquemia, necessita-se estabelecer métricas de avaliação. Interessa-se principalmente pela confiabilidade dos métodos, isto é, sua capacidade de classificar corretamente batimentos cardíacos em isquêmicos ou não isquêmicos. A seguir será apresentada a tradicional matriz confusão e dela serão derivadas seis métricas de avaliação de confiabilidade.

4.1 Matriz de confusão

Uma metodologia bastante popular de avaliação é por meio da **matriz de confusão**, um tipo especial de tabela de contingência[1]. Ela separa os resultados de um teste diagnóstico em quatro categorias, que são as possíveis combinações de resultado positivo ou negativo no teste avaliado, com resultado positivo ou negativo num diagnóstico de referência, sendo este proveniente do conhecimento de especialistas. A tabela 4.1 mostra o formato geral da matriz de confusão.

Resultado do teste	Condição real	
	Presente	Ausente
Positivo	verdadeiro positivo (VP)	falso positivo (FP)
Negativo	falso negativo (FN)	verdadeiro negativo (VN)

Tabela 4.1 – Matriz de confusão para testes diagnósticos.

Para o trabalho desenvolvido, assim como para os métodos de detecção de isquemia originalmente propostos, o diagnóstico de referência está armazenado juntamente com os registros de ECG sob a forma de anotações. As anotações fornecem a localização dos batimentos cardíacos no tempo de gravação e dizem se um batimento foi considerado por cardiologistas como isquêmico ou não. Na verdade, as anotações referentes a isquemia estão organizadas de acordo com **episódios isquêmicos**. Os episódios têm início quando uma quantidade mínima de batimentos apresenta uma característica comum representando sintoma de isquemia e se estendem até o momento em que esta característica se torna ausente. Quatro tipos possíveis de característica sinalizam ocorrência de episódio isquêmico: elevação do segmento ST (denotada pelo símbolo "ST+"), depressão do segmento ST ("ST-"), elevação da onda T ("T+") e depressão da onda T ("T-").

[1] tabelas de contingência são usadas para registrar observações independentes de duas ou mais variáveis aleatórias.

4.2 Sensibilidade e Especificidade

A sensibilidade é a capacidade do teste fornecer verdadeiros positivos (interpretação correta de ocorrência de isquemia) entre os batimentos verdadeiramente isquêmicos. Ela é obtida por

$$SE = \frac{VP}{VP+FN}. \tag{4.1}$$

No contexto de epidemiologia[2], sabe-se que um teste diagnóstico é sensível se ele raramente deixa de identificar indivíduos doentes. Este fato é importante na detecção de isquemia pois se deseja que, no melhor caso, todas as ocorrências de batimento isquêmico sejam identificadas.

A especificidade, por outro lado, é a capacidade do teste obter verdadeiros negativos (interpretação correta de ausência de isquemia) entre os batimentos verdadeiramente não isquêmicos. Ainda no contexto de epidemiologia, um teste é específico se ele raramente comete equívoco em identificar indivíduos sadios. A fórmula para o cálculo da especificidade é

$$ES = \frac{VN}{VN+FP}. \tag{4.2}$$

4.3 Valores Preditivos Positivo e Negativo

O valor preditivo positivo (também chamado de precisão ou preditividade positiva) é a razão entre os verdadeiros positivos e os diagnosticados como positivos pelo teste. Ele pode ser entendido como a probabilidade de um resultado positivo do teste refletir efetivamente a condição testada. Sua expressão é

$$PP = \frac{VP}{VP+FP}. \tag{4.3}$$

Em contrapartida, o valor preditivo negativo (ou preditividade negativa) é a proporção de verdadeiros negativos em relação aos diagnosticados como negativos pelo teste. Pode ser entendido como a probabilidade de um resultado negativo do teste refletir efetivamente a ausência da condição. Sua expressão é

$$PN = \frac{VN}{VN+FN}. \tag{4.4}$$

Os valores preditivos não se mantêm constantes para toda instância de teste realizada. Eles dependem da prevalência da doença na população. Se a prevalência for baixa, isto é, se relativamente poucos indivíduos se encontrarem doentes, obterá-se muitos falsos positivos, mesmo que o teste tenha especificidade alta. Reciprocamente, se a prevalência for alta, pode-se esperar um maior número de falsos negativos, ainda que o teste seja sensível. Portanto, quanto menor a prevalência, menor o PP e maior o PN, sendo a relação inversa também válida.

[2] epidemiologia é a ciência que estuda a distribuição de fenômenos patológicos e suas causas, nos seres humanos.

4.4 Acurácia e Taxa de Falha

A acurácia é a proporção de acertos na população e fornece uma medida do quão próximo o resultado do teste está do diagnóstico de referência. Ela é calculada por

$$AC = \frac{VP + VN}{N}. \tag{4.5}$$

A taxa de falha é a proporção de erros na população e fornece uma medida geral de falha do teste. Ela é dada por

$$TF = \frac{FP + FN}{N}. \tag{4.6}$$

5 MÉTODOS DE DETECÇÃO DE ISQUEMIA

Neste capítulo serão introduzidos os métodos de detecção de isquemia selecionados para implementação. Antes de detalhar cada método nas seções 5.2, 5.3 e 5.4, far-se-á na seção 5.1 uma pequena discussão sobre as possíveis categorias de métodos existentes. Ao final do capítulo, um breve resumo é apresentado contendo os aspectos principais dos métodos estudados.

5.1 Categorização de métodos

Na literatura, os métodos para detecção de isquemia, assim como para detecção de outros tipos de doença cardíaca, empregam técnicas variadas de (i) processamento de sinais, (ii) extração de características e (iii) classificação das características. Cada um destes grupos corresponde a uma etapa do processamento, sendo que todas incluem técnicas amplamente diversificadas.

O primeiro grupo constitui o que se denomina etapa de **pré-processamento**. A tabela 5.1 mostra algumas categorias de técnicas desta etapa e referências de algoritmos em cada categoria. A tabela 5.2 agrupa alguns métodos de detecção de acordo com técnicas usadas na **extração**, enquanto a tabela 5.3 agrupa-os de acordo com técnicas usadas na **classificação**. Contudo, as técnicas de análise para extração e classificação não se limitam às das tabelas apresentadas. Há ainda métodos que utilizam modelagem paramétrica, mineração de dados, autômatos finitos, métodos sintáticos, entre outras. Também é importante mencionar que muitos dos algoritmos referenciados na primeira tabela são incluídos como parte da etapa de pré-processamento dos métodos de detecção de isquemia, que por sua vez são referenciados nas demais tabelas.

Filtragem linear (FIR/IIR, suavização, diferenciação, etc.)	(CHEN; CHEN; CHAN, 2006; ELGENDI, 2013; OKADA, 1979; PAN; TOMPKINS, 1985; DASKALOV; DOTSINSKY; CHRISTOV, 1998)
Filtragem adaptativa	(PARK; LEE; YOON, 1998)
Interpolação polinomial	(BADILINI; MOSS; TITLEBAUM, 1991)
Técnicas não-lineares (matemática morfológica, produtório, máximo/mínimo, etc.)	(CHU; DELP, 1989; OKADA, 1979; SUN; CHAN; KRISHNAN, 2005; SUPPAPPOLA; SUN, 1994; TRAHANIAS, 1993; PAN; TOMPKINS, 1985)
Decomposição wavelet	(CHEN; CHEN; CHAN, 2006; PARK; LEE; YOON, 1998)
Thresholding adaptativo	(CHEN; CHEN; CHAN, 2006; ELGENDI, 2013; SUN; CHAN; KRISHNAN, 2005; PAN; TOMPKINS, 1985)

Tabela 5.1 – Categorias de métodos de acordo com técnicas comuns de pré-processamento.

Uso de template	(AKSELROD et al., 1987; GARCIA et al., 2000; COUCEIRO et al., 2008; MOHEBBI; MOGHADAM, 2007)
Decomposição wavelet	(RANJITH; BABY; JOSEPH, 2003; SENHADJI et al., 1995; MILOSAVLJEVIC; PETROVIC, 2006)
Transform. ortogonal (PCA, Karhunen-Loève, Hermite)	(CASTELLS et al., 2007; ROCHA et al., 2010; AFSAR; ARIF, 2007; GOPALAKRISHNAN; ACHARYA; MUGLER, 2004; PANG et al., 2005)
Análise frequencial ou tempo-frequencial	(ROCHA et al., 2010; SENHADJI et al., 1995; BADILINI et al., 1992; COUCEIRO et al., 2008)
Propriedades estatísticas (correlação, erro médio, etc.)	(RANJITH; BABY; JOSEPH, 2003; BADILINI et al., 1992; COUCEIRO et al., 2008; GARCIA et al., 2000)
Conhecimento prévio	(ELGENDI, 2013; PAPALOUKAS et al., 2000)
Características pontuais (pontos J e isoelétrico, picos de ondas, desvio de segmento ST)	(AKSELROD et al., 1987; GOLETSIS et al., 2004; PAPALOUKAS et al., 2000; RANJITH; BABY; JOSEPH, 2003; ROCHA et al., 2010; SENHADJI et al., 1995; VILA et al., 1997; BADILINI et al., 1992; COUCEIRO et al., 2008; MOHEBBI; MOGHADAM, 2007)

Tabela 5.2 – Categorias de métodos de acordo com técnicas comuns de extração.

Redes neurais artificiais	(PAPALOUKAS et al., 2001; ROCHA et al., 2010; AFSAR; ARIF, 2007; COUCEIRO et al., 2008; GOPALAKRISHNAN; ACHARYA; MUGLER, 2004; MOHEBBI; MOGHADAM, 2007; PANG et al., 2005; STAMKOPOULOS et al., 1997; MAGLAVERAS et al., 1998)
Fuzzy ou neuro-fuzzy	(EXARCHOS et al., 2007; VILA et al., 1997)
Conjunto de regras	(AKSELROD et al., 1987; EXARCHOS et al., 2006; PAPALOUKAS et al., 2002)
Clusterização	(BADILINI et al., 1992)
Modelos de Markov	(ANDREAO et al., 2004)
Algoritmos genéticos	(GOLETSIS et al., 2004)

Tabela 5.3 – Categorias de métodos de acordo com técnicas comuns de classificação.

5.2 Método de Rocha et al.

Este método é baseado em dois pontos-chave: (i) desvio do segmento ST com relação à linha de base e (ii) expansão do complexo QRS e da onda T em funções de Hermite. A figura 5.1 mostra o diagrama em blocos da estratégia adotada. As etapas do método são detalhadas a seguir.

Pré-processamento

Nesta etapa, um sinal discreto contendo as amplitudes do ECG é processado com vistas à redução de ruído, segmentação em ondas características, eliminação de extra-sístoles ventriculares e, por último, remoção da linha de base.

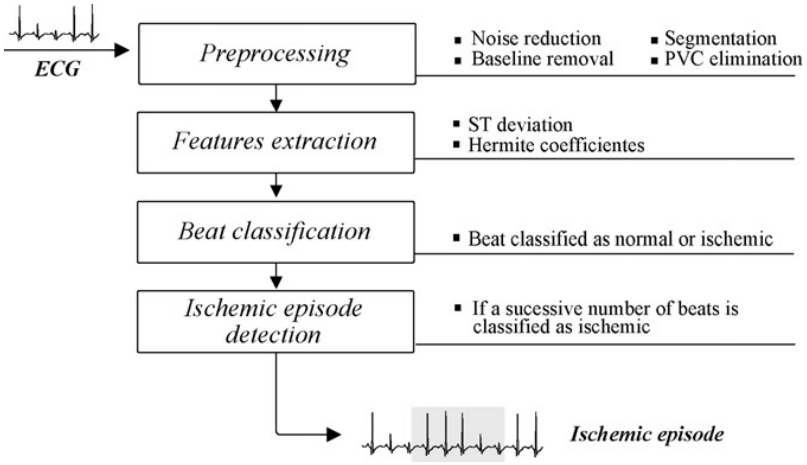


Figura 5.1 – Diagrama de blocos da estratégia proposta por Rocha et al. Extraído de (ROCHA et al., 2010)

A redução de ruído é realizada com um filtro passa-baixas de Butterworth de 4ª ordem e frequência de corte 40 Hz. O sinal filtrado é então submetido a um procedimento de segmentação, em que as ondas características de cada batimento são identificadas. Na proposta dos autores, utiliza-se o algoritmo descrito em (SUN; CHAN; KRISHNAN, 2005), baseado em derivação morfológica do sinal (um tipo de filtragem não linear, usando máximos e mínimos). Na saída obtêm-se as localizações de início, pico e fim de cada onda característica.

Após a segmentação, os batimentos passam por um procedimento de remoção de extra-sístole ventricular (ou PVC, do inglês *premature ventricular contraction*), conforme descrito em (COUCEIRO et al., 2008). O último procedimento nesta etapa consiste em remover a linha de base em cada batimento cardíaco. Isto é necessário para as próximas etapas, onde se exige que os batimentos estejam bem alinhados com o nível isoelétrico do sinal.

Extração

Nesta etapa, dois grupos de características são extraídos do ECG: (i) a medida do desvio de segmento ST e (ii) os coeficientes da expansão de Hermite do complexo QRS e das ondas T. Variações no desvio do segmento ST são usadas para discriminar os batimentos isquêmicos dos normais. De maneira similar, variações na morfologia do complexo QRS e da onda T indicam presença ou não de isquemia.

Para compor o primeiro grupo de características, dois valores são obtidos através de dois métodos distintos. Um deles é baseado na localização dos picos de onda R e na frequência cardíaca, conforme descrito em (PANG et al., 2005). O outro é derivado de uma análise do ECG em tempo-frequência, usando a transformada de Wigner-Ville descrita a seguir.

A distribuição de Wigner-Ville permite representar o espectro de frequências de um sinal

contínuo ao longo do tempo. Ela é definida pela equação

$$W_x(t,f) = \int_{-\infty}^{\infty} x\left(t+\frac{\tau}{2}\right) x^*\left(t-\frac{\tau}{2}\right) e^{-j2\pi f\tau} d\tau, \tag{5.1}$$

onde o termo em x é a função de autocorrelação instantânea e o operador * indica o conjugado complexo. Em tempo discreto, ela é chamada de pseudo-transformada de Wigner-Ville e assume a forma abaixo:

$$W_x(nT,f) = 2T \sum_{p=-L_1}^{L_2} x[n+p]x^*[n-p]w[p]w^*[-p]e^{-j4\pi fp}. \tag{5.2}$$

Nesta equação, T é o período de amostragem e w é uma janela simétrica de largura L_1+L_2+1. Os autores afirmam que o tamanho da janela deve ser maior que o do complexo QRS mais largo, para garantir que as características do formato de onda não sejam perdidas na transformação. Além disso, sabe-se que esta transformada pode ser calculada a partir da DFT (transformada discreta de Fourier) da autocorrelação de x no tamanho da janela. Uma escolha conveniente para o tamanho da janela é uma potência de 2, pois neste caso o cálculo da DFT é mais eficiente. Por exemplo, assumindo que um complexo QRS não ultrapasse a largura de 100 ms e tomando uma frequência de amostragem de 250 Hz, pode-se utilizar uma janela de tamanho 32, pois $250 \cdot 0,1 = 25$.

Após a transformação, soma-se os valores absolutos das componentes de baixa frequência do sinal em dois intervalos de tempo distintos: um à esquerda do pico de onda R e outro à direita do mesmo ponto. Os autores dizem utilizar os índices de frequência correspondentes a frequências entre 0 e 0,2 na escala normalizada. Entretanto, não mencionam quais os limites das faixas de tempo analisadas, nem mesmo o tipo de janela utilizado na Eq. 5.2. Assume-se que a janela é do tipo retangular e que os limites das faixas de tempo devam ser estabelecidos conforme o caso.

O resultado da soma das componentes frequenciais provê informações sobre a concentração de energia nas duas faixas de tempo, sendo o restante do algoritmo uma simples busca pelo ponto mínimo em cada uma delas. O ponto mínimo à esquerda é o chamado ponto isoelétrico, enquanto o ponto mínimo à direita é chamado ponto J. Estes conceitos são discutidos em detalhe no artigo original. Basta apenas mencionar que a medida do desvio de segmento ST é a diferença entre as amplitudes nos pontos J e isoelétrico.

Para o segundo grupo de características, os autores propuseram uma técnica baseada na expansão em funções de Hermite. Já foi vista no capítulo 3 a definição dessas funções em tempo contínuo para o caso particular em que elas não apresentam um parâmetro de dilatação. Aqui será reintroduzida a expressão da função de Hermite de tempo contínuo para o caso dilatado por um fator l, e usando a notação em t da variável independente:

$$\psi_n(t,l) = \left(\frac{e^{-(t/l)^2}}{n!2^n\sqrt{\pi}}\right)^{\frac{1}{2}} H_n\left(\frac{t}{l}\right) \tag{5.3}$$

Pode-se perceber que aqui os autores optam por uma representação em tempo contínuo. A expansão consiste em obter uma lista de coeficientes que satisfazem a equação

$$\hat{y}(t) = \sum_{j=0}^{M-1} c_j \psi_j(t,l), \tag{5.4}$$

onde c_j é o coeficiente da j-ésima função de Hermite e M é o número de funções utilizadas na aproximação. Os coeficientes são dados pela solução algébrica do problema de minimização do erro quadrático, conforme a equação

$$C = (H^T H)^{-1} H^T Y, \tag{5.5}$$

onde C é um vetor coluna contendo os M coeficientes da expansão, Y é um vetor coluna contendo as amplitudes do sinal original e H é uma matriz cujas colunas são as funções de Hermite:

$$H = [\psi_0 \ \psi_1 \ \cdots \ \psi_{m-1}]. \tag{5.6}$$

Antes deste procedimento, contudo, o sinal de um batimento cardíaco é separado em dois segmentos: um deles contendo as amostras do complexo QRS e o outro, as da onda T. Essa separação permite aproximar melhor o formato do batimento em cada região, já que o complexo QRS e a onda T possuem morfologia distinta. Além disso, essas ondas estão bastante afastadas temporalmente, o que indica a necessidade de muitas funções de Hermite na aproximação caso se queira aproximar o sinal do batimento na íntegra. Os autores, pelo contrário, sugerem a utilização de apenas 6 funções na aproximação e, portanto, se justifica a separação em dois segmentos.

Os autores ainda mencionam a necessidade de reamostrar cada segmento para 64 amostras, afim de utilizar sempre a mesma largura para as funções de Hermite. Isto significa que a matriz H pode ser calculada previamente e não se modifica durante a execução do algoritmo, permitindo aumento de eficiência na computação. Assim, dois conjuntos de 64 amostras por batimento são submetidos ao procedimento de expansão de Hermite, obtendo-se como resultado duas listas de 6 coeficientes. Estas listas são então incorporadas ao conjunto de características, de cuja parte já fazem as medidas do desvio de segmento ST.

Classificação

O esquema de classificação usado por Rocha et al. envolve o uso de redes neurais artificiais. Para cada derivação de ECG, duas redes do tipo *feed-forward* são treinadas com as características obtidas na etapa de extração. O algoritmo utilizado no treinamento é o Levenberg-Marquardt. O número de camadas é o mesmo para todas as derivações, sendo duas camadas ocultas mais a de entrada e a de saída. A função de transferência é a tangente sigmóide. O número de neurônios na entrada corresponde ao número de características extraídas, qual seja, $2+6+6=14$, enquanto a saída conterá apenas um neurônio, que fornece o resultado correspondente a elevação ou a

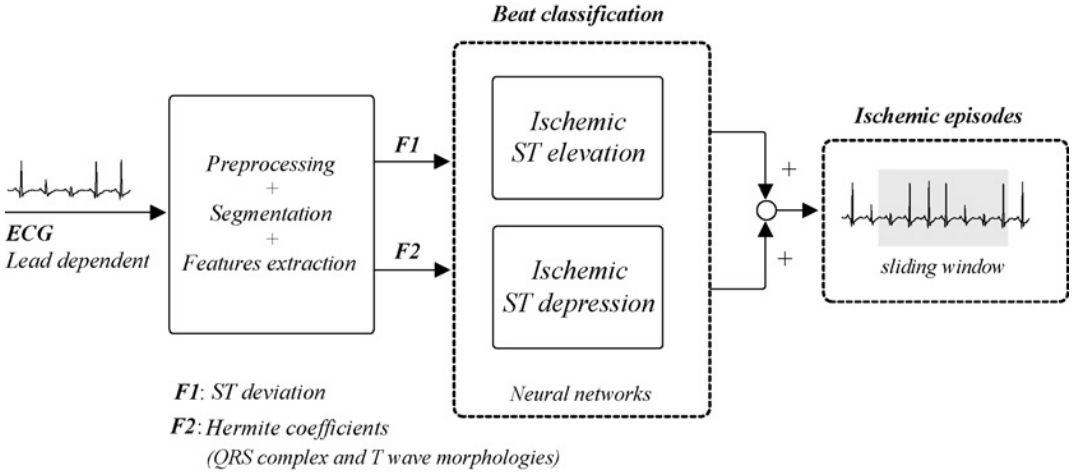


Figura 5.2 – Diagrama da estratégia de classificação de Rocha et al. Extraído de (ROCHA et al., 2010)

depressão do segmento ST. O critério de inversão da onda T não é utilizado nesta estratégia. A figura 5.2 ilustra o esquema de classificação assim proposto.

Ainda nesta etapa, há um procedimento de detecção de episódios isquêmicos. Sugerem os autores que um episódio isquêmico seja detectado toda vez que uma sequência de 40 batimentos tiver pelo menos metade classificada como isquêmica. Assim, uma janela de tamanho 40 percorre o resultado da classificação avançando um batimento por vez e sinaliza a ocorrência de um episódio quando o número de batimentos isquêmicos ultrapassa 20, ou sinaliza a ausência de episódio quando este número cai a 20 ou menos. Num segundo passo, episódios adjacentes com separação menor que 40 batimentos são unidos.

5.3 Método de Mohebbi e Moghadam

Este método tem como principal recurso a obtenção de um modelo (ou *template*, em inglês) de batimento cardíaco considerado normal, extraído do próprio registro sobre o qual se deseja fazer a detecção de isquemia. As etapas do algoritmo podem ser visualizadas no diagrama da figura 5.3 e seus detalhes são discutidos a seguir.

Pré-processamento

Esta etapa consiste de identificação dos complexos QRS, obtenção de um modelo de batimento, remoção de ruído, rejeição de artefatos e extração do segmento ST. A identificação dos complexos QRS e dos picos de onda R é feita através de um algoritmo descrito em (TOMPKINS, 1993), que faz uso extensivo de filtragem para realçar os complexos QRS e de *thresholding* para a detecção. Na saída obtêm-se uma lista com as localizações dos picos de onda R.

Em seguida, os primeiros 30 segundos do registro de ECG são inspecionados afim de detectar artefatos ou batimentos considerados anômalos. Tal procedimento é feito através de uma

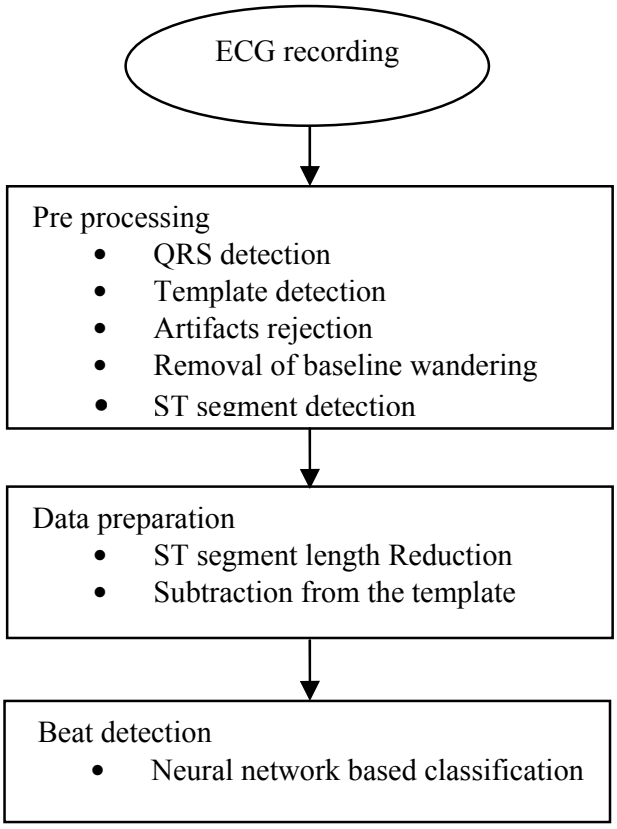


Figura 5.3 – Diagrama de blocos da estratégia proposta por Mohebbi e Moghadam. Extraído de (MOHEBBI; MOGHADAM, 2007)

estimativa do ponto isoelétrico e do ponto J de cada batimento. A estimativa se dá pela análise do gradiente do sinal. Caso nenhum dos dois pontos seja detectado, o batimento em questão é considerado como artefato e removido da lista. Os batimentos remanescentes são usados para construção do modelo, que é a média aritmética entre as amplitudes dos batimentos. Em seguida, o resto do sinal é processado e a linha de base é estimada através de interpolação *spline* cúbica, segundo o método descrito em (BADILINI; MOSS; TITLEBAUM, 1991).

