Compressão de Sinais Biomédicos

Hugo Neves de Oliveira

Dissertação apresentada à Universidade Federal da Paraíba, para a obtenção de Título de Mestre em Ciência da Computação, na área de Sinais, Sistemas Digitais e Gráficos.

Orientador: Leonardo Vidal Batista

João Pessoa, PB
Fevereiro 2016

Catalogação na publicação
Seção de Catalogação e Classificação

O48c Oliveira, Hugo Neves de.
Compressão de Sinais Biomédicos / Hugo Neves de Oliveira. - João Pessoa, 2019.
209 f. : il.

Dissertação (Mestrado) - UFPB/CI.

1. compressão de dados. 2. sinais biomédicos. 3. transformadas de domínio. 4. quantização vetorial. 5. codificadores de entropia. I. Título

UFPB/BC

Comissão Julgadora:

Prof. Dr.
Hugo Leonardo Davi de Souza Cavalcante

Profª. Drª.
Thaís Gaudencio do Rêgo

Prof. Dr.
Leonardo Vidal Batista

Prof. Dr.
Elmar Uwe Kurt Melcher

Hugo Neves de Oliveira
Compressão de Sinais Biomédicos
209 páginas
Dissertação (Mestrado) - Universidade Federal da Paraíba. Departamento de Informática.

1. Compressão de Dados
2. Sinais Biomédicos
3. Compressão Baseada em Transformadas

Universidade de Federal da Paraíba. Centro de Informática. Departamento de Informática.

Aquilo que jaz no coração de cada ser vivo não é uma chama, nem um hálito quente nem uma "faísca de vida". É a informação, palavras, instruções. Se quiser uma metáfora, não pense em fogos, faíscas ou hálitos. Pense em vez disso num bilhão de caracteres distintos gravados em tabuletas de cristal.

— Richard Dawkins, O Relojoeiro Cego, 1986

Agradecimentos

Expresso por meio deste a minha gratidão a todos que ajudaram de alguma forma na realização deste trabalho, em especial:

Ao meu orientador, o Professor Leonardo Vidal Batista que, durante inúmeras discussões profissionais e conversas amigáveis, mostrou-se um verdadeiro tutor, orientando não só para o trabalho, mas também para a vida em muitos aspectos.

À Professora Thaís Gaudencio do Rêgo que me ensinou bastante não apenas sobre o viés técnico, mas também sobre o lado humano das exatas e do ambiente acadêmico.

A todos os outros professores que ministraram aulas no meu curso de Graduação e Mestrado, pela paciência e sapiência com as quais trataram a mim e aos meus colegas de turma nesses árduos seis anos e meio de luta.

Aos grandes amigos Yuri Dantas, Glauco Sousa, Igor Malheiros, José Ivan, Arnaldo Gualberto, Ygor Crispim e Raul Felipe, por desempenharem papéis tão maiores que meros companheiros de trabalho e estudo.

Aos membros do PET.Com, pelos diversos momentos de ensinamento, discussão, descontração, reflexão, e, inclusive, pelas desavenças, as quais ajudaram a fundamentar minhas bases éticas, humanas e profissionais.

A todos os meus colegas de turma, de laboratórios e de pesquisa, por desenvolverem comigo relacionamentos de ajuda múltipla desde o início.

À minha mãe, por fornecer apoio psicológico em meio a todo tipo de adversidades e pelo amor transparecido em todos os momentos.

Resumo

Compressores de eletrocardiogramas e eletroencefalogramas vêm sendo descritos na literatura ao longo das últimas décadas, mas há uma falta de trabalhos que apresentem soluções gerais para todos os sinais biomédicos. Visando preencher essa lacuna, este trabalho discute métodos de compressão que funcionem satisfatoriamente para os sinais biomédicos. Os métodos de compressão sem perdas de informação estudados se baseiam em transformadas *wavelet* e preditores lineares associados a diversos codificadores de entropia. As estratégias de compactação com perdas de informação apresentadas usam transformadas trigonométricas e *wavelets* seguidas por quantizadores vetoriais baseados em quantização com zona-morta gerados por Minimização Lagrangiana e em Quantizações por Aproximações Sucessivas. Os codificadores de entropia usados se baseiam em Predição por Casamento Parcial, Codificação por Comprimento de Sequência e Particionamento de Conjuntos em Árvores Hierárquicas. Os métodos foram testados usando os sinais do MIT/BIH *Polysomnographic Database*. Os compressores sem perdas de informação obtiveram razões de compressão até 4,818 : 1, dependendo do tipo de registro biomédico, enquanto os métodos com perdas obtiveram razões de compressão até 818,055 : 1. Os sinais mais suaves, como os de respiração e saturação de oxigênio, apresentaram melhores reconstruções com *wavelets* e Quantização por Aproximações Sucessivas, apesar dos menores desempenhos de compressão. Contrastivamente, para a mesma qualidade de reconstrução visual, transformadas trigonométricas e Minimização Lagrangiana obtiveram melhores eficácias de compressão para sinais mais variáveis – como eletroencefalogramas e eletromiogramas.