Extração

Na etapa de extração, detecta-se o ponto J de cada batimento usando novamente o gradiente do sinal. Neste caso, uma janela de 100 ms à direita do pico de onda R é suavizada por um filtro de média móvel. Em seguida, busca-se na janela o primeiro intervalo de 20 ms cujo declive médio seja menor que 2,5 mV/s em valor absoluto. Toma-se o ponto central deste intervalo como o ponto J. Obtidos os pontos J, assume-se que os segmentos ST iniciem neste ponto e possuam largura pré-definida de 160 ms. O mesmo se aplica ao batimento modelo, do qual se extrai o segmento ST usado como referência no algoritmo.

Os dados que constituem a entrada para a etapa de classificação são a diferença entre o

segmento ST dos batimentos e o segmento ST de referência. Dada uma taxa de amostragem f_s, o segmento ST possui $\lfloor 0.16 f_s \rfloor$ amostras. Afim de reduzir o tamanho do conjunto de características, os sinais de segmento ST (inclusive o do modelo) são reamostrados para 20 amostras.

Classificação

Mohebbi e Moghadam também fazem uso de redes neurais artificiais para a etapa de classificação. Aqui, uma única rede neural é treinada usando o algoritmo de *backpropagation* com taxa de aprendizado adaptativa, combinado ainda com a técnica de treinamento com *momentum*. Os autores justificam o uso dessas técnicas argumentando que o desempenho do *backpropagation* é bastante sensível ao valor estabelecido para a taxa de aprendizagem, sendo que o valor ótimo deste parâmetro varia de acordo com a complexidade da superfície do erro. Dessa forma, uma taxa de aprendizado adaptativa se manterá no maior valor possível enquanto também permanece estável. O treinamento com *momentum* significa que a rede responderá não apenas ao gradiente local, mas também a tendências recentes na superfície do erro.

A camada de entrada da rede possui 20 neurônios, que recebem a sequência de valores oriunda da etapa de extração. A rede possui uma camada oculta também com 20 neurônios. A camada de saída contém dois neurônios, cujo valor de saída está no intervalo real de 0 a 1. As saídas da rede neural são o grau de pertinência a cada uma das classes (presença ou ausência de isquemia), sendo que o máximo entre as duas designa a classificação final. O treinamento da rede se encerra quando a soma do erro quadrático for menor que 0,01 ou quando o número limite de 2000 épocas é atingido.

5.4 Método de Gopalakrishnan et al.

O terceiro e último método estudado faz uso de uma série de propriedades da álgebra linear e também da relação entre as funções discretas de Hermite e a matriz de Fourier[1]. As etapas do algoritmo são ilustradas pelo diagrama da figura 5.4, sendo detalhada a seguir.

Pré-processamento

O método proposto não inclui nenhuma informação especial sobre procedimentos na etapa de pré-processamento, com exceção da própria detecção de batimentos cardíacos. Neste caso, os autores utilizam o mesmo algoritmo de detecção QRS mencionado para o caso do método anterior, qual seja, o algoritmo de Tompkins. Após obtenção dos picos de onda R, cada batimento deve ser centralizado neste ponto e sua largura é equivalente ao intervalo entre o seu pico e o do batimento vizinho (chamado intervalo R-R, ou apenas RR). Entretanto, técnicas como as de remoção da linha de base ou redução de ruído/interferência não estão presentes neste método.

[1] matriz de Fourier é aquela que, multiplicada por um vetor, resulta na DFT do vetor

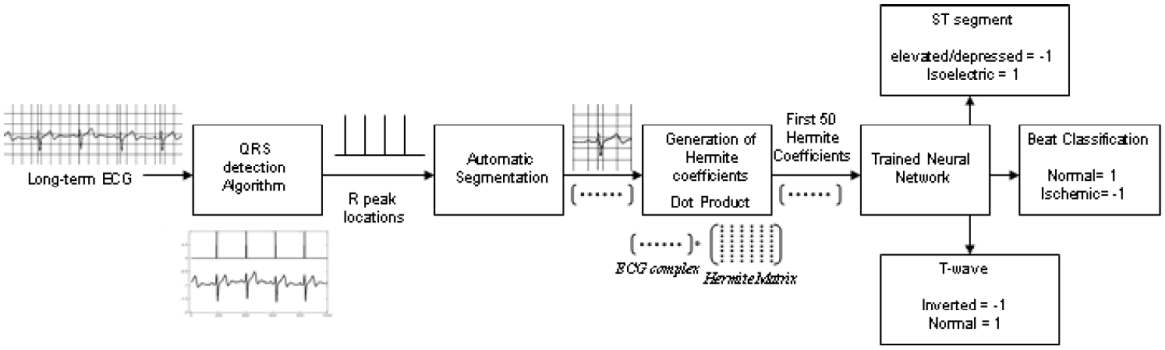


Figura 5.4 – Diagrama de blocos da estratégia adotada por Gopalakrishnan et al. Extraído de (GOPALAKRISHNAN; ACHARYA; MUGLER, 2004)

Extração

Para cada batimento, os primeiros 50 coeficientes de Hermite são extraídos, utilizando exatamente o mesmo ferramental introduzido no capítulo 3, no que diz respeito às funções discretas de Hermite. Estes coeficientes constituem, integralmente, o conjunto de características fornecido à etapa de classificação. Os autores ainda explicam, baseando-se em trabalhos passados e também pela observação do erro RMS (do inglês, *root mean square*) percentual, que o uso de 50 coeficientes e de um parâmetro de dilatação $b = 1$, garantem uma aproximação bastante boa.

Classificação

Gopalakrishnan et al. utilizaram cinco redes neurais em seu esquema de classificação. A justificativa é que o resultado de redes individuais pode variar em casos limítrofes de isquemia. Desse modo, o resultado majoritário de um comitê de cinco redes fornece a decisão final sobre a classe a que pertence um batimento. As redes possuem três camadas ocultas com diferentes números de neurônios em cada uma. A camada de saída possui dois neurônios, que indicam presença ou não de alterações no segmento ST e presença ou não de inversão da onda T.

Os autores não especificam um algoritmo para o treinamento das redes. Contudo, relatam a realização de um procedimento adicional em cada registro de ECG de longa duração. Eles explicam que as redes são retreinadas usando alguns exemplares de ciclo cardíaco normal do ECG e argumentam que isso fornece às redes um "toque" das características normais de segmento ST e onda T daquele registro em particular.

5.5 Resumo

Neste capítulo foram apresentados os métodos de detecção de isquemia estudados no trabalho. Primeiramente, algumas categorias foram usadas para agrupar os métodos de acordo com as principais técnicas de detecção utilizadas.

Depois, foi visto em detalhe o método proposto por Rocha et al., que emprega um procedimento sofisticado de segmentação de batimentos e também várias técnicas para eliminação de ruído, linha de base e extra-sístoles ventriculares (PVCs). O método sugere a utilização de dois grupos de características: o do desvio de segmento ST e o dos coeficientes da expansão em funções de Hermite. Particularmente, este método faz uso de análise espectral dos batimentos para obtenção do ponto J. Na etapa de classificação, cada derivação tem uma rede neural treinada somente com características advindas daquela derivação. As características usadas no treinamento das redes são a elevação e a depressão do segmento ST.

O método de Mohebbi et al., embora um pouco menos sofisticado, também emprega diversas técnicas de pré-processamento para remover a linha de base, detectar os pontos J e isoelétrico e eliminar artefatos. Em especial, este método constrói um modelo de batimento cardíaco e de segmento ST, do qual se obtém a diferença com os demais segmentos ST extraídos. A etapa de classificação deste método é bastante simples, já que emprega apenas uma rede neural para todas as derivações. Aqui também são usadas a elevação e a depressão do segmento ST no treinamento.

Por fim, foi apresentado o método proposto por Gopalakrishnan et al. Este poderia ser considerado o mais simples dos três, já que não faz uso de nenhuma técnica especial na etapa de pré-processamento. Sua etapa de extração também é simples, pois se resume a um produto matricial correspondendo à expansão em funções discretas de Hermite. A etapa de classificação do método emprega cinco redes neurais, sem que haja distinção entre as derivações, e computa o resultado como a decisão majoritária das cinco redes. Todas as características indicativas de isquemia são usadas no treinamento das redes deste método.

6 PROJETO E IMPLEMENTAÇÃO

Este capítulo descreve o projeto e a implementação dos métodos de detecção de isquemias cardíacas propostos por Rocha et al., por Mohebbi e Moghadam e por Gopalakrishnan et al.. Será abordado o projeto das três etapas dos métodos – pré-processamento, extração e classificação – assim como o projeto dos testes. Ao longo da discussão, será feito um relacionamento dos itens de projeto com a sua contra-parte na implementação (código-fonte). Ao final do capítulo será apresentado uma síntese sobre o projeto. O objetivo aqui é definir e mostrar como foi construído o ferramental utilizado para obtenção dos resultados práticos. Este trabalho foi realizado em cooperação com Guilherme Lazarotto de Lima, Mestrando em Computação pela UFRGS.

6.1 Projeto do pré-processamento

Como foi discutido no capítulo 5, cada método emprega técnicas variadas de pré-processamento do sinal de ECG. Não obstante essa diversidade, dois deles utilizam algum tipo de condicionamento do sinal (eliminação de ruído, linha de base, etc.) e todos se utilizam de algum algoritmo de detecção e segmentação de batimentos cardíacos. Um deles, o de Mohebbi e Moghadam, ainda emprega uma técnica de remoção de artefatos com base num *template*. Outro, o de Rocha et al., usa um procedimento de remoção de PVCs, adaptado de (COUCEIRO et al., 2008).

No início do projeto, fez-se uma tentativa de implementar cada método de detecção separadamente, cada qual utilizando suas técnicas, conforme proposto nos artigos originais. Contudo, essa abordagem se mostrou ineficaz e propícia a erros de implementação. Ineficaz porque há redundância de informações que são obtidas na etapa de pré-processamento, como filtragem do sinal para redução de ruído e interferências, remoção da linha de base, detecção de batimentos e delimitação das ondas do ciclo cardíaco. Propícia a erros de implementação porque cada técnica tem como base um algoritmo proposto em um artigo científico diferente (portanto, num contexto de processamento distinto, com sua própria fundamentação teórica).

Dado que o prazo para implementação, assim como o nível de conhecimento do aluno de graduação, são bastante limitados, não se podia consumir tempo demais na tentativa de reproduzir um método tal qual o original. Dessa forma, adotou-se uma abordagem que favorece a reutilização de código e simplifica o desenvolvimento.

Antes de mais nada, deve-se salientar que a detecção de batimentos cardíacos – também conhecida na literatura como detecção de QRS – aliada a um algoritmo de segmentação dos mesmos é, sem dúvida, uma ferramenta indispensável tanto para a etapa de pré-processamento como para a de extração. Sem ela não há como identificar os batimentos e, obviamente, se torna impossível inspecioná-los. Enquanto existem técnicas de processamento de sinais baseadas

em análise tempo-frequencial (espectrograma, Wavelet, Wigner-Ville), acredita-se que seja impossível identificar um episódio isquêmico usando apenas as informações de espectro do sinal, em lugar de extrair características de um batimento cardíaco bem delimitado. Esta intuição se corrobora pelo fato de que a grande maioria das metodologias de detecção de isquemia, como aquelas relacionadas no capítulo 5, se resume a algum tipo de inspeção de batimentos cardíacos (seja individualmente ou em coletividade).

Estratégia de implementação

A figura 6.1 ilustra a estratégia adotada para a etapa de pré-processamento. Note que esta configuração engloba a maioria dos procedimentos sugeridos pelos autores dos três métodos. Sua saída deve então conter todas as informações necessárias para dar sequência a qualquer dos métodos, na etapa de extração. Nesta figura, é possível ver que a saída da etapa de pré-processamento é composta por cinco itens de informação:

R uma lista com a localização dos picos de onda R no ECG;

RR uma lista de intervalos entre batimentos (intervalo R-R);

FP uma lista de pontos de interesse nos ciclos cardíacos;

Batimentos uma lista dos batimentos cardíacos extraídos; e

Template um modelo construído a partir de batimentos "normais".

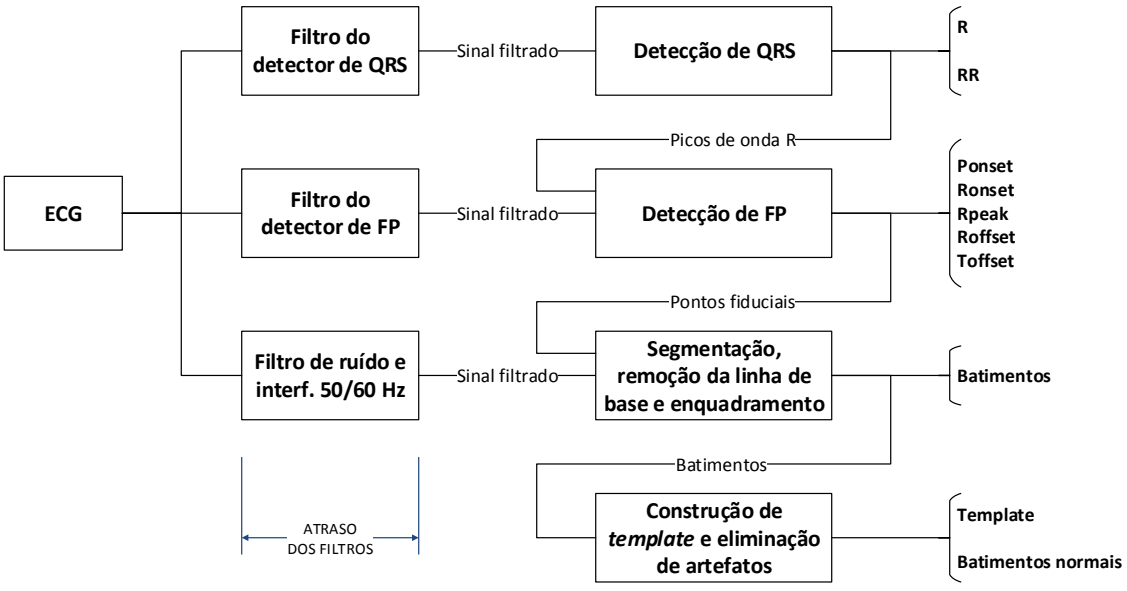


Figura 6.1 – Diagrama de blocos da estratégia de pré-processamento. Produzida no Microsoft Visio.

O primeiro item de informação provém da detecção de QRS. O algoritmo utilizado para tal fim é aquele descrito por Pan e Tompkins (PAN; TOMPKINS, 1985). Este algoritmo é também descrito no livro de um dos autores (TOMPKINS, 1993, pp. 245-262) e avaliado em outro artigo (HAMILTON; TOMPKINS, 1986). A escolha deste algoritmo segue a escolha feita por dois dos métodos de detecção de isquemia, o de Mohebbi e Moghadam e o de Gopalakrishnan et al., mas também se deve ao fato de que ele constitui uma técnica robusta e tradicional na área de processamento de sinais biomédicos.

Na prática, a lista R é desnecessária aos métodos de detecção de isquemia, pois, como veremos mais adiante, os batimentos são segmentados e embutidos num quadro (*frame*) de tamanho pré-definido. Sendo este tamanho um número ímpar e estabelecendo que os batimentos dentro do quadro estejam centralizados no pico de onda R, tem-se que o pico está sempre no ponto central do quadro, dispensando o uso da lista na etapa de extração.

Esta lista serve, porém, para apuração do resultado dos métodos, em termos das métricas introduzidas no capítulo 4. Sabemos que as anotações do ECG fornecem a localização dos batimentos conforme auditada por especialistas e, portanto, se faz necessário para fins de avaliação conhecer a localização obtida pelo detector de QRS. Dadas duas listas de pontos, há um algoritmo que faz o casamento cruzado entre elas. Isto é, que combina batimentos de uma lista com os da outra contanto que estejam muito próximos temporalmente. Este algoritmo é apresentado na rotina `match_qrs` (código-fonte A.33), escrita em código MATLAB.

O segundo item de informação não é simplesmente a diferença entre picos de onda R sucessivos, mas sim uma estimativa média do intervalo RR calculada a cada novo batimento. A estimativa tem portanto o efeito de "suavizar" a medida, eliminando variações demasiado abrutas de frequência cardíaca[1]. A lista RR provém do mesmo algoritmo de detecção de QRS, sendo necessária por dois motivos: primeiro porque o método de Rocha et al. emprega uma técnica de determinação do ponto J (definido logo adiante) fazendo uso da medida de frequência cardíaca; segundo porque Gopalakrishnan et al. lançam mão do intervalo RR na sua etapa de extração.

O terceiro item advém de um algoritmo proposto por Yan Sun et al. (SUN; CHAN; KRISHNAN, 2005). Originalmente, o procedimento faz a marcação dos seguintes pontos de interesse (ou *fiducial points*):

$\{P_{onset}, P_{peak}, P_{offset}\}$ pontos inicial, de pico e final da onda P;

$\{Q_{onset}, Q_{peak}, Q_{offset}\}$ pontos inicial, de pico e final da onda Q;

$\{R_{onset}, R_{peak}, R_{offset}\}$ pontos inicial, de pico e final da onda R;

$\{S_{onset}, S_{peak}, S_{offset}\}$ pontos inicial, de pico e final da onda S; e

[1] a frequência cardíaca (em batimentos por minuto) pode ser obtida a partir do intervalo RR, pela expressão $\frac{60 f_s}{RR}$, onde f_s é a frequência de amostragem do conversor A/D

$\{T_{onset}, T_{peak}, T_{offset}\}$ pontos inicial, de pico e final da onda T.

Contudo, apenas os pontos de início da onda P, os pontos da onda R e o ponto final da onda T são realmente necessários para os fins deste trabalho. R_{onset} e R_{offset} servem para realizar a busca de dois outros pontos de interesse: o isoelétrico e o *J*, definidos informalmente abaixo.

Definição 1 *Ponto isoelétrico é o local designado para o batimento onde este apresenta amplitude de "nível zero" antes do início da onda R. Isto é, o ponto à esquerda de R_{onset} que pode ser considerado como referência para a troca de polaridade das amplitudes do batimento cardíaco.*

Definição 2 *Ponto J é o local designado para o batimento onde o seu "traço" se torna mais horizontal do que vertical, após o término da onda R. Isto é, o ponto à direita de R_{offset} que pode ser considerado como início de um período de inatividade elétrica do coração.*

O ponto isoelétrico (aqui, apelidado de *I*) e o ponto *J* são obtidos por um algoritmo distinto, descrito por Daskalov em (DASKALOV; DOTSINSKY; CHRISTOV, 1998). A técnica em essência detecta as bordas do batimento cardíaco com auxílio da derivada de primeira ordem do sinal. Quem a sugere é Mohebbi e Moghadam, para uso na delimitação do segmento ST. Os mesmos pontos são necessários ao método de Rocha et al., por dois motivos: primeiro porque o ponto *I* servirá de referência para a estimativa do desvio ST, medido como a diferença entre as amplitudes no ponto *J* e naquele; segundo porque o ponto *J* serve como separador dos segmentos utilizados na etapa de extração do método (rever seção 5.2). Rocha et al., que sugerem o uso do algoritmo de Sun citado anteriormente, requerem os pontos inicial e final do batimento, P_{onset} e T_{offset}, principalmente por dois motivos: para remoção da linha de base e para delimitação dos segmentos usados na etapa de extração. A identificação da linha de base é feita por uma aproximação polinomial envolvendo as bordas de cada batimento, daí a necessidade destes pontos.

No método de Mohebbi e Moghadam, a remoção da linha de base seria alcançada através de interpolação *spline* cúbica, segundo um artigo de Badilini (BADILINI; MOSS; TITLEBAUM, 1991). A técnica requer conhecimento apenas da localização dos picos de onda R e é de fácil implementação para o caso de trincas – isto é, usando, além do batimento atual, o seu precedente e o sucessor. Contudo, num sistema de tempo-real, haverá necessidade de aguardar a detecção de um novo batimento cardíaco para que possa ser feita a interpolação para o batimento atual, o que incorre em um atraso equivalente a um batimento. No pior caso, o batimento seguinte pode nunca ser detectado. Ademais, a interpolação é mais precisa se se lançar mão das derivadas do sinal nas extremidades, implicando a necessidade de dois pontos adicionais (um antes do precedente e um depois do sucessor). Finalmente, a aproximação se deteriora à medida que a frequência cardíaca diminui, pois então o maior afastamento entre os pontos na interpolação prejudica o ajuste de amplitudes em torno do ponto central. Em virtude dessas desvantagens, neste trabalho foi escolhida a técnica anterior.

O quarto item de informação é constituído pelos próprios batimentos cardíacos, depois de segmentados, filtrados, sem linha de base e enquadrados num *frame* de tamanho fixo. O tamanho do *frame* foi estabelecido como $1,2$ vezes o número de amostras em 1 segundo do sinal. De acordo com (CLIFFORD; AZUAJE; MCSHARRY, 2006), a duração de um ciclo cardíaco está entre 0,6 e 1 segundo. Dando uma margem de tolerância de $0,2$ segundos ao valor máximo, garante-se que a vasta maioria dos batimentos caibam no *frame*. Caso o tamanho resultante seja um número par, o incrementa-se de 1 para que a condição previamente especificada seja satisfeita (pico de onda R no centro do emphframe).

A segmentação, portanto, consiste em delimitar o batimento cardíaco por seus pontos inicial e final, centralizá-lo pelo seu pico de onda R e enquadrá-lo no *frame*. O enquadramento remove qualquer trecho que esteja fora do quadro, tanto à esquerda quanto à direita do centro, bem como preenche com zeros caso "sobre" espaço no quadro. A remoção da linha de base ocorre antes deste procedimento, porque utiliza a informação completa do sinal do batimento. Conforme sugerido por Rocha et al., a técnica para identificação da linha de base é adaptada de Wolf (WOLF, 2004), que faz a interpolação usando um polinômio de primeiro grau.

O quarto item corresponde ao *template*, que é construído progressivamente ao longo da execução. O uso de *template* se faz necessário para o método de Mohebbi e Moghadam, que o utiliza para extrair um modelo de segmento ST. Os autores também o requerem afim de eliminar artefatos que foram identificados erroneamente como batimento pelo detector QRS. A eliminação se dá por meio de uma medida de erro, neste caso a distância Euclideana, obtida pela norma da diferença entre os vetores que descrevem o template e o batimento avaliado. Aproveitando-se dessa exigência, incorporou-se um algoritmo iterativo de construção do *template* à etapa de pré-processamento. Dessa forma, todos os métodos de detecção de isquemia se beneficiam da eliminação de artefatos, o que propicia uma comparação mais justa. Para o método de Mohebbi e Moghadam, que necessita de um *template* explicitamente, toma-se então a versão de *template* que ocorre depois de processados os primeiros 30 ciclos cardíacos.

A filtragem do sinal de ECG para remoção de ruído e interferência, conforme ilustra a figura 6.1, é realizada em paralelo com a detecção de QRS e de FP. Essa filtragem se dá pela aplicação de dois filtros IIR: um passa-baixas de Butterworth seguindo a especificação dada por Rocha et al., para redução de ruído de alta-frequência; e um rejeita-faixas Chebyshev tipo I com banda de rejeição de 2 Hz, para eliminação da interferência da rede elétrica. O projeto desses filtros, diferentemente do projeto dos filtros da detecção QRS e FP, é feito de antemão por intermédio do MATLAB, para várias frequências de amostragem comuns em eletrocardiografia.

Uma decisão de projeto que merece menção é o fato de ter-se aberto mão da transformada discreta de Wigner-Ville. Este procedimento era requerido pelo método de Rocha et al. para a localização dos pontos *I* e *J*. Entretanto, a implementação desta transformada em linguagem de programação tradicional não é trivial. Ela envolve o uso de variáveis complexas, da transformada de Fourier discreta (DFT) e da transformada de Hilbert, além de ser bastante custosa em termos

de processamento. O uso de variáveis complexas significa que deve-se trabalhar com dois vetores em memória, um para a parte real e outro para a parte imaginária do sinal. O uso da transformada de Fourier implica que devemos selecionar e incorporar um algoritmo de FFT (*Fast Fourier Transform*) ao ferramental do trabalho. A transformada de Hilbert, que, segundo os autores, é necessária para obtenção de um sinal analítico antes da aplicação da Wigner-Ville, parece ser tão cheia de peculiaridades que alguns professores a consideram matéria de pós-graduação. Esses fatores tornam a implementação mais complicada e demorada, não só pela complexidade mas porque todo código desenvolvido deve ser testado e validado.

Há ainda o fato de que, dado um vetor de tamanho n, são necessários n cálculos de DFT para obter a transformada de Wigner-Ville do vetor. Não bastasse isso, a matriz resultante deve passar por operações de valor absoluto e de soma, em que as linhas correspondentes a baixas frequências são superpostas para formar o vetor subjacente à busca dos pontos desejados. As duas últimas observações indicam que o procedimento exige bastante processamento. E de fato isso ocorre: fez-se uma tentativa de implementação usando a linguagem MATLAB (que possui funções prontas de DFT, Hilbert, manipulação de matrizes e de números complexos), e determinou-se que o tempo de processamento deste item perfazia mais da metade do tempo de extração de características do método.

Outra questão é que, nesta tentativa de implementação da transformada de Wigner-Ville, o resultado da identificação dos pontos I e J foi medíocre. O que ocorre é que os pontos de mínimo não são tão bem definidos e, muitas vezes, não existem dentro dos intervalos de busca (lembre-se de que os próprios autores não deixam claro os limites dos intervalos). Em virtude de todas estas considerações, decidiu-se não utilizar a transformada de Wigner-Ville para localização dos pontos I e J. Em vez disso, reaproveitou-se a técnica utilizada por Mohebbi e Moghadam (aquela que faz uso de diferenciação e *thresholding*).

Implementação

A implementação aconteceu de três formas, cada uma em um momento diferente. Primeiro criou-se um ambiente de trabalho usando a ferramenta computacional MATLAB. Neste caso, diversas rotinas de tratamento dos sinais de ECG foram implementadas em código MATLAB, e serviram para testar e validar o funcionamento da estratégia adotada. Em seguida, criou-se códigos na linguagem C que realizam as mesmas tarefas de maneira equivalente, e estes foram validados pela comparação de seus resultados com aqueles obtidos usando o código MATLAB. Num terceiro momento, construiu-se um modelo de simulação em tempo real, usando a ferramenta Simulink do MATLAB. Nesta implementação, utilizou-se a linguagem C++, que oferece recursos de orientação a objetos e permite uma melhor organização do código-fonte.

O desenvolvimento em C e em C++ se deu através do próprio MATLAB, que oferece uma API (*Application Programming Interface*) para uso e compilação de códigos C/C++ na sua plataforma.

Não serão feitas aqui referências ao código em C nem C++, apenas ao código MATLAB. Além disso, devido ao tamanho do trabalho, somente algumas rotinas em MATLAB serão apresentadas, todas no apêndice A desta monografia. Caso o leitor deseje visualizar o código completo, tanto em MATLAB quanto em C ou C++, refira-se ao seguinte link: `https://github.com/dsogari/tg-ecp-ufrgs`. Ali estará o repositório do trabalho desenvolvido disponível para *download* gratuito. Este repositório será atualizado constantemente até a conclusão do trabalho, o que pode não coincidir com a entrega desta monografia.

Os sinais de ECG são os da base de dados ST-T da Sociedade Européia de Cardiologia (TADDEI et al., 1992), obtidos a partir do *website* PhysioBank.org da PhysioNet (GOLDBERGER et al., 2000). A leitura dos arquivos da base foi realizada com auxílio do pacote de software WFDB, disponibilizado gratuitamente pelo mesmo *site*. Na verdade, para facilitar o trabalho no MATLAB, criou-se algumas rotinas que encapsulam as chamadas do WFDB e produzem arquivos .mat contendo todas as informações de um ECG. Assim, para ler um arquivo de ECG basta carregar o .mat correspondente usando as funções nativas do MATLAB, que são bastante simples e eficientes.

Uma estrutura em MATLAB contém as amostras do sinal e outras informações a respeito deste, como a frequência de amostragem, o valor de referência e a resolução do conversor A/D, anotações, entre outras. Esta estrutura serve como entrada para a etapa de pré-processamento. A etapa começa pela filtragem do sinal de ECG, descrita em detalhes a seguir.

A filtragem para detecção de complexos QRS se dá pela função `qrs_filter` (código A.1). Esta função recebe como parâmetros as amplitudes do sinal de ECG e a frequência de amostragem, f_s. Ela retorna o sinal filtrado e o atraso total dos filtros, d_{QRS}. Os filtros deste procedimento são quatro: um passa-baixas de segunda ordem[2] com frequência de corte em aproximadamente 11 Hz; um passa-altas de primeira ordem[3]. com frequência de corte em aproximadamente 5 Hz; um diferenciador de 5 pontos; e um filtro de média-movel com largura aproximada de 150 ms. A figura 6.2 mostra o diagrama de blocos da filtragem QRS, enquanto as figuras 6.3 e 6.4 ilustram os diversos sinais intermediários dessa filtragem.

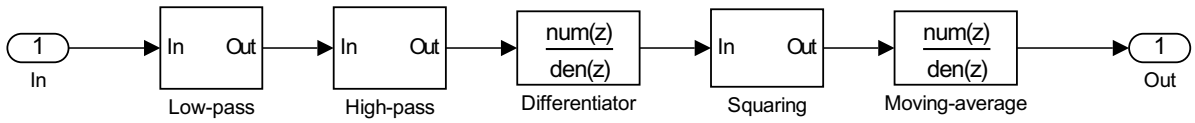
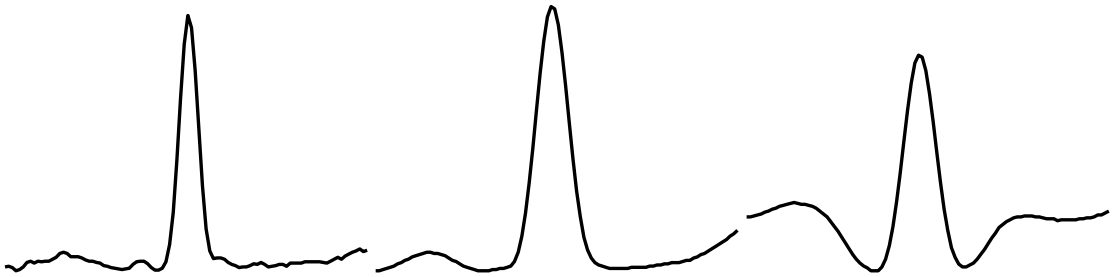


Figura 6.2 – Diagrama de blocos do filtro para detecção de complexos QRS.

A filtragem para detecção de pontos de interesse se dé pela função `fp_filter` (código A.2). Esta função recebe como parâmetros as amplitudes do sinal de ECG, a frequência de amostragem e o atraso d_{QRS}. Ela retorna dois sinais, um que corresponde à derivada de primeira ordem do

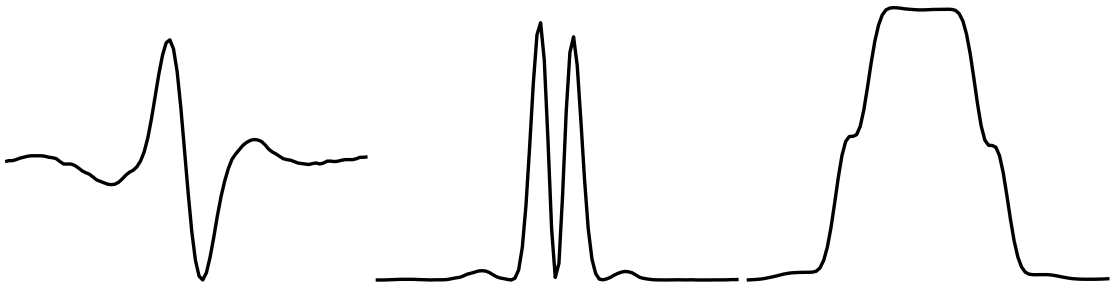
[2] esta ordem não diz respeito ao número de coeficientes do filtro, mas sim tem a ver com os coneitos apresentados no capítulo 3, no que tange à técnica de projeto de filtros FIR com coeficientes inteiros

[3] Ver nota de rodapé 2



(a) Sinal de ECG (b) Saída do passa-baixas (c) Saída do passa-altas

Figura 6.3 – Sinais intermediários da filtragem QRS.



(a) Saída do diferenciador (b) Saída do operador quadrado (c) Saída do média-móvel

Figura 6.4 – Sinais intermediários da filtragem QRS (continuação).

sinal de entrada suavizado e outro que é a derivada morfológica (SUN, 2002) do sinal suavizado. Os filtros deste procedimento são quatro: um passa-baixas de segunda ordem[4] com frequência de corte em aproximadamente 11 Hz; um filtro de média-móvel com largura de aproximadamente 50 ms; um diferenciador de dois pontos; e um filtro de média-movel com largura aproximada 150 ms.

A filtragem para remoção de ruído se dé pela função `noise_filter` (código A.3). Esta função recebe como parâmetros as amplitudes do sinal de ECG, a frequência de amostragem, a frequência da rede elétrica, f_m, e novamente o atraso d_{QRS}. Ela retorna o sinal filtrado, isto é, sem ruído de alta frequência e sem interferência da rede elétrica. Há dois filtros neste procedimento: um passa-baixas de Butterworth de quarta ordem com frequência de corte em aproximadamente 40 Hz; e um rejeita-faixas de quarta ordem Chebyshev tipo I, com frequência central em f_m e largura da banda de rejeição de aproximadamente 2 Hz.

Os atrasos das três filtragens seriam dados a grosso modo por: $d_{QRS} \approx 0.18 f_s$, $d_{FP} \approx 0.09 f_s$ e $d_{noise} \approx 0.01 f_s$. Entretanto, deseja-se "emparelhar" os sinais filtrados. Isto é, fazer com que eles estejam na mesma linha de tempo após a filtragem. Isso facilita o trabalho dos detectores na etapa de pré-processamento. Sabendo que d_{QRS} é o maior dos três atrasos, tem-se que as demais devem ser igualadas àquela. A equivalência de atrasos é alcançada pela introdução de filtros passa-tudo, com um atraso dado pela diferença entre d_{QRS} e d_{FP} ou entre d_{QRS} e d_{noise}. Para uma frequência de amostragem de 250 Hz, que é aquela da base de dados utilizada, tem-se

[4] Ver nota de rodapé 2

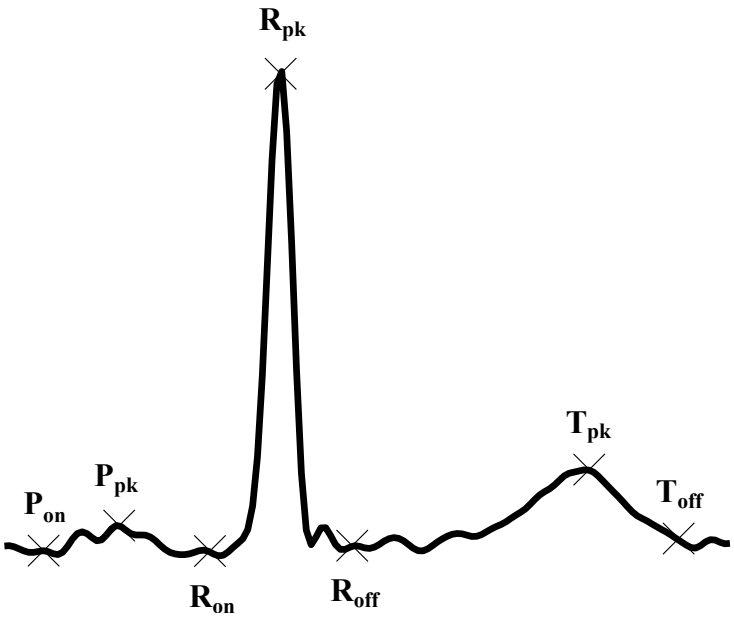
$d_{QRS} = 46.5$ amostras. Também vale dizer que a frequência da rede elétrica usada nos testes foi 50 Hz, uma vez que a base de dados é européia.

O próximo passo é detectar os batimentos. A função que realiza tal procedimento é `detect_qrs` (código A.4). Ela recebe como parâmetros as amplitudes do sinal filtrado pelos filtros QRS e a taxa de amostragem. Ela retorna a localização aproximada dos picos de onda R e uma estimativa média dos intervalos RR. Existem cinco componentes principais no laço deste algoritmo: detecção de picos, atualização de *thresholds*, determinação de complexos QRS, atualização de medidas de intervalo RR e busca de QRS em retrospecto (*searchback*). Para cada pico detectado, verifica-se se a amplitude é maior que um *threshold*. Se for, então o pico é candidato a ser de um complexo QRS. Medidas de nível de ruído e de sinal são atualizadas de acordo com a decisão tomada para o pico. O *threshold* é computado em função desses níveis. Caso nenhum complexo QRS seja detectado dentro de um determinado período, a busca em retrospecto é ativada para recuperar um possível complexo perdido. A localização do pico do complexo, que quase sempre corresponde ao pico de onda R, não é identificada exatamente pelo algoritmo. O importante é saber que há um complexo QRS nos instantes próximos ao pico detectado no sinal filtrado. A localização exata no ECG é obtida pela detecção de pontos fiduciais, descrita a seguir.

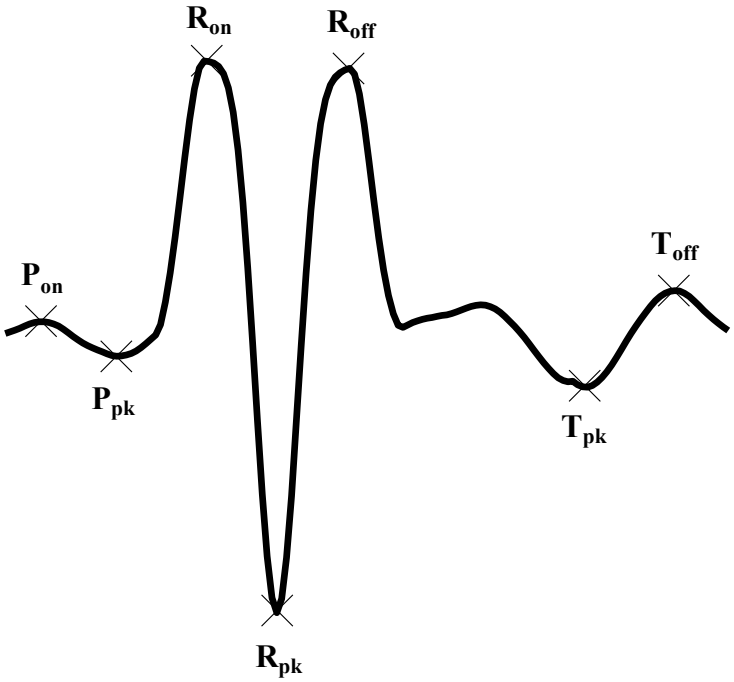
A detecção de pontos fiduciais se dá pela função `detect_fp` (código A.5). Ela recebe como parâmetros os dois sinais filtrados pela filtragem FP e a lista R fornecida pelo detector QRS. Sua saída é uma lista com os pontos fiduciais correspondentes a cada item da lista R. A busca é dividida em três partes. Em primeiro lugar são detectados os pontos referentes à onda R (R_{peak}, R_{onset} e R_{offset}). Em segundo lugar, tendo R_{onset}, detecta-se o pico e o início da onda P (P_{peak} e P_{onset}). Lembre-se de que P_{peak} não será necessário posteriormente, mas é imprescindível para a busca de P_{onset}. Por último, tendo R_{offset}, detecta-se o pico e o fim da onda T (T_{peak} e T_{offset}). Novamente, T_{peak} não será necessário para a etapa de extração, mas serve para encontrar T_{offset}. A busca é feita com ajuda de algumas rotinas especiais: `search_peak_abs`, `search_first_mark` e `search_best_mark`. Em particular, a busca dos picos (R_{peak}, P_{peak} e T_{peak}) desconsidera a polaridade do sinal. Isto é, um pico negativo é identificado da mesma maneira que um pico positivo, utilizando a rotina `search_peak_abs`. As figuras 6.5a e 6.5b ilustram os sinais de ECG e o da derivada morfológica, com a localização dos pontos fiduciais.

A segmentação dos batimentos pode agora ser realizada, pois tem-se as marcações P_{onset} e T_{offset}. Também com base nestes pontos, pode-se encontrar um polinômio de primeiro grau que melhor se ajusta à reta que conecta ambos. Mais especificamente, toma-se uma média das amplitudes nos cinco pontos próximos a P_{onset} e nos cinco pontos próximos a T_{offset}. Com as médias, calcula-se os parâmetros da reta que os liga. Em cada ponto do batimento, subtrai-se da amplitude original o valor da reta avaliado naquele ponto. Tem-se assim um batimento cuja linha de base é o próprio eixo horizontal de nível zero (ou isoelétrico). A figura 6.6 ilustra o procedimento de remoção (ou normalização) da linha de base.

Após a remoção da linha de base, o batimento é enquadrado num *frame*, da maneira como



(a) Sinal de ECG



(b) Sinal da derivada morfológica

Figura 6.5 – Sinais da filtragem FP e respectivos pontos.

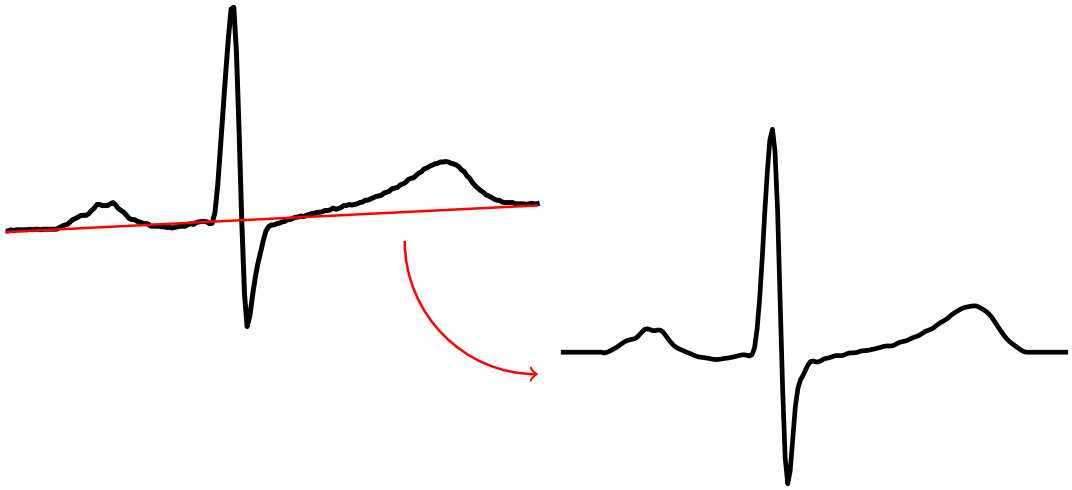


Figura 6.6 – Procedimento de remoção da linha de base.

foi explicada no início desta seção. A função que realiza estes procedimentos é `extract_beats` (código A.6), com suas rotinas auxiliares `polyfit_baseline` e `frame_beat`. Os parâmetros de entrada de `extract_beats` são as amplitudes do sinal filtrado pelos filtros de ruído, a lista FP e a taxa de amostragem. Sua saída é composta pela lista de batimentos extraídos e uma nova lista FP cujos valores dizem respeito à localização dos pontos de interesse no *frame* (e não mais no ECG).

Finalmente, toma lugar a construção de *template* e a remoção de artefatos. Para tanto, é usada a função `build_template` (código A.7). Esta função recebe como parâmetro a lista de batimentos e uma constante que diz a taxa de adaptação do *template*. Por exemplo, uma taxa de R significa que as amplitudes do *template* obedecem à regra $T_i[n] = T_{i-1}[n] + R^{-1}\left(B_i[n] - T_{i-1}[n]\right), 0 \leq n \leq N-1$, com N igual ao tamanho do *frame* dos batimentos. Esta é a equação de recorrência de um filtro de média-móvel exponencial com constante de decaimento igual a $1/R$. No entanto, o *template* só é atualizado se a diferença, E, entre o *template* e o batimento atual for menor que um *threshold*, C. Este *threshold* é calculado pela mesma relação recursiva: $C_i = C_{i-1} + R^{-1}\left(E_i - C_{i-1}\right)$.

O erro E_i é a medida de diferença na i-ésima iteração, dada pela distância Euclideana: $E_i = \|B_i - T_i\|$. Na verdade, dois valores de *threshold* são mantidos, um que equivale a 3 vezes o valor do erro médio e o outro, a 4 vezes o mesmo valor. Assim, se o erro na iteração atual for menor que o primeiro *threshold*, então o batimento é considerado muito bom e tanto o *template* quanto o *threshold* são atualizados. Senão, se o erro estiver acima do segundo *threshold*, então o batimento é considerado como artefato e descartado. Se o erro cair no intervalo entre os dois *thresholds*, o batimento é considerado bom e mantido sem que haja alteração de *template* ou de *threshold*. A figura 6.7 ilustra a comparação feita para determinar se um batimento é um artefato ou não.

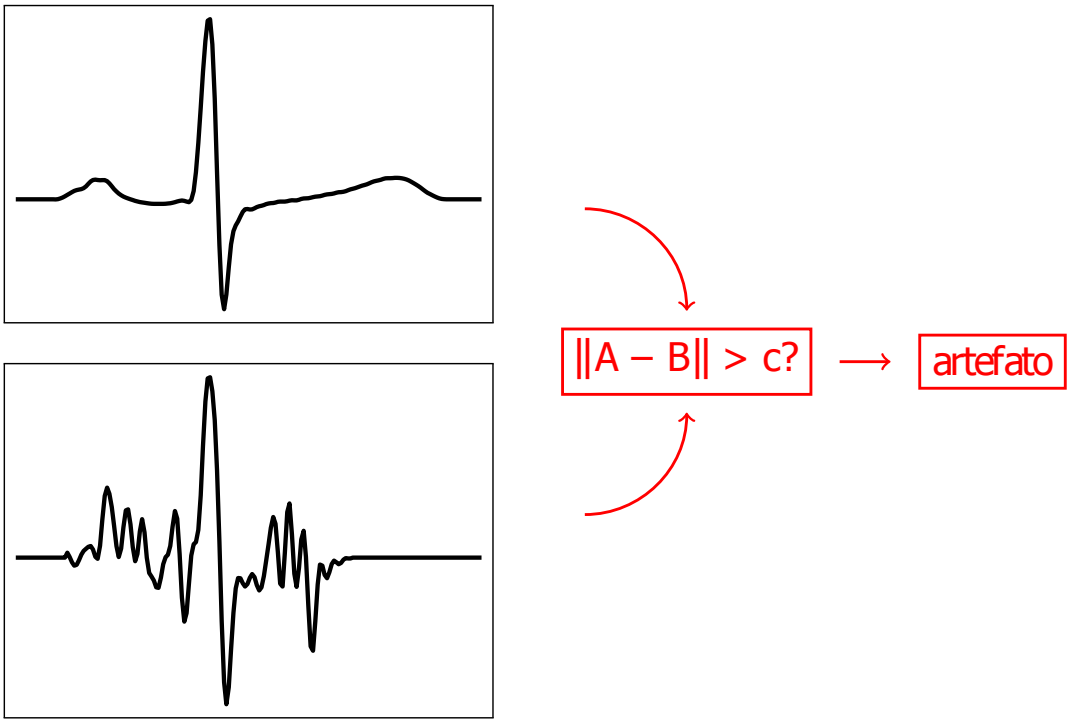


Figura 6.7 – Procedimento de remoção de artefatos.

6.2 Projeto da extração de características

Método de Rocha et al.

A extração de características do método de Rocha começa pela identificação do ponto J através da técnica de Pang (PANG et al., 2005). A função que realiza este procedimento é `pang_jpoint`. Ela recebe como parâmetro os intervalos RR e retorna a localização do ponto J de acordo com o intervalo em que se encontra a frequência cardíaca, conforme mostra a tabela 6.1.

Frequência cardíaca (em BPM)	Ponto de medida
< 100	$R + 120$ ms
$100 - 110$	$R + 112$ ms
$110 - 120$	$R + 104$ ms
> 120	$R + 100$ ms

Tabela 6.1 – Localização do ponto J determinada com base na frequência cardíaca.

Em seguida, o algoritmo de bordas é empregado para a busca do ponto isoelétrico e também para uma nova estimativa do ponto J. A função `edge_detection` (código A.12), que recebe como parâmetros o sinal do batimento, pontos de início e fim, além de um *threshold*, faz a busca do ponto desejado no intervalo, retornando este como saída. A busca do ponto I se dá pela aplicação da função no intervalo à esquerda de R_{peak}, enquanto a busca de J se dá no intervalo à direita de R_{peak}.

Com estes pontos, obtêm-se uma medida do desvio de segmento ST, dado pela diferença entre as amplitudes do batimento nos pontos *J* e aquela avaliada no ponto *I*. Essas duas medidas são as primeiras a compor o conjunto de características do método de Rocha et al. A figura 6.8 ilustra o procedimento de obtenção do desvio ST.

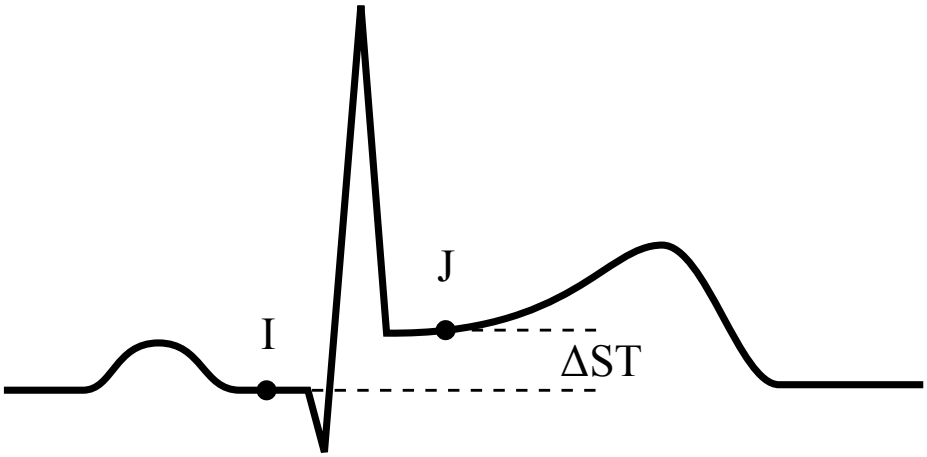


Figura 6.8 – Cálculo do desvio de segmento ST no método de Rocha et al.

O próximo passo é seccionar o batimento em dois segmentos: um contendo o complexo QRS e outro contendo a onda T. O primeiro segmento vai do ponto *I* até o ponto *J*, enquanto o segundo vai deste até o ponto final do batimento (T_{offset}). Cada um destes segmentos é então reamostrado para um sinal de 64 amostras. A função do MATLAB para reamostragem é `resample`. Ela recebe como parâmetros o sinal que se deseja reamostrar, um fator de interpolação p e um fator de dizimação q, retornando um sinal de largura $\lceil \frac{p}{q} \rceil$ vezes a largura original. Internamente, essa função faz o projeto de um filtro passa-baixas FIR, necessário ao algoritmo de reamostragem.

Na implementação em C da reamostragem, foi usada a técnica de projeto de filtros por truncagem da resposta ideal, conforme vista no capítulo 3. O filtro projetado possui as seguintes especificações: largura da região de transição de 2 Hz, frequência de corte em $\frac{\pi}{\max(p,q)}$ radianos e uso da janela Hamming (que apresenta atenuação média de 41 dB na banda de rejeição). Note que o projeto é refeito a cada nova chamada de reamostragem, pois a especificação depende do tamanho do sinal original. A figura 6.9 mostra a delimitação dos segmentos no batimento.

Os segmentos devidamente reamostrados servem como operando no produto matricial que fornece os coeficientes da expansão em funções de Hermite, sendo o outro operando a matriz discutida no capítulo 5 para o método de Rocha et al. Há uma matriz para o primeiro segmento e outra para o segundo segmento, cada qual com seu fator de dilatação (5 e 8, para ser exato). Ambas possuem 6 linhas e 64 colunas, de modo que o produto com o vetor coluna contendo as 64 amostras do segmento correspondente resulta nos coeficientes desejados. A figura 6.10 ilustra o procedimento de expansão em funções de Hermite e extração dos coeficientes.

Portanto, ao final da extração do método de Rocha et al., obtêm-se um conjunto de 14 características, quais sejam: duas medidas de desvio de segmento ST; seis coeficientes da

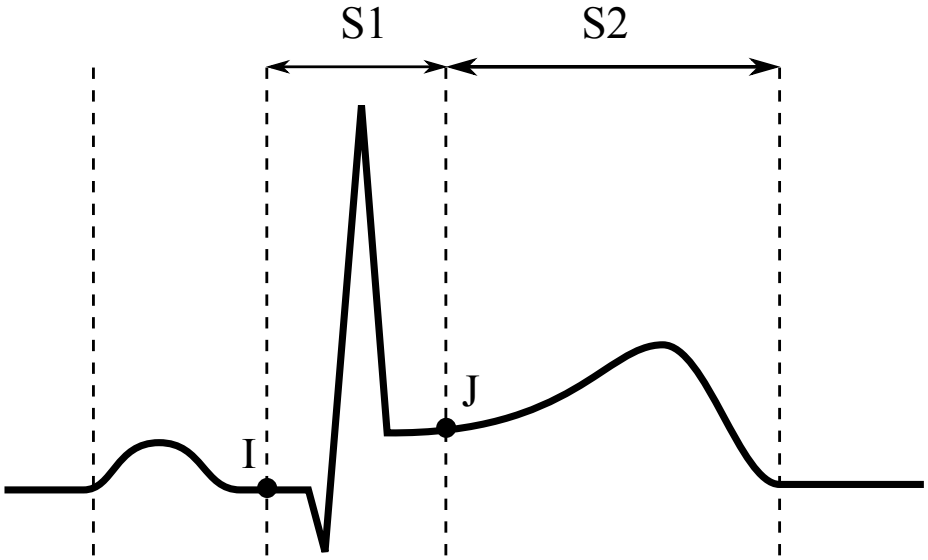


Figura 6.9 – Particionamento do batimento em dois segmentos para o método de Rocha et al.

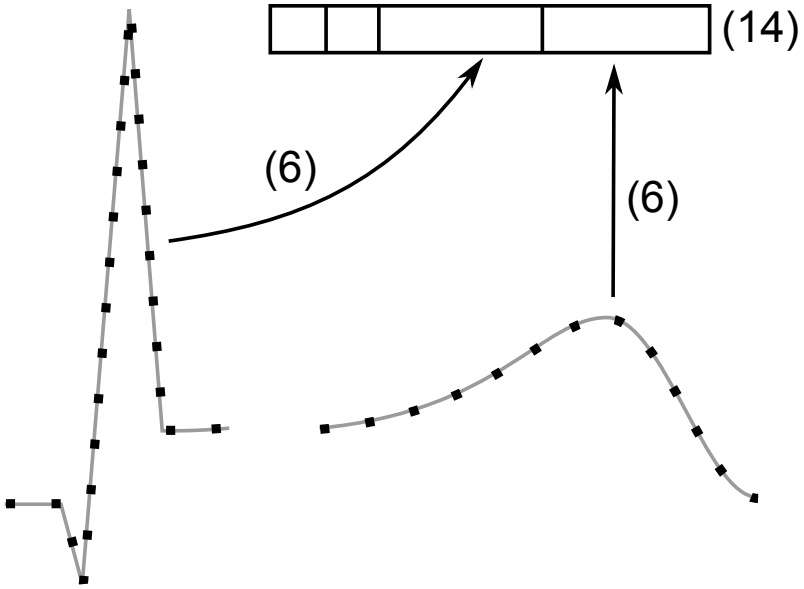


Figura 6.10 – Aproximação dos segmentos por funções de Hermite para o método de Rocha et al.

expansão de Hermite do primeiro segmento; e seis coeficientes da expansão do segundo segmento. A função responsável pelo garimpo de todos estes itens é `rocha_features` (código A.9), auxiliada pelas rotinas `pang_jpoints`, `rocha_ijpoints`, `rocha_stdev`, `rocha_segments` e `rocha_hermite` (códigos A.10, A.11, A.13, A.14 e A.15 respectivamente).

Método de Mohebbi e Moghadam

A etapa de extraçao do método de Mohebbi e Moghadam começa com a localização do ponto *J*, pelo mesmo procedimento já mencionado, o da função `edge_detection`. Depois, extrai-se o segmento ST, que inicia no ponto *J* e se estende por 160 ms (ou 40 amostras, numa frequência

de amostragem de 250 Hz). O segmento é reamostrado pra 20 amostras e sua diferença com o segmento ST do *template* é obtida por uma simples operação de subtração. A figura 6.11 ilustra como é feita a extração do segmento ST, enquanto a figura 6.12 mostra como é feita a composição do vetor de características do método.

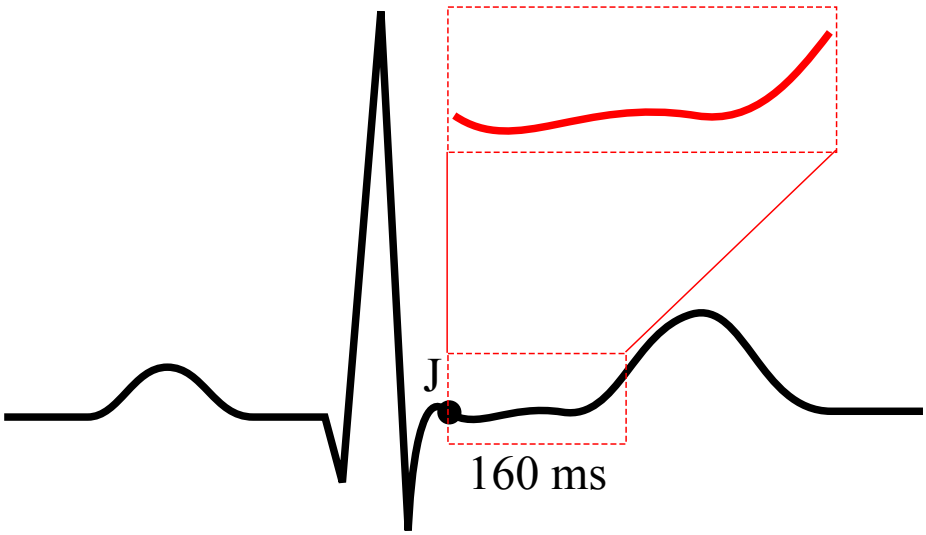


Figura 6.11 – Obtenção do segmento ST a partir do ponto *J* no método de Mohebbi e Moghadam.

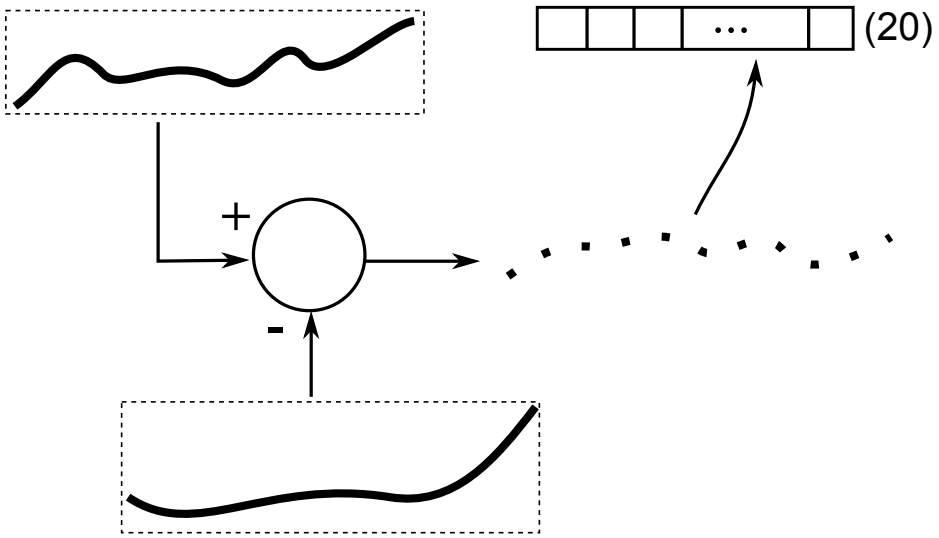


Figura 6.12 – Diferença entre o segmento ST de teste e o de referência, para composição das características do método de Mohebbi e Moghadam.

Note-se que o ponto *J* e o segmento ST do *template* são extraídos apenas uma vez, tão logo o *template* se faça presente. A versao de *template* utilizda é aquela computada após o 30º batimento. A função que executa estes procedimentos é `mohebbi_features` (código A.16), suportada pelas rotinas auxiliares `mohebb_ijpoints`, `mohebb_isegments` e `mohebb_istdiff` (códigos A.17, A.18 e A.19, respectivamente).

Método de Gopalakrishnan et al.

A extração do método de Gopalakrishnan et al. é bastante simples. Primeiro, um segmento é extraído contendo todo o batimento. Ele está centralizado no pico de onda R (ou o centro do *frame*) e possui largura igual ao intervalo RR correspondente ao batimento em questão. Subsequentemente, a função de reamostragem é aplicada para produzir um sinal de tamanho fixo igual a 250 amostras. A figura 6.13 mostra como é feita a delimitação do batimento, enquanto a figura

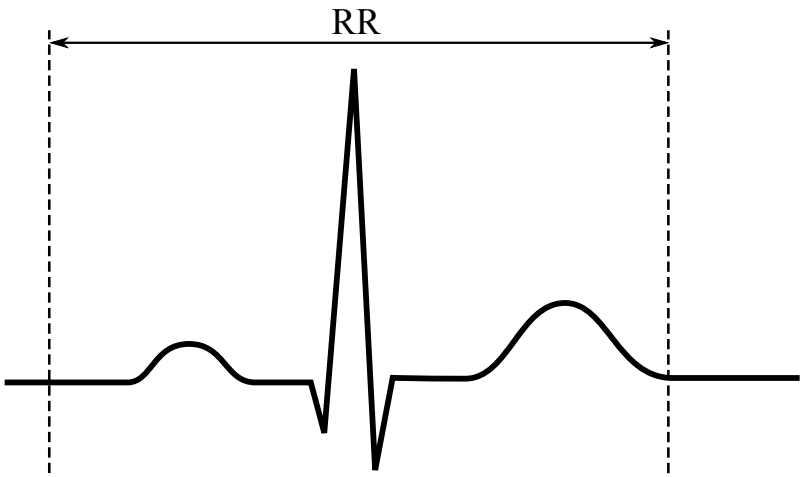


Figura 6.13 – Delimitação do batimento e centralização no intervalo RR, para uso o método de Gopalakrishnan et al.

Finalmente, ocorre o produto matricial entre a matriz de funções discretas de Hermite (ver capítulo 3), com 50 linhas e 250 colunas, e o vetor coluna com as 250 amostras do batimento. O resultado é um conjunto de 50 coeficientes, que compõem o conjunto de caracteristicas do método. A função que realiza estes procedimentos é `gopalak_features` (código A.20), juntamente com as rotinas auxiliares `gopalak_segments` e `gopalak_hermite` (códigos A.21 e A.22, respectivamente). A figura 6.14 ilustra o procedimetno de aproximação do batimento em funções de Hermite discretas e extração dos respectivos coeficientes.

6.3 Projeto da classificação

Para a etapa de classificação, cada autor sugere uma estrutura diferente de redes neurais, bem como procedimentos distintos de treinamento das mesmas. As tabelas 6.2 e 6.3 mostram os diversos aspectos dos três esquemas de classificação propostos.

Assim como na etapa de pré-processamento, isso gera um problema prático: a implementação fica difícil e propensa a erros, pois a quantidade de código a ser escrito, mantido e testado é maior. Há novamente aqui uma motivação para criar um código único, que sirva para todos os métodos. Não apenas essa abordagem tende a tornar mais justa a comparação entre os métodos, mas facilita a implementação e encurta o tempo de desenvolvimento, do qual se carece. A solução

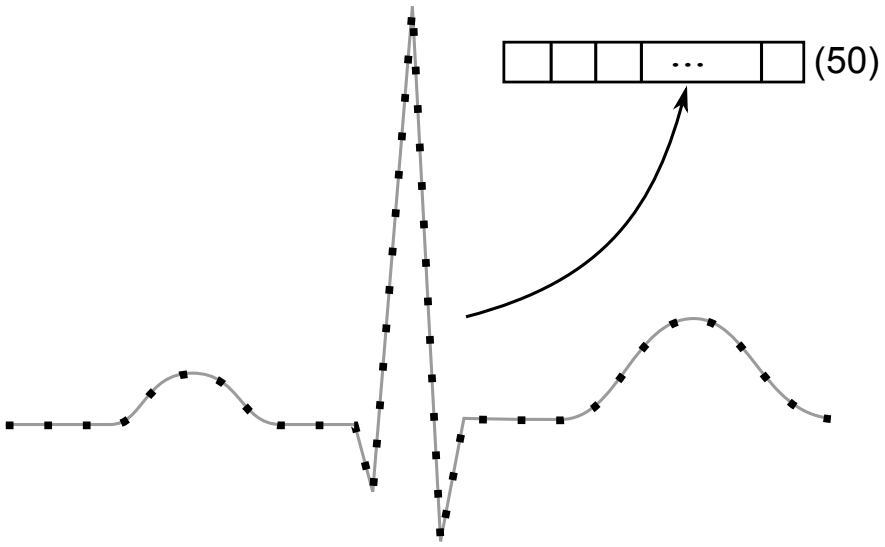


Figura 6.14 – Aproximação do batimento por funções discretas de Hermite e composição do vetor de características do método de Gopalakrishnan et al.

Item \ Método	Rocha et al.	Mohebbi e Moghadam	Gopalakrishnan et al.
Função de transferência	tangente sigmóide	log-sigmóide	tangente sigmóide
Função de normalização	não especificado	não especificado	não especificado
Nº de entradas	14	20	50
Nº de saídas	1	1	2
Intervalo da saída	[-1,1]	[0,1]	[-1,1]
Nº de camadas ocultas	2	1	1
Nº de neurônios	depende da derivação	20	não especificado
Nº de redes	2	1	5

Tabela 6.2 – Especificações da estrutura das redes neurais segundo cada autor.

então é combinar as diversas escolhas dos autores e encontrar uma combinação boa e conveniente para o esquema de classificação final. A seguir são descritas as escolhas feitas e o porquê de cada uma.

- Algoritmo de treinamento `trainlm`: é o padrão do MATLAB; usado pelo método de Rocha et al.; é geralmente o mais rápido a convergir, segundo a documentação do MATLAB; mas principalmnte, fez-se tentativas de treinamento usando outros algoritmos, como `traingdx` e `trainscg`, e os resultados pare estes foram geralmente inferiores.
- Função de transferência tangente sigmóide (`tansig`): usado por dois dos métodos.
- Função de normalização `mapstd` (média 0 e desvio padrão 1): segundo colegas de trabalho, é a mais comumente utilizada.

Item \ Método	Rocha et al.	Mohebbi e Moghadam	Gopalakrishnan et al.
Função de treinamento	Levenberg-Marquardt (`trainlm`)	*Backpropagation* adaptativo (`traingdx`)	Gradiente conjugado (`trainscg`)
Característica utilizada	ST	ST	ST e T
Proporção de ocorrência	não especificado	0.169	não especificado
Registros utilizados	conferir tabela 6.4	conferir tabela 6.4	não especificado
Divisão do conjunto para *cross-validation*	não especificado	não especificado	não especificado
Nº de amostras	depende da derivação	18047	236
Considera derivações separadamente	sim	não	não

Tabela 6.3 – Especificações do esquema de treinamento segundo cada autor.

Derivação \ Método	Rocha et al. (descartados)	Mohebbi e Moghadam (utilizados)
D3	–	–
MLI	e0207	–
MLIII	e0109, e0121, e0609, e0613	e0103, e0105, e0113, e0119, e0147
V1	e0403	–
V2	e0415 e0603	–
V3	–	–
V4	e0119 e0161	e0103, e0105, e0113, e0119, e0147
V5	e0207, e0213, e0303, e0405	–

Tabela 6.4 – Seleção de registros da base pelos métodos.

- Divisão aleatória dos dados na proporção 75/15/15: como nenhum dos autores explicita a sua escolha para fins de *cross-validation*, será utilizada a divisão padrão do MATLAB, que é de 75% para treinamento, 15% para validação e 15% para teste (de modo aleatório).

- 2 camadas ocultas com número aleatório de neurônios: infelizmente, não se dispôs de tempo para realizar uma identificação da estrutura ótima, pois este procedimento requer testes exaustivos com vários tipos de redes. Entretanto, privilegiando a proposta de Rocha et al., que é o método que mais informações expõe sobre a parte de classificação, e que também parece obter os melhores resultados (conforme será visto no capítulo 7), decidiu-se utilizar duas camadas ocultas com número aleatório de neurônios.

- Saída indica presença ou ausência da característica: tanto a característica de desvio de segmento ST quanto a da onda T na verdade são decompostas em outras 2, que são a elevação ou depressão do segmento ST e a elevação ou depressão da onda T. As possíveis classes então seriam aquelas dadas na tabela 6.5. No entanto, considerando as redes com saída única, tem-se um número elevado de redes a serem treinadas. Portanto, decidiu-se

utilizar a presença ou ausência de cada característica separadamente, mas sem distinguir entre elevação ou depressão. O resultado são duas redes: uma indicando presença/ausência de desnivelamento ST e a outra, presença/ausência de desnivelamento da onda T.

- Uso das proporções de ocorrência da base: apenas um dos autores sugere a composição do conjunto de dados usando uma proporção específica de batimentos normais *versus* isquêmicos. Assim, decidiu-se levantar estatísticas sobre a proporção de ocorrência de cada característica na base, e estas servirão como referência na seleção dos dados. Mais detalhes serão discutidos na seção 6.4.

- Uma rede para cada derivação: na implementação das etapas de processamento anteriores, percebeu-se que as formas de onda – e, consequentemente, as características extraídas – de diferentes derivações são bastante díspares; Rocha et al. também apontam este fato e dizem ser a separação uma boa escolha para melhorar os resultados. Como a base possui ECGs de 8 derivações diferentes, tem-se um total de 8 redes para cada método.

- Seleção dos dados de acordo com resultados de ECGs individuais: num primeiro momento, os conjuntos de dados continham exatamente os mesmos registros de ECG sugeridos por cada autor (ver tabela 6.4). Contudo, os resultados obtidos foram pífios. Ao fazer o treinamento das redes para cada registro individualmente, notou-se que muitos dos resultados eram bons, enquanto outros estavam aquém do esperado. Concluiu-se que os vetores de características da fase de extração para estes últimos não representavam corretamente os batimentos detectados. Porquanto não há uma maneira simples e automatizada de avaliar a consistência dos dados extraídos, optou-se por utilizar os próprios resultados individuais como critério de seleção. Mais detalhes serão discutidos na seção 6.4.

ST \ T	Normal	Elevado	Deprimido
Normal	C_{11}	C_{12}	C_{13}
Elevado	C_{21}	C_{22}	C_{23}
Deprimido	C_{31}	C_{32}	C_{33}

Tabela 6.5 – Possíveis classes do procedimento de classificação.

Há que se mencionar que os dois primeiros métodos não utilizam as características de desvio da onda T no seu procedimento de classificação. Rocha et al. deixam claro esta escolha, enquanto Mohebbi e Moghadam não explicitam a sua. Acredita-se que estes tenham utilizado apenas o desvio de segmento ST, pois é com base neste segmento que o vetor de características do método é extraído. Gopalakrishnan et al., por outro lado, dizem claramente que utilizam as duas características (ST e T). Assim, tem-se um total de (2 métodos $\times$ 1 característica $+$ 1 método $\times$ 2 características) $\times$ 8 derivações $=$ 32 redes neurais a serem treinadas para uma dada seleção de conjuntos de dados.

A classe a que pertence um batimento é dada pela condição $y > 0$, onde y é um valor real entre -1 e 1, correspondente à saída de uma rede neural. A condição satisfeita indica que a característica observada está presente, e o contrário é verdadeiro ($y \leq 0 \rightarrow$ característica ausente). Para julgar se um batimento é isquêmico, basta perguntar se $y > 0$ nos dois primeiros métodos, ou perguntar se $y_1 > 0 \parallel y_2 > 0$ no método de Gopalakrishnan et al., pois este tem duas redes (ST e T).

Outra questão importante é manter a proporção de ocorrência das características na população quando se faz uma seleção dos dados de treinamento das redes neurais. Este é o conceito de **estratificação**. Se existir muito mais batimentos com a característica observada do que realmente existe na população, a probabilidade de que a rede produza muitos falsos positivos é grande. Portanto, deve-se procurar manter a proporção original da população (que seria a base de ECGs).

Alguns códigos da etapa de classificação em MATLAB são apresentados no apêndice A.3. As rotinas principais de treinamento são `train_network` e `train_networks` (códigos A.24 e A.25); as rotinas `generate_datasets` e `generate_lead_dataset` (códigos A.26 e A.27) são relativas à seleção de dados para composição dos conjuntos de treinamento, a ser discutida na próxima seção. A figura 6.15 dá uma ideia geral de como fica etapa de classificação.

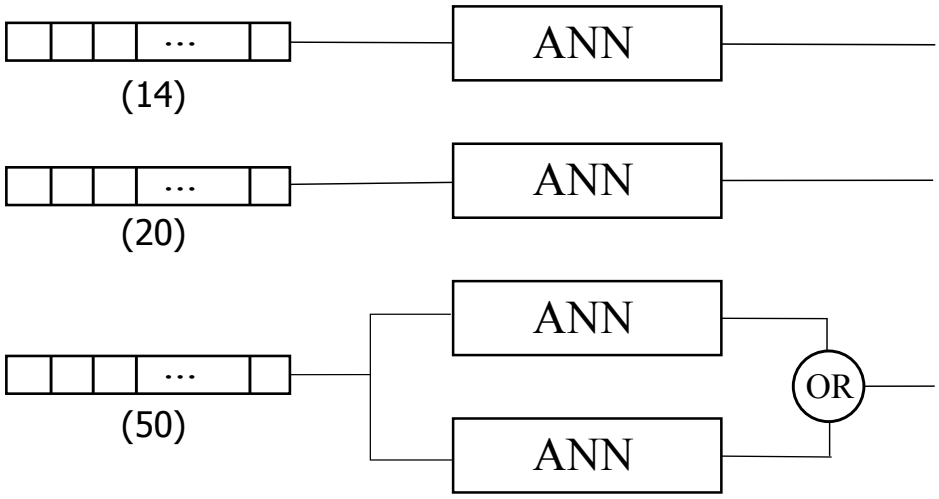
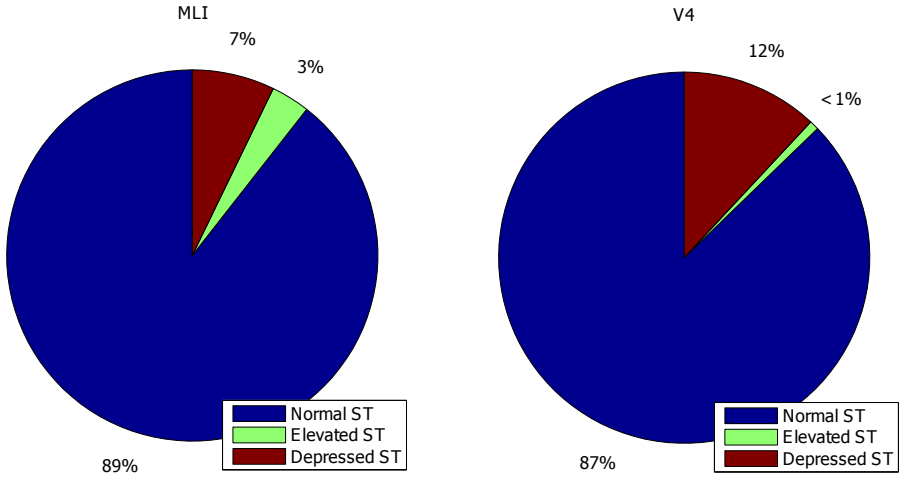


Figura 6.15 – Visualização da etapa de classificação dos métodos.

6.4 Projeto dos testes

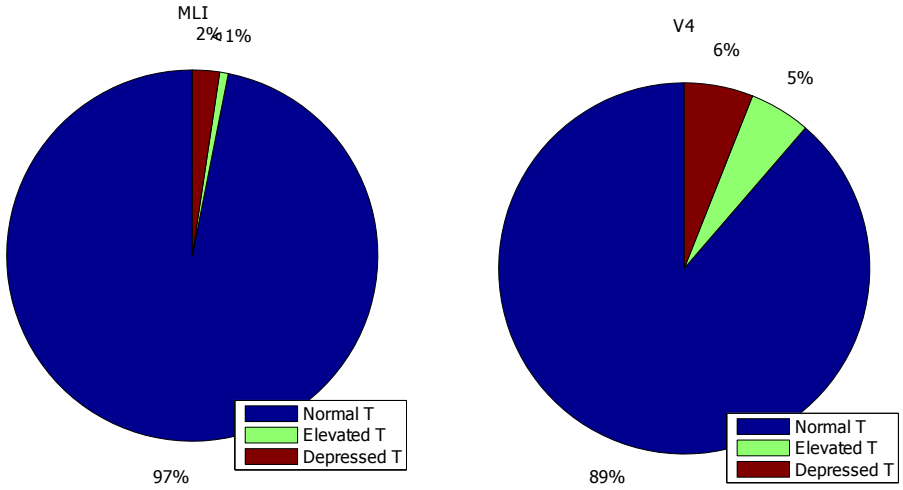
Conforme discutido anteriormente, fez-se necessário o levantamento das proporções de ocorrência das características de isquemia (ST e T) na base de ECGs. Já foi mencionado que a base contém registros de 8 derivações diferentes. Cada uma destas apresenta o seu percentual de ocorrência tanto no quesito ST como no T. A título de exemplo, as figuras 6.16a e 6.16b mostram num gráfico do tipo "pizza" os percentuais encontrados para a característica ST nas derivações MLI e V4, respectivamente. Da mesma forma, as figuras 6.17a e 6.17b ilustram a distribuição de

ocorrência da característica T. O mesmo estudo foi feito para as demais derivações, de modo a possuir todos os percentuais de ocorrência necessários à construção dos conjuntos de dados.



(a) Derivação MLI (b) Derivação V4

Figura 6.16 – Gráficos de distribuição da característica ST na base. Produzidos no MATLAB.



(a) Derivação MLI (b) Derivação V4

Figura 6.17 – Gráficos de distribuição da característica T na base. Produzidos no MATLAB.

Foi discutido também que a proposta original dos autores para seleção dos dados não rendeu resultados satisfatórios. Portanto, far-se-á um estudo para determinar quais os registros de ECG da base que mais prejudicam o desempenho geral. Quer-se então criar 5 configurações diferentes de seleção de conjuntos de dados para treinamento:

Configuração 1 proposta original dos autores, mantendo inclusive o número de batimentos.

Configuração 2 proposta criada com base no desempenho obtido com redes individuais.

Configuração 3 da configuração anterior, faz-se uma junção (ou *merger*) entre as listas dos diferentes métodos, privilegiando a decisão da maioria.

Configuração 4 mesma seleção da configuração 3, só que restringindo o número de batimentos selecionados (mas mantendo a estratificação).

Configuração 5 conjunto completo extraído da base, sem descarte

Nas configurações 1 e 2, cada método possui sua própria seleção de registros e número de batimentos, enquanto nas demais a seleção se mantém igual para os três métodos. Um resumo desta estratégia pode ser visualizado na figura 6.18. Na figura, os canais 0 ou 1 dizem respeito a um dos vetores de amostras de ECG contidos num registro da base (todos contêm 2 canais).

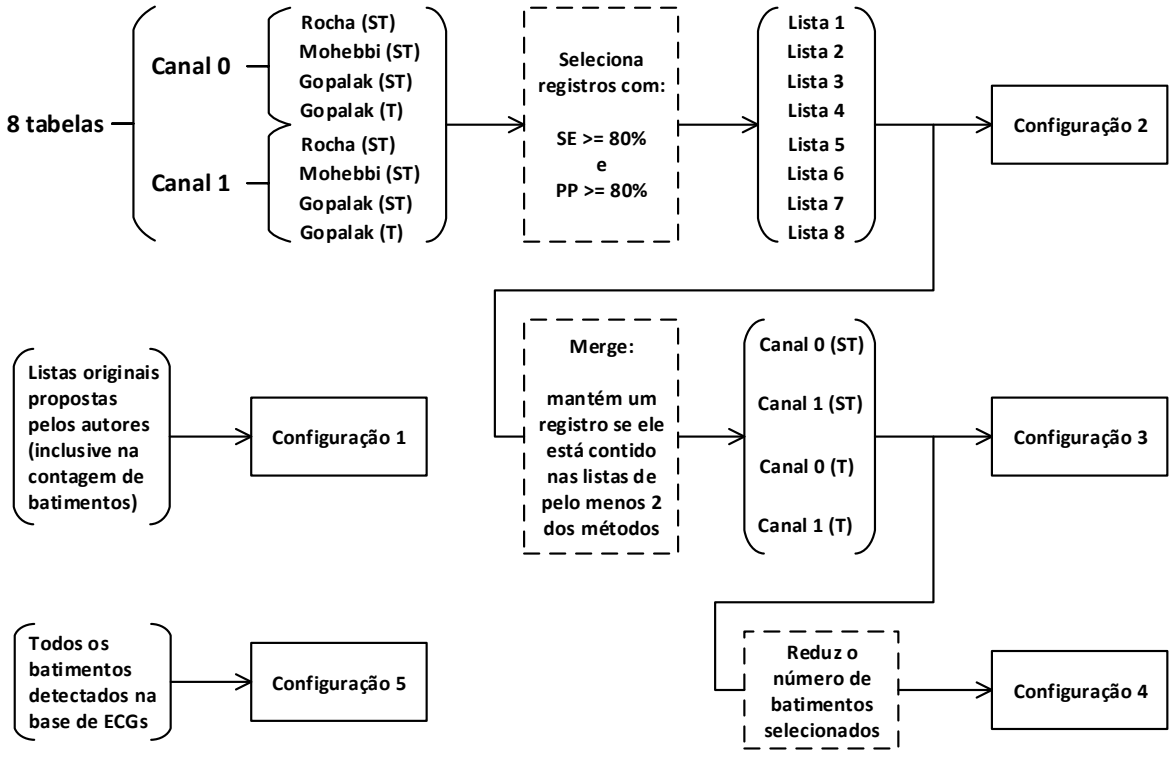


Figura 6.18 – Esquema de seleção de conjuntos de dados para o treinamento das redes neurais. Produzido no Microsoft Visio.

Com esta estratégia, haverá 5 configurações $\times$ 32 $=$ 160 redes neurais a serem treinadas. Também seria necessário apresentar, no capítulo seguinte, 8 tabelas de resultados individuais e 20 tabelas de resultados coletivos, pois são 5 configurações $\times$ 4 instâncias de treinamento e teste (Rocha et al. para ST, Mohebbi e Moghadam para ST e Gopalakrishnan et al. para ST e T). A fim de não poluir o espaço do próximo capítulo com diversas tabelas, será feita uma análise um pouco diferente: primeiro será apresentada uma tabela de exemplo contendo a primeira instância

de teste; em seguida serão mostradas tabelas contendo o resultado médio de todos os registros para o caso individual e de todas as derivações para o caso coletivo. Dessa forma, haverá cerca de 9 ou 10 tabelas concisas, contendo as informações relevantes para uma análise comparativa de confiabilidade dos métodos.

6.5 Resumo

Neste capítulo foram apresentados os projetos das etapas de pré-processamento, extração e classificação, bem como um projeto dos testes a serem realizados para comparação de desempenho dos métodos.

Primeiramente, na etapa de pré-processamento, viu-se que há a quatro procedimentos principais: a detecção de complexos QRS; a detecção de pontos de interesse; a segmentação e eliminação da linha de base; e por último a construção de *template* e eliminação de artefatos. Os três primeiros são precedidos, cada qual, por uma filtragem do sinal de ECG. A filtragem realça características que são importantes ao procedimento correspondente. O resultado desta etapa são os batimentos cardíacos, os intervalos RR, os pontos de interesse em cada batimento e também um *template* de batimento normal. Vale lembrar que os procedimentos desta etapa são os mesmos para todos os métodos de detecção de isquemia.

Na etapa de extração, os batimentos submetem-se às técnicas de processamento propostas pelos autores dos métodos, embora com algumas modificações. No método de Rocha et al., o desvio de segmento ST é calculado com base nos pontos I e J, e os coeficientes de Hermite são obtidos após uma nova segmentação do batimento em duas partes, com subsequente reamostragem para limitar o número de amostras. O vetor de características deste método contém 14 valores. No método de Mohebbi e Moghadam, o ponto J serve como referência para obtenção do segmento ST, que também é reamostrado. O vetor de características deste método tem tamanho 20 e corresponde à diferença entre o segmento ST testado e o do *template*. Por fim, o método de Gopalakrishnan et al. utiliza o intervalo RR para limitar ainda mais o batimento dentro do quadro e fazer a reamostragem para 250 amostras. Subsequentemente, produz-se 50 coeficientes na expansão de Hermite, compondo estes o vetor de características do método.

A etapa de classificação merece atenção especial, pois cada autor sugere uma estratégia diferente. Com o intuito de concluir o trabalho em tempo hábil, optou-se por uniformizar o esquema de treinamento e classificação. As redes possuirão saída única, indicando presença ou ausência de uma determinada característica de isquemia. Apenas o método de Gopalakrishnan et al. utiliza-se de duas características, viz. o desvio de segmento ST e a inversão da onda T, enquanto os demais utilizam apenas aquela relativa ao segmento ST. Dessa forma, tem-se 4 grupos de treinamento, cada qual com 8 redes neurais (uma para cada derivação de ECG), totalizando 32 redes a serem treinadas. Porém, conforme foi visto na seção 6.4, serão criadas 5 configurações diferentes de seleção dos conjuntos de dados de treinamento, fazendo com que o

total suba para 160 redes.

Ainda na seção 6.4, abordou-se a questão da estratificação dos dados, que está relacionada ao percentual de ocorrência de cada característica na população de ECGs. Uma levantamento foi feito para descobrir as diversas proporções de ocorrência tanto do desvio de segmento ST quanto do desvio da onda T.

Este capítulo constitui a parte mais importante do trabalho, pois foi aqui que as diversas escolhas de projeto e de implementação foram descritas e devidamente justificadas.

7 TESTES E RESULTADOS

Aqui serão apresentados os resultados dos testes realizados. Quer-se num primeiro momento verificar se as medidas obtidas são compatíveis com aquelas apontadas pelos autores. A tabela 7.1 resume os valores das medidas originais de confiabilidade dos métodos de detecção de isquemia. A seguir será visto o desempenho dos métodos para instâncias de teste em que as redes neurais foram treinadas para cada registro de ECG individualmente. Subsequentemente, verá-se como os métodos se comportam em instâncias de teste onde o treinamento é feito utilizando conjuntos amplos de batimentos cardíacos, obtidos de diversos registros da base. Por fim, apresentar-se-á algumas informações adicionais sobre os testes.

Método	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha et al. (ST)	98.85	–	98.30	–	–	–
Mohebbi e Moghadam. (ST)	97.22	–	98.80	–	–	–
Gopalakrishnan et al. (ST)	97.25	98.61	99.15	95.55	97.76	2.24
Gopalakrishnan et al. (T)	98.33	93.32	96.09	97.11	96.45	3.55

Tabela 7.1 – Estatísticas obtidas pelos autores em seus artigos.

7.1 Estatísticas individuais

A tabela 7.2 é um arranjo com as medidas de confiabilidade obtidas para cada ECG usando redes neurais treinadas somente com os dados extraídos do ECG em questão. De acordo com o que foi discutido no capítulo 6, estes valores fornecem um critério para determinar quais dos registros tiveram maior sucesso na etapa de extração. No caso da tabela 7.2, os registros eliminados seriam: e0136, e0139, e0170, e0204, e0208, e2011, e0213, e0415, e0418, e0602, e0609, e0615, e0808, e0818 e e1302. Estes são, portanto, os que fornecem dados inconsistentes para um treinamento de redes neurais mais abrangente (no método de Rocha et al., para o canal 0). Os valores NaN indicam que a característica observada não estava presente.

Tabela 7.2 – Estatísticas individuais do método de Rocha et al. para o canal 0.

Nome do registro	**SE (%)**	**ES (%)**	**PP (%)**	**PN (%)**	**AC (%)**	**TF (%)**
e0103	NaN	100.0	NaN	100.0	100.0	0.0
e0104	92.5	99.8	95.7	99.7	99.6	0.4
e0105	99.1	99.3	98.4	99.6	99.2	0.8
e0106	86.0	100.0	94.2	99.9	99.8	0.2
e0107	93.4	99.3	91.3	99.5	98.9	1.1
e0108	98.5	99.9	98.5	99.9	99.9	0.1

Tabela 7.2 – (continuada)

Nome do registro	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
e0110	95.3	100.0	95.3	100.0	99.9	0.1
e0111	96.7	100.0	98.9	99.9	99.9	0.1
e0112	99.4	99.8	99.7	99.6	99.6	0.4
e0113	93.3	99.8	95.2	99.6	99.4	0.6
e0114	92.6	99.5	94.7	99.3	98.9	1.1
e0115	82.9	99.9	90.0	99.8	99.8	0.2
e0116	89.6	99.4	88.4	99.5	98.9	1.1
e0118	96.4	99.8	97.9	99.7	99.5	0.5
e0119	95.4	99.5	96.3	99.3	98.9	1.1
e0121	97.9	99.9	98.2	99.9	99.8	0.2
e0122	99.5	100.0	99.5	100.0	100.0	0.0
e0123	96.5	99.6	97.9	99.3	99.1	0.9
e0124	88.2	96.8	89.3	96.5	94.8	5.2
e0125	95.1	99.7	93.7	99.7	99.4	0.6
e0126	94.2	99.9	97.3	99.7	99.6	0.4
e0127	99.5	100.0	99.8	100.0	100.0	0.0
e0129	97.2	99.5	97.4	99.5	99.2	0.8
e0133	NaN	100.0	NaN	100.0	100.0	0.0
e0136	59.0	99.6	86.9	98.2	97.9	2.1
e0139	66.1	98.7	77.1	97.7	96.6	3.4
e0147	83.3	100.0	95.6	99.8	99.7	0.3
e0148	NaN	100.0	NaN	100.0	100.0	0.0
e0151	NaN	100.0	NaN	100.0	100.0	0.0
e0154	NaN	100.0	NaN	100.0	100.0	0.0
e0155	NaN	100.0	NaN	100.0	100.0	0.0
e0159	NaN	100.0	NaN	100.0	100.0	0.0
e0161	88.1	96.0	85.7	96.7	94.3	5.7
e0162	NaN	100.0	NaN	100.0	100.0	0.0
e0163	NaN	100.0	NaN	100.0	100.0	0.0
e0166	94.0	99.5	94.2	99.5	99.1	0.9
e0170	66.5	99.6	80.1	99.2	98.8	1.2
e0202	88.4	96.0	89.8	95.4	93.9	6.1
e0203	97.7	99.2	97.6	99.2	98.8	1.2
e0204	24.8	99.9	80.0	99.0	99.0	1.0
e0205	83.9	98.5	85.6	98.2	97.0	3.0
e0206	81.5	96.6	87.8	94.6	93.1	6.9
e0207	92.2	97.9	93.7	97.3	96.4	3.6
e0208	58.7	98.6	82.1	95.7	94.7	5.3

Tabela 7.2 – (continuada)

Nome do registro	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
e0210	98.5	99.1	96.1	99.6	99.0	1.0
e0211	69.6	96.9	83.5	93.5	92.0	8.0
e0212	95.0	99.7	94.8	99.7	99.5	0.5
e0213	64.1	91.1	73.9	86.6	83.5	16.5
e0302	89.3	99.7	94.8	99.3	99.0	1.0
e0303	NaN	100.0	NaN	100.0	100.0	0.0
e0304	NaN	100.0	NaN	100.0	100.0	0.0
e0305	82.5	92.9	89.1	88.3	88.6	11.4
e0306	90.4	98.5	92.2	98.2	97.2	2.8
e0403	89.8	99.9	89.1	99.9	99.7	0.3
e0404	96.6	99.5	94.5	99.7	99.3	0.7
e0405	84.7	98.4	86.6	98.1	96.9	3.1
e0406	94.2	98.2	95.2	97.8	97.1	2.9
e0408	90.3	98.1	83.2	99.0	97.4	2.6
e0409	NaN	100.0	NaN	100.0	100.0	0.0
e0410	97.3	100.0	98.6	99.9	99.9	0.1
e0411	80.1	98.9	85.8	98.4	97.6	2.4
e0413	88.7	99.9	93.2	99.9	99.8	0.2
e0415	70.9	89.6	77.8	85.7	83.3	16.7
e0417	81.9	98.8	92.7	96.8	96.2	3.8
e0418	47.3	98.6	70.2	96.5	95.4	4.6
e0501	NaN	100.0	NaN	100.0	100.0	0.0
e0509	NaN	100.0	NaN	100.0	100.0	0.0
e0515	98.6	99.8	96.6	99.9	99.8	0.2
e0601	NaN	100.0	NaN	100.0	100.0	0.0
e0602	40.9	97.7	67.3	93.4	91.7	8.3
e0603	92.8	99.7	94.8	99.5	99.2	0.8
e0604	93.4	99.4	95.7	99.0	98.6	1.4
e0605	97.7	99.4	97.2	99.5	99.1	0.9
e0606	99.4	99.9	99.6	99.9	99.8	0.2
e0607	98.1	93.0	96.1	96.7	96.3	3.7
e0609	61.4	98.8	75.9	97.6	96.6	3.4
e0610	93.7	99.7	98.1	99.0	98.8	1.2
e0611	NaN	100.0	NaN	100.0	100.0	0.0
e0612	97.9	99.9	97.9	99.9	99.9	0.1
e0613	85.0	97.1	90.2	95.3	94.2	5.8
e0614	87.9	82.6	87.6	83.0	85.7	14.3
e0615	51.9	100.0	93.2	99.5	99.4	0.6

Tabela 7.2 – (continuada)

Nome do registro	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
e0704	NaN	100.0	NaN	100.0	100.0	0.0
e0801	84.6	98.4	85.5	98.3	97.0	3.0
e0808	59.5	99.4	81.0	98.3	97.8	2.2
e0817	NaN	100.0	NaN	100.0	100.0	0.0
e0818	78.8	94.0	81.5	92.9	90.2	9.8
e1301	NaN	100.0	NaN	100.0	100.0	0.0
e1302	68.0	99.6	81.9	99.2	98.8	1.2
e1304	57.1	100.0	100.0	99.9	99.9	0.1
Média	84.8	98.8	90.9	98.3	97.9	2.1

Por causa do número elevado de registros na base (90), percebe-se que é impraticável mostrar aqui as tabelas de todas as instâncias de teste. Assim, as tabelas 7.3 e 7.4 fornecem as estatísticas médias obtidas para o primeiro e segundo canais dos registros de ECG, respectivamente. Elas foram colocadas apenas para o leitor ter uma noção do desempenho geral apresentado pelos métodos, mas a rigor não serve como fonte de comparação. O que importa é que, uma vez obtidas as estatísticas individuais, pode-se então gerar as listas de ECGs que serão utilizadas na seleção dos dados, de acordo com as configurações de seleção introduzidas no capítulo anterior.

Instância de teste	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha/ST	84.8	98.8	90.9	98.3	97.9	2.1
Mohebbi/ST	77.2	98.3	86.4	97.7	97.0	3.0
Gopalak/ST	90.9	99.1	93.7	98.8	98.5	1.5
Gopalak/T	85.1	98.9	90.2	98.5	98.2	1.8

Tabela 7.3 – Médias das estatísticas individuais para o canal 0.

Instância de teste	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha/ST	87.2	98.6	91.9	98.1	97.8	2.2
Mohebbi/ST	79.3	98.2	86.9	97.3	96.8	3.2
Gopalak/ST	92.0	98.9	93.5	98.7	98.4	1.6
Gopalak/T	86.9	99.0	91.3	98.8	98.5	1.5

Tabela 7.4 – Médias das estatísticas individuais para o canal 1.

7.2 Estatísticas coletivas

Aqui serão mostradas as tabelas de medidas obtidas para o caso coletivo. Todas as tabelas, exceto a 7.5, apresentam resultados médios levando em conta todas as derivações.

Configuração 1

A tabela 7.5 contém os resultados de cada derivação para o método de Rocha et al. na primeira configuração. Esta configuração de seleção é aquela proposta originalmente pelos autores. A derivação D3 tem resultado bom porque possui apenas um exemplar na base. As demais têm medidas de sensibilidade e preditividade positiva bem abaixo do esperado (a tabela 7.1 informava valores acima de 98%).

Nome da derivação	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
D3	95.4	98.3	81.7	99.6	98.1	1.9
MLI	76.5	98.5	86.1	97.3	96.2	3.8
MLIII	68.3	99.0	84.2	97.7	96.9	3.1
V1	78.6	96.9	81.9	96.1	94.1	5.9
V2	84.4	98.3	88.7	97.5	96.3	3.7
V3	90.3	99.8	95.4	99.5	99.3	0.7
V4	82.9	98.6	89.6	97.5	96.6	3.4
V5	68.1	97.4	82.6	94.5	93.0	7.0
Média	80.6	98.4	86.3	97.5	96.3	3.7

Tabela 7.5 – Estatísticas coletivas do método de Rocha et al. para a configuração 1.

A tabela 7.6 mostra o resultado médio para as 4 instâncias de teste nesta mesma configuração. Vê-se que os valores estão abaixo do esperado também para os outros dois métodos. Em especial, o de Gopalakrishnan et al. apresenta valores de sensibilidade bastante ruins. Acredita-se que isto aconteceu devido ao número extremamente limitado de batimentos usados no treinamento (236).

Instância de teste	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha/ST	80.6	98.4	86.3	97.5	96.3	3.7
Mohebbi/ST	82.7	98.7	88.1	97.9	97.0	3.0
Gopalak/ST	69.7	98.5	83.7	96.4	95.3	4.7
Gopalak/T	64.5	98.6	82.5	96.2	95.3	4.7

Tabela 7.6 – Médias das estatísticas coletivas da configuração 1.

Configuração 2

Nesta configuração foram selecionados somente ECGs que tiveram resultado bom no caso individual. Portanto, espera-se que as medidas estejam superiores àquelas da tabela anterior. De

fato isto ocorre para o primeiro e terceiro métodos, que apresentaram sensibilidade acima de 80%, conforme pode ser visto na tabela 7.7. O de Mohebbi e Moghadam teve valores ligeiramente menores. Acredita-se que isto tenha acontecido porque a seleção original do método já incluia ECGs "bons", além de considerar relativamente menos batimentos, o que facilitava o ajuste das redes. Aqui, o de Gopalakrishnan et al. se destaca com valores de 88% de SE e 91% de PP.

Instância de teste	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha/ST	85.4	98.0	89.6	97.2	96.1	3.9
Mohebbi/ST	82.7	97.3	87.2	96.4	95.2	4.8
Gopalak/ST	88.0	98.4	91.0	97.8	96.8	3.2
Gopalak/T	82.0	97.6	88.6	96.4	95.3	4.7

Tabela 7.7 – Médias das estatísticas coletivas da configuração 2.

Configuração 3

Na configuração 3 foi feita uma combinação entre as listas de ECG, de modo a utilizar os mesmos registros para todos os métodos. Somente foi feita separação entre as características ST e T. A tabela 7.8 apresenta os valores obtidos. Não há grandes diferenças em relação à anterior. Apenas nota-se que o método de Mohebbi e Moghadam teve resultado bem menor, com 74% de sensibilidade. O método de Gopalakrishnan et al. teve ligeiro aumento para cerca de 89%, enquanto o de Rocha et al. permaneceu praticamente inalterado.

Instância de teste	SE (%)	ES (%)	PP (%)	PN (%)	AC (%)	TF (%)
Rocha/ST	85.3	97.9	89.4	97.2	96.0	4.0
Mohebbi/ST	74.0	97.4	84.9	95.1	93.8	6.2
Gopalak/ST	89.2	98.3	91.7	98.0	97.0	3.0
Gopalak/T	83.9	97.5	88.3	96.7	95.4	4.6

Tabela 7.8 – Médias das estatísticas coletivas da configuração 3.

Configuração 4

Aqui foram usadas as mesmas listas do caso anterior, só que com número limitado de batimentos, em vez de considerar todos os batimentos de cada registro selecionado. De acordo com a tabela 7.9, os métodos de Mohebbi e Moghadam e de Gopalakrishnan et al. para a característica T tiveram resultado bastante deteriorado, enquanto os métodos de Rocha et al. e de Gopalakrishnan et al. para a característica ST pioraram em pouco menos de 5% no quesito sensibilidade. A tabela 7.10 mostra como ficaram as quantidades de batimentos selecionados e a proporção de ocorrência da característica observada.

Instância de teste	**SE (%)**	**ES (%)**	**PP (%)**	**PN (%)**	**AC (%)**	**TF (%)**
Rocha/ST	81.0	98.5	88.2	97.5	96.5	3.5
Mohebbi/ST	68.5	98.1	82.8	96.2	94.9	5.1
Gopalak/ST	86.3	98.8	90.5	98.2	97.3	2.7
Gopalak/T	73.0	98.5	86.1	97.4	96.4	3.6

Tabela 7.9 – Médias das estatísticas coletivas da configuração 4.

Nome da derivação	Total	Normais	Isquêmicos	Percentual de isquêmicos
D3	5000	4631	369	7.4
MLI	20000	17886	2114	10.6
MLIII	100000	93073	6927	6.9
V1	30000	25391	4609	15.4
V2	20000	17216	2784	13.9
V3	10000	9493	507	5.1
V4	100000	87235	12765	12.8
V5	100000	84836	15164	15.2

Tabela 7.10 – Parâmetros de seleção dos dados na configuração 4.

Configuração 5

Esta é a configuração em que todos os dados extraídos da base são utilizados no treinamento. Espera-se obter os piores resultados até agora. Por outro lado, eles devem mostrar de maneira mais realista o desempenho de um método face a um conjunto diverso de características extraídas. Isto significa, por exemplo, que se o método for testado em outra base de ECGs, ou dentro de um contexto clínico real, as chances de obter resultados bons serão maiores para aqueles métodos que tiverem sucesso nesta instância de teste.

Na tabela 7.11, vê-se claramente que o método de Gopalakrishnan et al. se destaca dos demais, mantendo de maneira resiliente o valor de sensibilidade acima dos 83%, e o de preditividade positiva em 89%. Isto vale para a característica ST, mas não para T. No fim das contas não há problema nisso, pois o que vale é o máximo entre os dois (lembre que neste método a condição de isquemia é dada pelo OU lógico entre o resultado das duas redes).

Instância de teste	**SE (%)**	**ES (%)**	**PP (%)**	**PN (%)**	**AC (%)**	**TF (%)**
Rocha/ST	77.0	98.4	86.6	97.0	96.0	4.0
Mohebbi/ST	65.8	98.3	82.5	95.8	94.7	5.3
Gopalak/ST	83.9	98.6	89.0	98.0	97.0	3.0
Gopalak/T	71.1	98.5	84.7	97.1	96.1	3.9

Tabela 7.11 – Médias das estatísticas coletivas da configuração 5.

7.3 Informações adicionais

A tabela 7.12 apresenta alguns dados sobre o treinamento das redes neurais nas várias configurações e também no caso individual. O motivo de ter-se realizado somente uma tentativa para o caso coletivo é que o tempo de treinamento é grande, e não se conseguiria concluir o trabalho em tempo hábil caso se fizesse mais tentativas. A título de curiosidade, o tempo para processar todos os ECGs e extrair as características foi cerca de 4 minutos, enquanto o tempo para construir os conjuntos de dados após a extração foi cerca de 24 minutos.

Procedimento	Nº de redes	Tentativas	Tempo (s)	Tempo (hh:mm:ss)
Redes individuais	720	3	7057	01:57:37
Redes coletivas (config. 1)	32	1	384	00:06:24
Redes coletivas (config. 2)	32	1	2455	00:40:55
Redes coletivas (config. 3)	32	1	2519	00:41:59
Redes coletivas (config. 4)	32	1	1127	00:18:47
Redes coletivas (config. 5)	32	1	4830	01:20:30
Total geral	880	–	18372	05:06:12

Tabela 7.12 – Tempo total de execução para treinamento das redes neurais.

A tabela 7.13 mostra algumas informações sobre a quantidade de código escrito ao longo do desenvolvimento. Os arquivos em C++, em conjunto com um modelo criado no Simulink, fornecem um ferramental valioso para simulação dos métodos de detecção em tempo real. Infelizmente, faltou tempo e espaço para incluí-los nesta monografia. Contudo, o leitor está convidado a fazer o *download* do repositório do GitHub (`https://github.com/dsogari/tg-ecp-ufrgs`) e passar os olhos no modelo Simulink, dentro da pasta "realtime". A versão de MATLAB utilizada foi a R2014a (a mais recente até o momento de entrega da monografia).

Tipo de arquivo	Extensão	Nº de arquivos	Nº de linhas
Código em C	.c	10	6598
Código em C++	.cpp	19	2188
Cabeçalho C/C++	.h	28	17678
Código em MATLAB	.m	104	2675
Total	–	161	29139

Tabela 7.13 – Número de arquivos e de linhas de código por tipo de arquivo.

8 CONCLUSÃO

8.1 Conclusão geral sobre os métodos

Com base nas informações do capítulo 7, pode-se chegar à conclusão de que o método de Gopalakrishnan et al. fornece os melhores resultados em termos de métricas de confiabilidade. Viu-se que as medidas de sensibilidade e preditividade positiva deste método estão sempre acima das dos outros métodos. A única exceção é na configuração 1, em que o método de Rocha et al. leva vantagem. Aliás, o método de Rocha et al. seria o segundo melhor, enquanto o de Mohebbi e Moghadam ficaria em última posição.

Não se pode deixar de lado o fato de que, nos testes realizados, todos os resultados ficaram aquém daqueles obtidos originalmente pelos autores dos métodos. Acredita-se que a impossibilidade de seguir a proposta original dos autores, devida à falta de informação nos artigos e devida também à dificuldade de se reproduzir um experimento científico como este de igual para igual, tenha prejudicado o desempenho final. Não obstante, com base nos resultados obtidos, quer-se defender a conclusão dada acima como honesta e verdadeira.

Assim, para fins práticos, o método de Gopalakrishnan et. al parece apresentar os melhores resultados de confiabilidade, tornando-o, dentre os três métodos avaliados, o mais confiável. Sugere-se portanto o seu uso em um sistema de monitoramento cardíaco com suporte à tomada de decisão médica, em detrimento do uso dos métodos de Rocha et al. e de Mohebbi e Moghadam. Adicionalmente, há que se mencionar o fato de que o método de Gopalakrishnan et. al é também o método mais simples de implementar, conforme foi discutido ao longo do trabalho.

8.2 Contribuição da pesquisa

É necessário ressaltar a importância da utilização da técnica de expansão em funções de Hermite, por ela estar presente em dois dos métodos com maior confiabilidade. Também merece destaque a implementação dos procedimentos de detecção de batimentos cardíacos, de detecção de pontos de interesse e de construção de *template*.

Em especial, a construção de *template* é importante não apenas porque permite fácil eliminação de artefatos, mas porque é a técnica empregada originalmente pelos criadores do banco de ECGs *European ST-T*. Os especialistas responsáveis pela análise dos batimentos cardíacos tiveram acesso a um modelo de batimento construído com os primeiros 30 segundos de ECG, e fizeram as anotações de desvio de segmento ST e de onda T com base nesse modelo. Daí a importância do uso de *template* na detecção de isquemia usando esta base de dados.

8.3 Trabalhos Futuros

Futuramente, pensa-se em implementar o método selecionado num dispositivo móvel do tipo *smartphone*, com o intuito de testar efetivamente o método e detectar a doença num paciente.

REFERÊNCIAS

ABRAMOWITZ, M. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables,*. [S.l.]: Dover Publications, Incorporated, 1974. ISBN 0486612724. Citado na página 31.

AFSAR, F.; ARIF, M. Detection of st segment deviation episodes in the ecg using klt with an ensemble neural classifier. In: *Emerging Technologies, 2007. ICET 2007. International Conference on*. [S.l.: s.n.], 2007. p. 11–16. Citado 2 vezes nas páginas 15 e 48.

AKSELROD, S. et al. Computerised analysis of st segment changes in ambulatory electrocardiograms. *Medical and Biological Engineering and Computing*, Kluwer Academic Publishers, v. 25, n. 5, p. 513–519, 1987. ISSN 0140-0118. Citado na página 48.

ANDREAO, R. et al. St-segment analysis using hidden markov model beat segmentation: application to ischemia detection. In: *Computers in Cardiology, 2004*. [S.l.: s.n.], 2004. p. 381–384. Citado 2 vezes nas páginas 15 e 48.

ANTIPUESTO, D. J. *Myocardial Infarction*. 2014. Disponível em: <http://nursingcrib-.com/nursing-care-plan/nursing-care-plan-myocardial-infarction/>. Citado na página 24.

ATKIELSKI, A. *Schematic diagram of normal sinus rhythm for a human heart as seen on ECG*. 2007. Disponível em: <http://en.wikipedia.org/wiki/Qrs_complex>. Citado na página 23.

AUGUSTYNIAK, P.; TADEUSIEWICZ, R. *Ubiquitous cardiology: emerging wireless telemedical applications*. [S.l.]: Information Science Reference, 2009. Citado na página 17.

BADILINI, F. et al. Beat-to-beat quantification and analysis of st displacement from holter ecgs: a new approach to ischemia detection. In: *Computers in Cardiology 1992, Proceedings of.* [S.l.: s.n.], 1992. p. 179–182. Citado 2 vezes nas páginas 15 e 48.

BADILINI, F.; MOSS, A. J.; TITLEBAUM, E. L. Cubic spline baseline estimation in ambulatory ecg recordings for the measurement of st segment displacements. In: *Engineering in Medicine and Biology Society, 1991. Vol.13: 1991., Proceedings of the Annual International Conference of the IEEE*. [S.l.: s.n.], 1991. p. 584–585. Citado 3 vezes nas páginas 47, 53 e 60.

BLANCHARD, J. *FIR filter structure*. 2008. Disponível em: <http://en.wikipedia.org/wiki-/Finite_impulse_response>. Citado na página 35.

BRASIL. *Portaria n.Ã‚Âº 1101/GM*. [S.l.], 2002. Disponível em: <http://dtr2001.saude.gov.br-/sas/PORTARIAS/Port2002/Gm/GM-1101.htm>. Citado na página 16.

CASTELLS, F. et al. Principal component analysis in ecg signal processing. *EURASIP Journal on Advances in Signal Processing*, v. 2007, n. 1, p. 074580, 2007. ISSN 1687-6180. Citado na página 48.

CHEN, S.-W.; CHEN, H.-C.; CHAN, H.-L. A real-time qrs detection method based on moving-averaging incorporating with wavelet denoising. *Computer Methods and Programs in Biomedicine*, v. 82, n. 3, p. 187 – 195, 2006. ISSN 0169-2607. Citado na página 47.

CHU, C.-H.; DELP, E. Impulsive noise suppression and background normalization of electrocardiogram signals using morphological operators. *Biomedical Engineering, IEEE Transactions on*, v. 36, n. 2, p. 262–273, Feb 1989. ISSN 0018-9294. Citado na página 47.

CLIFFORD, G. D.; AZUAJE, F.; MCSHARRY, P. *Advanced Methods And Tools for ECG Data Analysis*. Norwood, MA, USA: Artech House, Inc., 2006. ISBN 1580539661. Citado na página 61.

COUCEIRO, R. et al. On the detection of premature ventricular contractions. In: *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. [S.l.: s.n.], 2008. p. 1087–1091. ISSN 1557-170X. Citado 3 vezes nas páginas 48, 49 e 57.

CRILEY, D. *Normal Heart Conduction System*. Disponível em: <www.blaufuss.org>. Citado na página 20.

DAMATO, A. *Plot of first 5 Hermite polynomials*. 2009. Disponível em: <http://en.wikipedia-.org/wiki/Hermite_polynomials>. Citado na página 30.

DASKALOV, I.; DOTSINSKY, I.; CHRISTOV, I. Developments in ecg acquisition, preprocessing, parameter measurement, and recording. *Engineering in Medicine and Biology Magazine, IEEE*, v. 17, n. 2, p. 50–58, March 1998. ISSN 0739-5175. Citado 2 vezes nas páginas 47 e 60.

DEMS, M. *Plot of first 5 Hermite functions*. 2009. Disponível em: <http://en.wikipedia.org-/wiki/Hermite_polynomials>. Citado na página 31.

DUBIN, D. *Rapid interpretation of EKG's: an interactive course*. [S.l.]: Cover Pub. Co., 2000. ISBN 9780912912066. Citado 2 vezes nas páginas 16 e 19.

ELGENDI, M. Fast qrs detection with an optimized knowledge-based method: Evaluation on 11 standard ecg databases. *PLoS ONE*, Public Library of Science, v. 8, n. 9, p. e73557, 09 2013. Citado 2 vezes nas páginas 47 e 48.

EXARCHOS, T. et al. An association rule mining-based methodology for automated detection of ischemic ecg beats. *Biomedical Engineering, IEEE Transactions on*, v. 53, n. 8, p. 1531–1540, Aug 2006. ISSN 0018-9294. Citado na página 48.

EXARCHOS, T. P. et al. A methodology for the automated creation of fuzzy expert systems for ischaemic and arrhythmic beat classification based on a set of rules obtained by a decision tree. *Artificial Intelligence in Medicine*, v. 40, n. 3, p. 187 – 200, 2007. ISSN 0933-3657. Citado na página 48.

FISCH, C. Centennial of the string galvanometer and the electrocardiogram. *Journal of the American College of Cardiology*, v. 36, n. 6, p. 1737 – 1745, 2000. ISSN 0735-1097. Citado na página 16.

GARCIA, J. et al. Automatic detection of st-t complex changes on the ecg using filtered rms difference series: application to ambulatory ischemia monitoring. *Biomedical Engineering, IEEE Transactions on*, v. 47, n. 9, p. 1195–1201, Sept 2000. ISSN 0018-9294. Citado 2 vezes nas páginas 15 e 48.

GOLDBERGER, A. L. et al. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, v. 101, n. 23, p. e215–e220, 2000. Citado na página 63.

GOLETSIS, Y. et al. Automated ischemic beat classification using genetic algorithms and multicriteria decision analysis. *Biomedical Engineering, IEEE Transactions on*, v. 51, n. 10, p. 1717–1725, Oct 2004. ISSN 0018-9294. Citado 2 vezes nas páginas 15 e 48.

GOPALAKRISHNAN, R.; ACHARYA, S.; MUGLER, D. Real time monitoring of ischemic changes in electrocardiograms using discrete hermite functions. In: *Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE*. [S.l.: s.n.], 2004. v. 1, p. 438–441. Citado 3 vezes nas páginas 15, 48 e 55.

GROUP, Y. M. *Illustration of the Coronary Arteries*. Disponível em: <www.yalemedicalgroup-.org>. Citado na página 24.

HAMILTON, P. S.; TOMPKINS, W. J. Quantitative investigation of qrs detection rules using the mit/bih arrhythmia database. *Biomedical Engineering, IEEE Transactions on*, BME-33, n. 12, p. 1157–1165, Dec 1986. ISSN 0018-9294. Citado na página 59.

HARRIS, F. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, v. 66, n. 1, p. 51–83, Jan 1978. ISSN 0018-9219. Citado na página 40.

KAISER, J.; SCHAFER, R. On the use of the i0-sinh window for spectrum analysis. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, v. 28, n. 1, p. 105–107, Feb 1980. ISSN 0096-3518. Citado na página 40.

KLABUNDE, R. E. *Precordial leads*. 2008. Disponível em: <http://www.cvphysiology.com-/Arrhythmias/A013c.htm>. Citado na página 22.

LYNN, P. Recursive digital filters for biological signals. *Medical and biological engineering*, Kluwer Academic Publishers, v. 9, n. 1, p. 37–43, 1971. ISSN 0025-696X. Citado na página 35.

LYNN, P. Online digital filters for biological signals: some fast designs for a small computer. *Medical and Biological Engineering and Computing*, Kluwer Academic Publishers, v. 15, n. 5, p. 534–540, 1977. ISSN 0140-0118. Citado 2 vezes nas páginas 35 e 36.

MAGLAVERAS, N. et al. An adaptive backpropagation neural network for real-time ischemia episodes detection: development and performance analysis using the european st-t database. *Biomedical Engineering, IEEE Transactions on*, v. 45, n. 7, p. 805–813, July 1998. ISSN 0018-9294. Citado na página 48.

MILOSAVLJEVIC, N.; PETROVIC, A. St segment change detection by means of wavelets. In: *Neural Network Applications in Electrical Engineering, 2006. NEUREL 2006. 8th Seminar on*. [S.l.: s.n.], 2006. p. 137–140. Citado 2 vezes nas páginas 15 e 48.

MOHEBBI, M.; MOGHADAM, H. A. Real-time ischemic beat classification using backpropagation neural network. In: *Signal Processing and Communications Applications, 2007. SIU 2007. IEEE 15th*. [S.l.: s.n.], 2007. p. 1–4. Citado 3 vezes nas páginas 15, 48 e 53.

MUGLER, D.; CLARY, S.; WU, Y. Discrete hermite expansion of digital signals: applications to ecg signals. In: *Digital Signal Processing Workshop, 2002 and the 2nd Signal Processing Education Workshop. Proceedings of 2002 IEEE 10th*. [S.l.: s.n.], 2002. p. 262–267. Citado na página 31.

OKADA, M. A digital filter for the ors complex detection. *Biomedical Engineering, IEEE Transactions on*, BME-26, n. 12, p. 700–703, Dec 1979. ISSN 0018-9294. Citado na página 47.

OLIVEIRA, L. A. P. de; CAVARARO, R. *Estatísticas da Saúde: Assistência Médico-Sanitária*. [S.l.], 2009. Disponível em: <http://www.ibge.gov.br/home/estatistica/populacao-/condicaodevida/ams/2009/ams2009.pdf>. Citado na página 16.

OPPENHEIM, A. V.; SCHAFER, R. W. *Discrete-Time Signal Processing*. Third. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009. Citado na página 32.

PAN, J.; TOMPKINS, W. J. A real-time qrs detection algorithm. *Biomedical Engineering, IEEE Transactions on*, BME-32, n. 3, p. 230–236, March 1985. ISSN 0018-9294. Citado 2 vezes nas páginas 47 e 59.

PANG, L. et al. Real time heart ischemia detection in the smart home care system. In: *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*. [S.l.: s.n.], 2005. p. 3703–3706. Citado 4 vezes nas páginas 15, 48, 49 e 68.

PAPALOUKAS, C. et al. A knowledge-based technique for automated detection of ischaemic episodes in long duration electrocardiograms. *Medical and Biological Engineering and Computing*, Springer-Verlag, v. 39, n. 1, p. 105–112, 2000. ISSN 0140-0118. Citado na página 48.

PAPALOUKAS, C. et al. Use of a novel rule-based expert system in the detection of changes in the st segment and the t wave in long duration ecgs. *Journal of electrocardiology*, v. 35, n. 1, p. 27—34, January 2002. ISSN 0022-0736. Citado na página 48.

PAPALOUKAS, C. et al. An ischemia detection method based on artificial neural networks. *Artif. Intell. Med.*, Elsevier Science Publishers Ltd., Essex, UK, v. 24, n. 2, p. 167–178, Jun 2001. ISSN 0933-3657. Citado 2 vezes nas páginas 15 e 48.

PARK, K.; LEE, K.; YOON, H. Application of a wavelet adaptive filter to minimise distortion of the st-segment. *Medical and Biological Engineering and Computing*, Kluwer Academic Publishers, v. 36, n. 5, p. 581–586, 1998. ISSN 0140-0118. Citado na página 47.

RANJITH, P.; BABY, P.; JOSEPH, P. Ecg analysis using wavelet transform: application to myocardial ischemia detection. *ITBM-RBM*, v. 24, n. 1, p. 44 – 47, 2003. ISSN 1297-9562. Citado na página 48.

ROCHA, T. et al. A lead dependent ischemic episodes detection strategy using hermite functions. *Biomedical Signal Processing and Control*, v. 5, n. 4, p. 271 – 281, 2010. ISSN 1746-8094. Citado 4 vezes nas páginas 15, 48, 49 e 52.

SENHADJI, L. et al. Comparing wavelet transforms for recognizing cardiac patterns. *Engineering in Medicine and Biology Magazine, IEEE*, v. 14, n. 2, p. 167–173, Mar 1995. ISSN 0739-5175. Citado na página 48.

STAMKOPOULOS, T. et al. Ischemic classification techniques using an advanced neural network algorithm. In: *Computers in Cardiology 1997*. [S.l.: s.n.], 1997. p. 351–354. ISSN 0276-6547. Citado 2 vezes nas páginas 15 e 48.

SUN, Y. *Arrhythmia Recognition from Electrocardiogram Using Non-linear Analysis and Unsupervised Clustering Techniques*. Tese (Doutorado) — Nanyang Technological University, School of Electrical and Electronic Engineering, 2002. Disponível em: <http://hdl.handle.net/10356/3312>. Citado na página 64.

SUN, Y.; CHAN, K.; KRISHNAN, S. Characteristic wave detection in ecg signal using morphological transform. *BMC Cardiovascular Disorders*, BioMed Central, v. 5, n. 1, Sep 2005. Citado 3 vezes nas páginas 47, 49 e 59.

SUPPAPPOLA, S.; SUN, Y. Nonlinear transforms of ecg signals for digital qrs detection: a quantitative analysis. *Biomedical Engineering, IEEE Transactions on*, v. 41, n. 4, p. 397–400, April 1994. ISSN 0018-9294. Citado na página 47.

TADDEI, A. et al. The european st-t database: standard for evaluating systems for the analysis of st-t changes in ambulatory electrocardiography. *European Heart Journal*, v. 13, n. 9, p. 1164–1172, 1992. Disponível em: <http://eurheartj.oxfordjournals.org/content/13/9/1164.long>. Citado na página 63.

TOMPKINS, W. J. *Biomedical Digital Signal Processing: C-language Examples and Laboratory Experiments for the IBM PC*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993. ISBN 0-13-067216-5. Citado 3 vezes nas páginas 37, 52 e 59.

TRAHANIAS, P. An approach to qrs complex detection using mathematical morphology. *Biomedical Engineering, IEEE Transactions on*, v. 40, n. 2, p. 201–205, Feb 1993. ISSN 0018-9294. Citado na página 47.

VETTERLI, M.; KOVACEVIC, J. *Wavelets and Subband Coding*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995. ISBN 0-13-097080-8. Citado 2 vezes nas páginas 27 e 29.

VILA, J. et al. Sutil: Intelligent ischemia monitoring system. *International Journal of Medical Informatics*, v. 47, n. 3, p. 193 – 214, 1997. ISSN 1386-5056. Citado na página 48.

WOLF, A. S. *Análise Automática de Sinais Eletrocardiográficos por Redes Neurais Artificiais*. Dissertação (Mestrado) — Departamento de Engenharia Elétrica da PUC-Rio, Rio de Janeiro, RJ, Feb 2004. Disponível em: <http://www2.dbd.puc-rio.br/pergamum/tesesabertas-/0210429_04_pretextual.pdf>. Citado na página 61.

Apêndices

APÊNDICE A – TRECHOS DE CÓDIGO MATLAB

A.1 Pré-processamento

Código-fonte A.1 – Função qrs_filter

```
function [sigI,delay] = qrs_filter(x, Fs)
% filtro passa-baixas com frequencia de corte ~11 Hz
[bl,al,gl,dl] = intfdesign.lowpass('N,F3db',2,11,Fs);
% filtro passa-altas com frequencia de corte ~5 Hz
[bh,ah,gh,dh] = intfdesign.highpass('N,F3db',1,5,Fs);
% filtro 'diferenciador' de 5 pontos
[bd,ad,gd,dd] = intfdesign.derivative('N,M',1,3);
% filtro de media-movel com largura de ~150 ms
[bi,ai,gi,di] = intfdesign.maverage('Width',0.15,Fs);
% aplica os filtros
sigL = filter(bl, al, x)    ./gl;
sigF = filter(bh, ah, sigL) ./gh;
sigD = filter(bd, ad, sigF) ./gd;
sigS = min(sigD.^2./2^3, 2^15-1);
sigI = filter(bi, ai, sigS) ./gi;
% calcula o atraso total
delay = dl + dh + dd + di;
```

Código-fonte A.2 – Função fp_filter

```
function [sigD,sigM] = fp_filter(x, Fs, delay)
% filtro passa-baixas com frequencia de corte ~11 Hz
[bl,al,~,dl] = intfdesign.lowpass('N,F3db',2,11,Fs);
% filtro de media-movel com largura de ~50 ms
[bi,ai,~,di] = intfdesign.maverage('Width',0.05,Fs);
% filtro 'diferenciador' de 2 pontos
[bd,ad,~,dd] = intfdesign.derivative('N,M',1,0);
% filtro derivativo morfologico de escala s
s = round(0.06*Fs); B = ones(2*s+1,1);
% aplica os filtros
sigL = filter(bl, al, x);
sigI = filter(bi, ai, sigL);
sigD = filter(bd, ad, sigI);
sigM = (imerode(sigL,B)+imdilate(sigL,B)-2*sigL);
% atraso artificial dos sinais
d = ceil(delay - (dl + di + dd));
sigD = [zeros(d,1); sigD(1:end-d)];
d = ceil(delay - dl);
sigM = [zeros(d,1); sigM(1:end-d)];
```

Código-fonte A.3 – Função noise_filter

```
function sigN = noise_filter(x, Fs, Fm, delay)
% filtro rejeita-faixa com frequencia central em Fm
[bs,as] = cheby1(2,0.1,[Fm-1 Fm+1]*2/Fs,'stop');
ds = mean(grpdelay(bs,as,[5 15]*2/Fs));
% filtro passa-baixas com frequencia de corte ~40Hz
[bl,al] = butter(4,40*2/Fs);
dl = mean(grpdelay(bl,al,[5 15]*2/Fs));
% aplica os filtros
sigS = filter(bs, as, x);
sigL = filter(bl, al, sigS);
% atraso artificial so sinal
d = ceil(delay - ds - dl);
sigL = [zeros(d,1); sigL(1:end-d)];
```

Código-fonte A.4 – Função detect_qrs

```
function [R,RR] = detect_qrs(sigI, Fs)

%% initializations
N = length(sigI);               % length of the signal
RR = zeros(N,1);                % buffer for the RR mean history
R = zeros(N,1);                 % buffer for the QRS locations
training_period = 2*Fs;         % length of training period
threshold = 0;                  % signal threshold
sig_level = 0;                  % signal level
noise_level = 0;                % noise level
est_ratio = 0.25;               % ratio of signal/noise level estimation
qrs_count = 0;                  % count of QRS complex in main QRS buffer
qrs_count2 = 0;                 % current position in second QRS buffer
searchback_idx = 0;             % index of searchback starting point
last_peak_idx = 0;              % index of the last peak detected
last_peak_amp = 0;              % amplitude of the last peak detected
last_qrs_idx = 0;               % index of last detected QRS complex
rr_mean = 0;                    % running average of RR intervals
rr_mean2 = 0;                   % secondary running average of RR intervals
rr_miss = 0;                    % interval for qrs to be assumed as missed
signal_rising = false;          % flag to indicate a rise in the signal

%% algorithm
for i = 1:N

    % get current sample
    a = sigI(i);

    % initialize iteration flags
    peak_detected = false;
    qrs_detected = false;

    %% peak detection
    if a > last_peak_amp

        % signalize positive slope
        signal_rising = true;

        % update peak info
        last_peak_amp = a;
```

```
        last_peak_idx = i;

    elseif ~signal_rising

        % update current amplitude
        last_peak_amp = a;

    elseif a < 0.5*last_peak_amp

        % signalize peak detection
        peak_amp = last_peak_amp;
        peak_idx = last_peak_idx;
        peak_detected = true;

        % reset state
        signal_rising = false;
        last_peak_amp = a;

    end

    %% qrs detection and level estimation
    if peak_detected
        if i <= training_period
            if peak_amp >= threshold
                sig_level = max(sig_level,peak_amp);
                last_qrs_idx = peak_idx;
            else
                noise_level = max(noise_level,peak_amp);
            end
        elseif peak_amp >= threshold
            sig_level = estimate(sig_level,est_ratio,peak_amp);
            qrs_detected = ~istwave(sigI, peak_idx, last_qrs_idx, rr_mean2, Fs);
        else
            noise_level = estimate(noise_level,est_ratio,peak_amp);
        end
    end

    %% search back
    if qrs_detected || signal_rising || searchback_idx == 0

        % do nothing if:
        %   1. a qrs complex was detected in the current iteration
        %   2. a new qrs might be detected in the following iterations
        %   3. the search back procedure is not yet enabled

    elseif i-searchback_idx >= rr_miss

        % search back and locate the max in this interval
        interval = floor(rr_mean2);
        peak_idx = findmax(sigI, i-interval+1, interval);
        peak_amp = sigI(peak_idx);

        % check if candidate peak is from qrs
        if (peak_amp < threshold) && (peak_amp >= 0.5*threshold) && ...
                ~istwave(sigI, peak_idx, last_qrs_idx, rr_mean2, Fs)
            % signalize qrs detection
            qrs_detected = true;
            % adjust signal levels
            sig_level = estimate(sig_level,0.25,peak_amp);
```

```
        else
            % reduce levels by half
            sig_level = 0.5*sig_level;
            noise_level = 0.5*noise_level;
            % postpone searchback
            searchback_idx = searchback_idx + interval;
        end
    end


    %% update QRS info and RR-interval
    if qrs_detected

        % push new QRS location to output buffer
        qrs_count = qrs_count + 1;
        R(qrs_count) = peak_idx;

        % update RR intervals and limits
        new_rr = peak_idx - last_qrs_idx;
        [rr_mean,rr_mean2] = update_rr(new_rr);
        rr_miss = round(1.66*rr_mean2);
        RR(qrs_count) = rr_mean;

        % update indices
        last_qrs_idx = peak_idx;
        searchback_idx = peak_idx;
    end

    %% updates
    threshold = noise_level + 0.25*abs(sig_level - noise_level);

    % decrease estimation ratio for times beyond the training period
    if i == training_period
        est_ratio = est_ratio/2;
    end
end

% trim the output vectors
R(qrs_count+1:end) = [];
RR(qrs_count+1:end) = [];


function Result = estimate(x,r,v)
Result = (1-r)*x + r*v;

function Result = findmax(signal, begin, interval)
[~,x] = max(signal(begin:begin+interval-1));
Result = x + begin - 1;

function Result = maxdiff(signal, begin, interval)
Result = max(diff(signal(begin:begin+interval-1)));

function Result = istwave(sigI, candQrs, lastQrs, rrMean, Fs)
rr = candQrs - lastQrs;
if rr <= floor(0.2*Fs)
    Result = true;
elseif rr > floor(0.5*rrMean)
    Result = false;
else
```

```
    % half the length of a QRS
    len = round(0.1*Fs);
    % find max slopes
    slope1 = maxdiff(sigI,candQrs-len+1,len);
    slope2 = maxdiff(sigI,lastQrs-len+1,len);
    % check condition for T wave
    Result = slope1 < 0.5*slope2;
end

function [rr_mean,rr_mean2] = update_rr(new_rr)
persistent init rr_last rr_last2 rr_idx rr_idx2 rr1 rr2 rr_low rr_high
if isempty(init)
    rr_idx = 1;
    rr_idx2 = 1;
    rr_last(1:8) = new_rr;
    rr_last2(1:8) = new_rr;
    rr_low = 0.92*new_rr;
    rr_high = 1.16*new_rr;
    rr1 = new_rr;
    rr2 = new_rr;
    init = true;
else
    rr_idx = mod(rr_idx,8) + 1;
    rr1 = rr1 + 0.125*(new_rr - rr_last(rr_idx));
    rr_last(rr_idx) = new_rr;
    if rr_low <= new_rr && new_rr <= rr_high
        rr_idx2 = mod(rr_idx2,8) + 1;
        rr2 = rr2 + 0.125*(new_rr - rr_last2(rr_idx2));
        rr_last2(rr_idx2) = new_rr;
        rr_low = 0.92*rr2;
        rr_high = 1.16*rr2;
    end
end
rr_mean = rr1;
rr_mean2 = rr2;
```

Código-fonte A.5 – Função detect_fp

```
function FP = detect_fp(sigD,sigI,Rpeaks,Fs)
% Funcao para detectar os pontos caracteristicos das ondas de ECG, com
% base na localizacao dos picos de onda R. Os pontos detectados sao:
%   Pon - inicio da onda P
%   Ppk - pico da onda P
%   Ron - inicio da onda R
%   Rpk - pico da onda R
%   Rof - fim da onda R
%   Tpk - pico da onda T
%   Tof - fim da onda T

% initializations
N = length(sigI);
M = length(Rpeaks);
RR = diff(Rpeaks);
RR1 = [RR(1); RR];
RR2 = [RR; RR(end)];

% limits
```

```
L1 = round(0.10*Fs);
L2 = round(0.02*Fs);
L3 = round(0.15*Fs);

% outputs
FP = zeros(7,M);

% algorithm
for i = 1:M
    Rpk = Rpeaks(i);

    % R-wave
    Rpk = search_peak_abs(sigD,sigI,max(1,Rpk-L1),1,min(N,Rpk+L1),Rpk);
    if sigI(Rpk) > 0
        % inverted
        Ron = search_first_mark(-sigI,Rpk-L2,-1,max(1,Rpk-L1),Rpk);
        Roff = search_first_mark(-sigI,Rpk+L2,1,min(N,Rpk+L1),Rpk);
    else
        % normal
        Ron = search_first_mark(sigI,Rpk-L2,-1,max(1,Rpk-L1),Rpk);
        Roff = search_first_mark(sigI,Rpk+L2,1,min(N,Rpk+L1),Rpk);
    end

    % P-wave
    len = floor(0.375*RR1(i));
    Ppk = search_peak_abs(sigD,sigI,Ron-1,-1,max(1,Ron-len),Ron);
    if sigI(Ppk) > 0
        % inverted
        Pon = search_best_mark(-sigI,Ppk-1,-1,max(1,Ppk-L3),Ppk);
    else
        % normal
        Pon = search_best_mark(sigI,Ppk-1,-1,max(1,Ppk-L3),Ppk);
    end

    % T-wave
    len = floor(0.5*RR2(i));
    Tpk = search_peak_abs(sigD,sigI,Roff+1,1,min(N,Roff+len),Roff);
    if sigI(Tpk) > 0
        % inverted
        Toff = search_best_mark(-sigI,Tpk+1,1,min(N,Tpk+L3),Tpk);
    else
        % normal
        Toff = search_best_mark(sigI,Tpk+1,1,min(N,Tpk+L3),Tpk);
    end

    % save FP
    FP(:,i) = [Pon Ppk Ron Rpk Roff Tpk Toff]';
end


function pos = search_peak_abs(dataD,dataI,istart,inc,iend,default)
idx = [default default];
val = [0 0];
for i = istart+inc:inc:iend-inc
    y = dataD(i);
    if (dataD(i-1) < y && y >= dataD(i+1))
        if y > val(1)
            idx(1) = i;
            val(1) = y;
```

```
        end
    elseif (dataD(i-1) > y && y <= dataD(i+1))
        if y < val(2)
            idx(2) = i;
            val(2) = y;
        end
    end
end
if idx(1) < idx(2)
    left = idx(1);
    right = idx(2);
else
    left = idx(2);
    right = idx(1);
end
if right - left <= 1
    pos = left;
else
    [y,x] = findpeaks(abs(dataI(left:right)));
    if ~isempty(x)
        [~,i] = max(y);
        x = x(i);
        pos = left + x - 1;
    else
        pos = left;
    end
end

function pos = search_first_mark(data,istart,inc,iend,default)
if abs(iend - istart - inc) > 0
    win = data(istart:inc:iend);
    [~,x] = findpeaks(win,'NPeaks',1);
    if isempty(x)
        [~,x] = max(win);
    end
    pos = istart + inc*(x-1);
else
    pos = default;
end

function pos = search_best_mark(data,istart,inc,iend,default)
if abs(iend - istart - inc) > 0
    win = data(istart:inc:iend);
    [y,x] = findpeaks(win);
    if ~isempty(y)
        [~,i] = max(y);
        x = x(i);
    else
        [~,x] = max(win);
    end
    pos = istart + inc*(x-1);
else
    pos = default;
end
```

Código-fonte A.6 – Função extract_beats

```
function [Beats,FP] = extract_beats(data,FP,Fs)
N = length(data);
half = floor(Fs*0.6);
FrameSize = 2*half+1;
center = half + 1;
M = size(FP.R,1);
Beats = zeros(FrameSize,M);
for i = 1:M
    Beat = data(max(1,FP(i,1)-5):min(N,FP(i,7)+5));
    Beat = Beat - polyfit_baseline(Beat,5);
    r = FP(i,4) - FP(i,1) + 5 + 1;
    Beat = frame_beat(Beat,r,FrameSize);
    FP(i,1:3) = max(1,center - (FP(i,4) - FP(i,1:3)));
    FP(i,5:7) = min(FrameSize,center + (FP(i,5:7) - FP(i,4)));
    FP(i,4) = center;
    Beats(:,i) = Beat;
end

function Result = polyfit_baseline(Beat,l)
n = length(Beat);
y0 = mean(Beat(1:l));
y1 = mean(Beat(end-l+1:end));
P = polyfit([0 n-1],[y0 y1],1);
Result = polyval(P,(0:n-1)');

function Result = frame_beat(Beat, r, L)
half = floor(L/2);
N = length(Beat);
trim1 = max(0,r-1-half);
pad1 = max(0,half-r+1);
trim2 = max(0,N-r-half);
pad2 = max(0,half-N+r);
Result = [zeros(pad1,1); Beat(trim1+1:end-trim2); zeros(pad2,1)];
```

Código-fonte A.7 – Função build_template

```
function [Templates,ArtifactIndices] = build_template(Beats,Tc)
[M,N] = size(Beats);
Templates = zeros(M,N);
Template = zeros(M,1);
ArtifactIndices = false(1,N);
thr = [0 0];
r = 1/Tc;
for i = 1:N
    metric = rms(Template - Beats(:,i))
    if (i == 1)
        % primeiro batimento
        Template = Beats(:,i);
    elseif (i <= Tc || metric < thr(1))
        % batimento excelente
        Template = Template + r * (Beats(:,i) - Template);
        thr = thr + r * ([3*metric 4*metric] - thr);
    elseif metric > thr(2)
        % batimento ruim
        ArtifactIndices(i) = true;
    else
        % batimento bom
```

```
    end
    Templates(:,i) = Template;
end
```

Código-fonte A.8 – Função preprocess

```
function [R,RR,FP,Beats,Template,delay] = preprocess(signal, Fm)
% obtem as amostras e a taxa de amostragem
data = signal.data - signal.inival;
Fs = signal.fs;
% faz a filtragem do sinal
[sigI,delay] = qrs_filter(data, Fs);
[sigD,sigM] = fp_filter(data, Fs, delay);
sigN = noise_filter(data, Fs, Fm, delay);
% faz a deteccao de pontos e de batidas
[R,RR] = detect_qrs(sigI, Fs);
FP = detect_fp(sigD, sigM, R, Fs);
[Beats,FP] = extract_beats(sigN, FP, Fs);
[Templates,idx] = build_template(Beats, 30);
Template = Templates(:,30);
% remove os artefatos
R(idx) = [];
RR(idx) = [];
FP(:,idx) = [];
Beats(:,idx) = [];
```

A.2 Extração de características

Código-fonte A.9 – Função rocha_features

```
function [C,S1,S2] = rocha_features(signal, RR, F, Beats)

center = floor(size(Beats,1)/2)+1;
Fs = signal.fs;
gain = signal.gain;

% extracao do ponto J de acordo com Pang
Jp = features.pang_jpoints(F(4,:)', RR, Fs);

% extracao dos pontos isoeletrico e J de acordo com Rocha
[Ir,Jr] = features.rocha_ijpoints(Beats, center, F(3,:)', F(5,:)', Fs, gain);

% extracao dos segmentos de batida
[S1,S2] = features.rocha_segments(Beats, Ir, Jr, F(end,:)');

% extracao dos desvios de segmento ST
C0 = features.rocha_stdev(Beats, Ir, Jp, Jr);

% extracao dos coeficientes de hermite
[C1,C2] = features.rocha_hermite(S1, S2);

% composicao das caracteristicas
C = [C0; C1; C2];
```

Código-fonte A.10 – Função pang_jpoints

```
function J = pang_jpoints(R, RR, Fs)

% calcula a frequencia cardiaca e os intervalos
HR = RR*60./Fs;
A = HR < 100;
B = 100 <= HR & HR <= 110;
C = 100 <= HR & HR <= 11;
D = 120 < HR;

% calcula o ponto de medida de acordo com a frequencia cardiaca
L = zeros(size(R));
L(A) = 0.120;
L(B) = 0.112;
L(C) = 0.104;
L(D) = 0.100;

% calcula o ponto J
J = R + round(L.*Fs);
```

Código-fonte A.11 – Função rocha_ijpoints

```
function [I,J] = rocha_jpoints(Beats, R, defI, defJ, Fs, gain)

[bi,ai,gi,di] = intfdesign.maverage('Width',0.05,Fs);
[bd,ad,~,dd] = intfdesign.derivative('N,M',1,0);
delay = ceil(di + dd);

L1 = round(0.02*Fs);
L2 = round(0.12*Fs);
thr = gain*1.25/Fs;
M = size(Beats,2);
I = zeros(M,1);
J = zeros(M,1);
for i = 1:M
    sigI = filter(bi, ai, Beats(:,i)) ./ gi;
    sigD = filter(bd, ad, sigI);

    % detection of isoelectric point
    istart = R - L2 + delay;
    iend = R - L1 + delay;
    I(i) = edge_detection(abs(sigD), istart, iend, L1, thr, default(i)) - delay;

    % detection of jay point
    istart = R + L1 + delay;
    iend = R + L2 + delay;
    J(i) = edge_detection(abs(sigD), istart, iend, L1, thr, default(i)) - delay;
end
```

Código-fonte A.12 – Função edge_detection

```
function pos = edge_detection(data, istart, iend, L, thr, default)

M = floor(L/2);
pos = default;
```

```
for i = istart+M:iend-M
    if all(data(i-M:i+M) < thr)
        pos = i;
        break;
    end
end
```

Código-fonte A.13 – Função rocha_stdev

```
function C = rocha_stdev(Beats, I, J0, J1)

M = size(Beats,2);
C = zeros(2,M);
for i = 1:M
    C(1,i) = Beats(J0(i),i) - Beats(I(i),i);
    C(2,i) = Beats(J1(i),i) - Beats(I(i),i);
end
```

Código-fonte A.14 – Função rocha_segments

```
function [S1,S2] = rocha_segments(Beats, B, J, E)

M = size(Beats,2);
S1 = zeros(64,M);
S2 = zeros(64,M);
for i = 1:M
    if (B(i) <= J(i))
        segment1 = Beats(B(i):J(i),i);
        S1(:,i) = resample(segment1,64,J(i)-B(i)+1);
    end
    if (J(i) <= E(i))
        segment2 = Beats(J(i):E(i),i);
        S2(:,i) = resample(segment2,64,E(i)-J(i)+1);
    end
end
```

Código-fonte A.15 – Função rocha_hermite

```
function [C1,C2] = rocha_hermite(S1, S2)

[N,M] = size(S1);
C1 = zeros(6,M);
C2 = zeros(6,M);
H1 = math.hermite_matrix(N,6,5);
H2 = math.hermite_matrix(N,6,8);
H1PI = (H1'*H1)\H1';
H2PI = (H2'*H2)\H2';
for i = 1:M
    C1(:,i) = H1PI * S1(:,i);
    C2(:,i) = H2PI * S2(:,i);
end
```

Código-fonte A.16 – Função mohebbi_features

```
function [C,STb] = mohebbi_features(signal, F, Beats, Template)

center = floor(size(Beats,1)/2)+1;
defJ = mean(F(5,1:30));
Fs = signal.fs;
gain = signal.gain;

% extracao dos pontos isoeletrico e J de acordo com Mohebbi
Jt = features.mohebbi_jpoints(Template, center, defJ, Fs, gain);
Jb = features.mohebbi_jpoints(Beats, center, F(5,:)', Fs, gain);

% extracao dos segmentos ST (demorado)
STt = features.mohebbi_segments(Template, Jt, Fs);
STb = features.mohebbi_segments(Beats, Jb, Fs);

% extracao das diferencas entre os segmentos e o do template
C = features.mohebbi_stdiff(STt, STb);
```

Código-fonte A.17 – Função mohebbi_jpoints

```
function J = mohebbi_jpoints(Beats, R, default, Fs, gain)

[bi,ai,gi,di] = intfdesign.maverage('Width',0.05,Fs);
[bd,ad,~,dd] = intfdesign.derivative('N,M',1,0);
delay = ceil(di + dd);

L1 = round(0.02*Fs);
L2 = round(0.12*Fs);
thr = gain*1.25/Fs;
M = size(Beats,2);
J = zeros(M,1);
for i = 1:M
    sigI = filter(bi, ai, Beats(:,i)) ./ gi;
    sigD = filter(bd, ad, sigI);
    istart = R + L1 + delay;
    iend = R + L2 + delay;
    J(i) = edge_detection(abs(sigD), istart, iend, L1, thr, default(i)) - delay;
end
```

Código-fonte A.18 – Função mohebbi_segments

```
function Result = mohebbi_segments(Beats, J, Fs)

M = size(Beats,2);
L = round(0.08*Fs)*2;
Result = zeros(20,M);
for i = 1:M
    segment = Beats(J(i):J(i)+L-1,i);
    Result(:,i) = resample(segment,20,L);
end
```

Código-fonte A.19 – Função mohebbi_stdiff

```
function C = mohebbi_stdiff(Sref, Stest)

[N,M] = size(Stest);
C = zeros(N,M);
for i = 1:M
    C(:,i) = Stest(:,i) - Sref;
end
```

Código-fonte A.20 – Função gopalak_features

```
function [C,S] = gopalak_features(RR, Beats)

% extracao dos segmentos de batida
S = features.gopalak_segments(Beats, RR);

% extracao dos coefficientes de hermite
C = features.gopalak_hermite(S);
```

Código-fonte A.21 – Função gopalak_segments

```
function Result = gopalak_segments(Beats, RR)

[N,M] = size(Beats);
RR = min(N,RR);
Result = zeros(250,M);
for i = 1:M
    off = floor((N-RR(i))/2);
    segment = Beats(off+1:off+RR(i),i);
    Result(:,i) = resample(segment,250,RR(i));
end
```

Código-fonte A.22 – Função gopalak_hermite

```
function C = gopalak_hermite(Segments)

[N,M] = size(Segments);
C = zeros(50,M);
H = math.hermite(N,50,1.5)';
for i = 1:M
    C(:,i) = H * Segments(:,i);
end
```

Código-fonte A.23 – Função hermite

```
function Result = hermite(N, M, b)

% calculate vectors
Tb = math.tridiagonal_matrix(N, b);
[V,~] = eigs(Tb,M,'LA');
```

```
% correct polarity
Result = zeros(N,M);
k = floor(N/2)+1;
for m = 0:M-1
    Result(:,m+1) = V(:,m+1) .* sign(V(k,m+1)) * (-1)^floor(m/2);
end

function Result = tridiagonal_matrix(n, b)
i = 0:n-1;
j = 1:n-1;
tau = 1/(n*b^2);
D0 = -2*cos(pi*n*tau)*sin(pi*i*tau).*sin(pi*(n-i-1)*tau);
D1 = sin(pi*j*tau).*sin(pi*(n-j)*tau);
Result = diag(D0) + diag(D1,1) + diag(D1,-1);
```

A.3 Classificação

Código-fonte A.24 – Função train_network

```
function [net,tr] = train_network(inputs, targets, layers, trainfcn, ...
    transfcn, processfcn)

net = feedforwardnet(layers, trainfcn);
net.layers{:}.transferFcn = transfcn;
net.inputs{1}.processFcns = {processfcn};

net.divideFcn = 'dividerand';
net.divideMode = 'sample';
net.divideParam.trainRatio = 0.7;
net.divideParam.valRatio = 0.15;
net.divideParam.testRatio = 0.15;

net.trainParam.showWindow = false;
net.trainParam.showCommandLine = true;
net.trainParam.show = 20;
net.trainParam.time = 3*60;
net.trainParam.epochs = 1000;
net.trainParam.max_fail = 10;
net.trainParam.min_grad = 1e-10;

net.plotFcns = {'plotconfusion', 'plotperform', 'plottrainstate', ...
    'ploterrhist', 'plotregression'};

[net,tr] = train(net,inputs,targets);
```

Código-fonte A.25 – Função train_networks

```
function Result = train_networks(leadnames, datasets, trainfcn, transfcn, processfcn)

for i = 1:length(leadnames)
    lead = leadnames{i};
    if ~isfield(datasets, lead)
```

```
        continue;
    end
    disp(['Generating networks for lead ' lead '...']);
    [net,tr,stat] = inner_train(datasets, lead, trainfcn, transfcn, processfcn);
    Result.(lead).net = net;
    Result.(lead).tr = tr;
    Result.(lead).stat = stat;
end

function [net,tr,stat] = inner_train(datasets, lead, trainfcn, transfcn, processfcn)
layers = randi([10 20],1);
x = datasets.(lead).inputs;
t = datasets.(lead).targets;
[net,tr] = nnetwork.train_network(x, t, layers, trainfcn, transfcn, processfcn);
stat = utils.compute_statistics(t > 0, net(x) > 0);
```

Código-fonte A.26 – Função generate_datasets

```
function Result = generate_datasets(leadnames, basedir, methodname, ...
    recordmap, discard, countmap, ratiomap, feature)

for i = 1:length(leadnames)
    lead = leadnames{i};
    if countmap(lead) == 0
        continue;
    end

    if feature == 0
        disp(['Processing lead ' lead ' for ST segment diagnosis...']);
    else
        disp(['Processing lead ' lead ' for T wave diagnosis...']);
    end

    [inputs,targets] = selection.generate_lead_dataset(...
        basedir, lead, methodname, recordmap(lead), ...
        discard, countmap(lead), ratiomap(lead), feature);

    Result.(lead) = struct('inputs',inputs,'targets',targets);
end
```

Código-fonte A.27 – Função generate_lead_dataset

```
function [inputs,targets] = generate_lead_dataset(basedir, leadname, ...
    methodname, records, discard, selcount, ratio, feature)

files = dir([basedir '*.mat']);
list = cell(1,length(files));
rowcount = 0;
colcount = 0;
for i = 1:length(files)
    file = files(i);
    if isempty(file.name) || file.isdir
        continue;
    end
    [~,name,~] = fileparts(file.name);
```

```
    if discard == ismember(name, records)
        continue;
    end
    filepath = [basedir file.name];
    if isempty(who('-file', filepath, leadname))
        continue;
    end
    info = load(filepath, leadname);
    list{i} = info.(leadname).Datasets.(methodname);
    rowcount = rowcount + size(list{i},1);
    if colcount == 0
        colcount = size(list{i},2);
    end
end

dataset = zeros(rowcount,colcount);
iend = 0;
for i = 1:length(list)
    ibegin = iend + 1;
    iend = iend + size(list{i},1);
    dataset(ibegin:iend,:) = list{i};
end

if isempty(selcount)
    indices = true(rowcount,1);
else
    indices = selection.select_rows(dataset, selcount, ratio, feature);
end

inputs = dataset(indices,1:end-2)';
targets = 2*(dataset(indices,end-1+feature)'~=0)-1;
```

Código-fonte A.28 – Função generate_configurations

```
function generate_configurations
global edbleadnames

basedir = 'C:\physiobank\database\edb\';
count = utils.count_classes(basedir);
load('matfiles\indstats.mat');
lists = get_selection_lists(tables);
merge = get_merged_selection(lists);

%% configuration 1
[stratiomap,tratiomap] = get_edb_ratio_maps(count);
temp = cell2mat(stratiomap.values');
temp = [temp(:,1) temp(:,2)+temp(:,3)];
mohebbiratiomap = containers.Map(edbleadnames, num2cell(temp,2));
[rocharecordmap,mohebbirecordmap,gopalakrecordmap] = ...
    get_original_record_maps();
[rochacountmap,mohebbicountmap,gopalakcountmap] = get_original_count_maps();

save('matfiles\configuration1.mat', 'stratiomap', 'tratiomap', ...
    'mohebbiratiomap', 'rocharecordmap', 'mohebbirecordmap', ...
    'gopalakrecordmap', 'rochacountmap', 'mohebbicountmap', 'gopalakcountmap');

%% configuration 2
```

```
ratiomap = containers.Map(edbleadnames, {[] [] [] [] [] [] [] []});
[rocharecordmap,mohebbirecordmap,gopalakstrecordmap,gopalaktrecordmap] = ...
    get_proposed_record_maps(lists, basedir);
countmap = containers.Map(edbleadnames, {[] [] [] [] [] [] [] []});

save('matfiles\configuration2.mat', 'ratiomap', 'rocharecordmap', ...
    'mohebbirecordmap', 'gopalakstrecordmap', 'gopalaktrecordmap', ...
    'countmap');

%% configuration 3
[strecordmap,trecordmap] = get_merged_record_maps(merge, basedir);
countmap = containers.Map(edbleadnames, {[] [] [] [] [] [] [] []});

save('matfiles\configuration3.mat', 'ratiomap', 'strecordmap', ...
    'trecordmap', 'countmap');

%% configuration 4
[stratiomap,tratiomap] = get_edb_ratio_maps(count);
countmap = containers.Map(edbleadnames, ...
    {5000 20000 100000 30000 20000 10000 100000 100000});

save('matfiles\configuration4.mat', 'stratiomap', 'tratiomap', ...
    'strecordmap', 'trecordmap', 'countmap');

%% configuration 5
ratiomap = containers.Map(edbleadnames, {[] [] [] [] [] [] [] []});
recordmap = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
countmap = containers.Map(edbleadnames, {[] [] [] [] [] [] [] []});

save('matfiles\configuration5.mat', 'ratiomap', 'recordmap', 'countmap');

function [stratiomap,tratiomap] = get_edb_ratio_maps(count)
global edbleadnames
N = length(edbleadnames);
ST = zeros(N,3);
T = zeros(N,3);
for i = 1:N
    lead = edbleadnames{i};
    total = sum(sum(count.(lead)));
    ST(i,:) = sum(count.(lead),2)/total;
    T(i,:) = sum(count.(lead),1)/total;
end
stratiomap = containers.Map(edbleadnames, num2cell(ST,2));
tratiomap = containers.Map(edbleadnames, num2cell(T,2));

function [rocha,mohebbi,gopalak] = get_original_record_maps()
global edbleadnames
rocha = containers.Map(edbleadnames, {...
    {}
    {'e0207'}
    {'e0109' 'e0121' 'e0609' 'e0613'}
    {'e0403'}
    {'e0415' 'e0603'}
    {}
    {'e0119' 'e0161'}
    {'e0207' 'e0213' 'e0303' 'e0405'}
});
mohebbi = containers.Map(edbleadnames, {...
    {}
```

```
    {}
    {'e0103' 'e0105' 'e0113' 'e0119' 'e0147'}
    {}
    {}
    {}
    {'e0103' 'e0105' 'e0113' 'e0119' 'e0147'}
    {}
});
gopalak = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});

function [rocha,mohebbi,gopalak] = get_original_count_maps()
global edbleadnames
rocha = containers.Map(edbleadnames, ...
    {1465 56990 133477 30548 35110 14487 108107 162747});
mohebbi = containers.Map(edbleadnames, ...
    {0 0 18047 0 0 0 18047 0});
gopalak = containers.Map(edbleadnames, ...
    {236 236 236 236 236 236 236 236});

function lists = get_selection_lists(tables)
lists = cell(length(tables),1);
for i = 1:length(tables)
    mytable = tables{i};
    sel1 = mytable.Sensitivity >= 0.8;
    sel2 = mytable.PositivePred >= 0.8;
    lists{i} = mytable(sel1 & sel2, 'RecordName');
end

function merge = get_merged_selection(lists)
merge = cell(4,1);
for i = 1:2
    list1 = lists{3*(i-1)+1};
    list2 = lists{3*(i-1)+2};
    list3 = lists{3*(i-1)+3};
    list4 = union(union(list1{:,:},list2{:,:}),list3{:,:});
    idx = true(size(list4));
    for j = 1:length(list4)
        name = list4{j};
        cond1 = any(strcmp(name, list1{:,:}));
        cond2 = any(strcmp(name, list2{:,:}));
        cond3 = any(strcmp(name, list3{:,:}));
        if cond1 + cond2 + cond3 <= 1
            idx(j) = false;
        end
    end
    merge{i} = array2table(list4(idx),'VariableNames',{'RecordName'});
end
merge{3} = lists{end-1};
merge{4} = lists{end};

function [rocha,mohebbi,gopalakst,gopalakt] = get_proposed_record_maps(lists, basedir)
global edbleadnames
rocha = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
mohebbi = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
gopalakst = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
gopalakt = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
for i = 1:2
    list = lists{3*(i-1)+1}{:,:};
    for j = 1:length(list)
```

```
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        rocha(lead) = [rocha(lead); list{j}];
    end
    list = lists{3*(i-1)+2}{:,:};
    for j = 1:length(list)
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        mohebbi(lead) = [mohebbi(lead); list{j}];
    end
    list = lists{3*(i-1)+3}{:,:};
    for j = 1:length(list)
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        gopalakst(lead) = [gopalakst(lead); list{j}];
    end
end
for i = 1:2
    list = lists{i+6}{:,:};
    for j = 1:length(list)
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        gopalakt(lead) = [gopalakt(lead); list{j}];
    end
end

function [strecordmap,trecordmap] = get_merged_record_maps(merge, basedir)
global edbleadnames
strecordmap = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
trecordmap = containers.Map(edbleadnames, {{} {} {} {} {} {} {} {}});
for i = 1:2
    list = merge{i}{:,:};
    for j = 1:length(list)
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        strecordmap(lead) = [strecordmap(lead); list{j}];
    end
    list = merge{i+2}{:,:};
    for j = 1:length(list)
        record = load([basedir list{j} '.mat']);
        lead = record.Info(i).Description;
        trecordmap(lead) = [trecordmap(lead); list{j}];
    end
end
```

Código-fonte A.29 – Função select_rows

```
function Result = select_rows(dataset, selcount, ratio, feature)

if length(ratio) == 1
    if size(dataset,1) < selcount
        warning('number of beats is less than expected: %d < %d', ...
            size(dataset,1), selcount);
        selcount = size(dataset,1);
    end
    Result = sort(randperm(size(dataset,1), selcount));
```

```
elseif length(ratio) == 2
    normalcount = ceil(ratio(1)*selcount);
    ischemiccount = selcount - normalcount;

    normal = find(dataset(:,end-1+feature) == 0);
    ischemic = find(dataset(:,end-1+feature) ~= 0);

    if length(normal) < normalcount
        warning('number of normal beats is less than expected: %d < %d', ...
            length(normal), normalcount);
        normalcount = length(normal);
        ischemiccount = ceil(normalcount*ratio(2)/ratio(1));
        fprintf('new count: %d\n', normalcount + ischemiccount);
    end
    if length(ischemic) < ischemiccount
        warning('number of ischemic beats is less than expected: %d < %d', ...
            length(ischemic), ischemiccount);
        ischemiccount = length(ischemic);
        normalcount = ceil(ischemiccount*ratio(1)/ratio(2));
        fprintf('new count: %d\n', normalcount + ischemiccount);
    end

    indices1 = randperm(length(normal), normalcount);
    indices2 = randperm(length(ischemic), ischemiccount);
    Result = unique([normal(indices1); ischemic(indices2)]);

elseif length(ratio) == 3
    normalcount = ceil(ratio(1)*selcount);
    ischemiccount = selcount - normalcount;
    elevcount = ceil(ratio(2)/(ratio(2)+ratio(3))*ischemiccount);
    depcount = ischemiccount - elevcount;

    normal = find(dataset(:,end-1+feature) == 0);
    elev = find(dataset(:,end-1+feature) > 0);
    dep = find(dataset(:,end-1+feature) < 0);

    if length(normal) < normalcount
        warning('number of normal beats is less than expected: %d < %d', ...
            length(normal), normalcount);
        normalcount = length(normal);
        elevcount = ceil(normalcount*ratio(2)/ratio(1));
        depcount = ceil(normalcount*ratio(3)/ratio(1));
        fprintf('new count: %d\n', normalcount + elevcount + depcount);
    end
    if length(elev) < elevcount
        warning('number of beats with elevation is less than expected: %d < %d', ...
            length(elev), elevcount);
        elevcount = length(elev);
        normalcount = ceil(elevcount*ratio(1)/ratio(2));
        depcount = ceil(elevcount*ratio(3)/ratio(2));
        fprintf('new count: %d\n', normalcount + elevcount + depcount);
    end
    if length(dep) < depcount
        warning('number of beats with depression is less than expected: %d < %d', ...
            length(dep), depcount);
        depcount = length(dep);
        normalcount = ceil(depcount*ratio(1)/ratio(3));
        elevcount = ceil(depcount*ratio(2)/ratio(3));
        fprintf('new count: %d\n', normalcount + elevcount + depcount);
```

```
    end

    indices1 = randperm(length(normal), normalcount);
    indices2 = randperm(length(elev), elevcount);
    indices3 = randperm(length(dep), depcount);
    Result = unique([normal(indices1); elev(indices2); dep(indices3)]);

else
    error('length of ratio must be between 1 and 3');
end
```

A.4 Utilidades

Código-fonte A.30 – Função compute_statistics

```
function Result = compute_statistics(Known, Predicted)
% Computa as estatisticas de um diagnostico com base numa referencia

if length(Known) ~= length(Predicted)
    error('parameter size mismatch');
end

VP = length(find( Known &  Predicted));     % verdadeiros positivos
VN = length(find(~Known & ~Predicted));     % verdadeiros negativos
FP = length(find(~Known &  Predicted));     % falsos positivos
FN = length(find( Known & ~Predicted));     % falsos negativos
TT = length(Known);                         % total conhecido

Result = [
    VP/(VP+FN)      % sensibilidade
    VN/(VN+FP)      % especificidade
    VP/(VP+FP)      % preditividade positiva
    VN/(VN+FN)      % preditividade negativa
    (VP+VN)/TT      % acuracia
    (FP+FN)/TT      % taxa de falha
];
```

Código-fonte A.31 – Função count_classes

```
function Result = count_classes(basedir)

files = dir([basedir '*.mat']);
Result = struct;

for i = 1:length(files)
    file = files(i);
    if isempty(file.name) || file.isdir
        continue;
    end
    disp(['Processing ' file.name '...']);

    record = load([basedir file.name]);
    for j = 1:record.SignalCount
```

```
        lead = record.Info(j).Description;
        atr = record.Annotations.atr;
        idx = ismember(atr.Type, 'NLRBAaJSVFejnE/fQ?');
        qrs = record.Annotations.atr.Sample(idx);
        ST = utils.extract_diagnosis(qrs, atr, 's', 'ST', j-1);
        T = utils.extract_diagnosis(qrs, atr, 'T', 'T', j-1);

        stnormal = ST.diagVal == 0;
        stelevated = ST.diagVal > 0;
        stdepressed = ST.diagVal < 0;
        tnormal = T.diagVal == 0;
        televated = T.diagVal > 0;
        tdepressed = T.diagVal < 0;

        C = zeros(3,3);
        C(1,1) = C(1,1) + length(find(stnormal & tnormal));
        C(2,1) = C(2,1) + length(find(stelevated & tnormal));
        C(3,1) = C(3,1) + length(find(stdepressed & tnormal));
        C(1,2) = C(1,2) + length(find(stnormal & televated));
        C(2,2) = C(2,2) + length(find(stelevated & televated));
        C(3,2) = C(3,2) + length(find(stdepressed & televated));
        C(1,3) = C(1,3) + length(find(stnormal & tdepressed));
        C(2,3) = C(2,3) + length(find(stelevated & tdepressed));
        C(3,3) = C(3,3) + length(find(stdepressed & tdepressed));

        if ~isfield(Result, lead)
            Result.(lead) = C;
        else
            Result.(lead) = Result.(lead) + C;
        end
    end
end
```

Código-fonte A.32 – Função extract_diagnosis

```
function Result = extract_diagnosis(Rk, atr, type, keyWord, id)

isqtable = atr(atr.Type == type, {'Sample', 'Comment'});

M = length(Rk);
N = height(isqtable);
L = length(keyWord);

Result.diagVal = zeros(M,1);
Result.peakVal = zeros(M,1);

k = 1;
while k <= N
    aux = isqtable.Comment{k};
    if (~isempty(aux) && ...
            aux(1) == '(' && ...
            strcmp(aux(1+(1:L)), keyWord) && ...
            str2double(aux(2+L)) == id)
        % inicio de episodio
        beginIndex = isqtable.Sample(k);
        ch = aux(3+L);
```

```
        % procura a marcacao de pico
        k = k + 1;
        while k <= N
            aux = isqtable.Comment{k};
            if (~isempty(aux) && ...
                    upper(aux(1)) == 'A' && ...
                    strcmp(aux(1+(1:L)), keyWord) && ...
                    str2double(aux(2+L)) == id)
                % fim de episodio
                peak = str2double(aux(4+L:end));
                break;
            else
                k = k + 1;
            end
        end

        % procura a marcacao de termino
        k = k + 1;
        while k <= N
            aux = isqtable.Comment{k};
            if (~isempty(aux) && ...
                    aux(end) == ')' && ...
                    strcmp(aux(1:L), keyWord) && ...
                    str2double(aux(1+L)) == id)
                % fim de episodio
                endIndex = isqtable.Sample(k);
                break;
            else
                k = k + 1;
            end
        end

        if k <= N
            % verifica se e elevacao ou depressao
            if ch == '+'
                j = beginIndex <= Rk & Rk <= endIndex;
                Result.diagVal(j) = 1;
                Result.peakVal(j) = peak;
            elseif ch == '-'
                j = beginIndex <= Rk & Rk <= endIndex;
                Result.diagVal(j) = -1;
                Result.peakVal(j) = peak;
            end
        end
    end
    k = k + 1;
end
```

Código-fonte A.33 – Função match_qrs

```
function [A, B] = match_qrs(Rref, Rtest, Fs)
%   Obtem os incides das batidas que foram corretamente idfentificadas.
%
% Entradas:
%   R1 - localizacao correta dos picos da onda R
%   R2 - localizacao predita dos picos da onda R
%
```

```
% Saida:
%   incides das batidas identificadas
%
VDI = round(0.15*Fs);

n = length(Rref);
m = length(Rtest);
A = false(n,1);
B = false(m,1);

Rref = [Rref(:); Inf];
Rtest = [Rtest(:); Inf];

i = 1;
j = 1;
while (i <= n) && (j <= m)
    % get differences
    d1 = Rref(i) - Rtest(j);
    d2 = Rref(i+1) - Rtest(j+1);

    % check precedence
    if d1 > 0
        % test annotation is earliest
        d3 = Rref(i) - Rtest(j+1);

        if (d1 <= VDI && (d1 < abs(d3) || abs(d2) < abs(d3)))
            % (1) current test annotation is within the validation interval
            % and is a better match than the next test annotation: pair it
            A(i) = true;
            B(j) = true;
            i = i + 1;
            j = j + 1;
        else
            % (2) there is no match to the current test annotation. hence,
            % do not do anything and go to the next test annotation.
            j = j + 1;
        end
    else
        % reference annotation is earliest
        d4 = Rref(i+1) - Rtest(j);

        if (-d1 <= VDI && (-d1 < abs(d4) || abs(d2) < abs(d4)))
            % (3) current ref. annotation is within the validation interval
            % and is a better match than the next ref. annotation: pair it
            A(i) = true;
            B(j) = true;
            j = j + 1;
            i = i + 1;
        else
            % (4) There is no match to the current ref. annotation. hence,
            % do not do anything and go to the next ref. annotation.
            i = i + 1;
        end
    end
end
```