Palavras-chave: compressão de dados, sinais biomédicos, transformadas de domínio, quantização vetorial, codificadores de entropia

Abstract

Compressors for electrocardiograms and electroencephalograms have been reported in the literature over the last decades, but there is a lack of works describing general solutions for all biomedical signals. Aiming to fill this gap, this work discusses compression methods that work well for all biomedical signals. The lossless compression methods are based in wavelet transforms and linear predictors associated with several entropy coders. The lossy compaction strategies use trigonometric and wavelet transforms, followed by vector quantizers based in dead-zone quantization by Lagrangian Minimization and Successive Approximation Quantizations. The entropy coders are based in Prediction by Partial Matching, Run Length Encoding and Set Partitioning in Hierarchical Trees. The methods were tested usign the signals of the MIT/BIH Polysomnographic Database. Lossless compressors achieved compression ratios of at most $4.818:1$, while the lossy methods achieved compression ratios of at most $818.055:1$. Smoother signals, as respiration and oxygen saturation records, presented better reconstructions with wavelets and Successive Approximation Quantization, despite the lower compression performances. Contrastively, for the same quality of visual reconstruction, trigonometric transforms and Lagrangian Minimization achieved better compression performances for rougher signals – as electroencephalograms and electromyograms.

Keywords: data compression, biomedical signals, domain transforms, vector quantization, entropy coders

Lista de Figuras

Lista de Tabelas

Glossário

***Fast* PPM** Variação do algoritmo do PPM.

AC Corrente Alternada (*Alternate Current*): termo usado para denotar coeficientes de índice diferente de 0 da DCT.

ADPCM *Adaptive Differential Pulse-Code Modulation*: CODEC de áudio baseado em codificação diferencial adaptativa.

BCI Borda Convexa Inferior: borda de um polígono convexo mais próxima da origem de um plano cartesiano.

BitGol Variação de um codificador de Golomb apresentado neste trabalho.

BP Pressão Sanguínea (*Blood Pressure*): sinal biomédico que detecta variações na pressão sanguínea.

BPPM Predição por Casamento Parcial Binário (*Binary Prediction by Partial Matching*): variação binária do algoritmo do PPM.

BWT Transformada de Burrows-Wheeler (*Burrows-Wheeler Transform*): transformada que visa o aumento da suscetibilidade de uma mensagem a outras técnicas de compressão.

CAB Corte e Alinhamento (*Cut and Align*): estratégia usada para seccionar ECGs e reorganizá-los para melhorar o desempenho de transformações de domínio.

CDF Função de Distribuição Cumulativa (*Cumulative Distribution Function*): funções altamente usadas para verificar a similaridade de conjuntos de dados com certas distribuições estatísticas.

CODEC *Compressor/Decompressor*: conjunto completo de uma estratégia de compressão composto do codificador e do decodificador.

D Distorção (*Distortion*).

D4 Daubechies 4: base *wavelet*.

DC Corrente Contínua (*Direct Current*): termo usado para denotar coeficientes de índice 0 da DCT.

DCT Transformada Discreta do Cosseno (*Discrete Cosine Transform*): transformada derivada da função cosseno que é altamente utilizada na compressão de dados e processamento de sinais.

DCT+MinLag Método de compressão *lossy* apresentado neste trabalho.

DHT Transformada Haar Discreta (*Discrete Haar Transform*): transformada *wavelet* baseada nas funções de Haar.

DPCM Modulação por Código de Pulso Diferencial (*Differential Pulse-Code Modulation*): método de codificação diferencial.

DST Transformada Discreta do Seno (*Discrete Sine Transform*): transformada derivada da função seno com aplicações na compressão de dados e processamento de sinais.

DWT Transformada *Wavelet* Discreta (*Discrete Wavelet Transform*): transformadas não-trigonométricas baseadas em funções ortogonais e biortogonais e altamente usadas no processamento de sinais e compressão de dados.

DWT+SPIHT Método de compressão *lossy* apresentado neste trabalho.

ECG Eletrocardiograma: sinal biomédico que detecta impulsos elétricos gerados pelo coração.

EEG Eletroencefalograma: sinal biomédico que detecta impulsos elétricos gerados pelo cérebro.

EM Maximização de Expectativa (*Expectation Maximization*): técnica de aprendizado não-supervisionado baseada em IDBC.

EMG Eletromiogramas: sinal biomédico que detecta impulsos elétricos gerados por músculos voluntários.

EOG Eletro-oculograma: sinal biomédico que detecta variações na pressão ocular.

EZW Árvores Embarcadas de Transformadas *Wavelet* (*Embedded Zerotrees of Wavelet Transforms*): algoritmo de codificação de coeficientes derivados de *wavelets* que serviu de base para a criação do codificador SPIHT.

FIR Resposta de Impulso Finita (*Finite Impulse Response*): filtro que gera uma resposta de duração finita.

GC Código Gray (*Gray Code*): codificação de dados alternativa à representação binária normal de números inteiros não-negativos que pode ser usada previamente a uma etapa de codificação de entropia.

H.264 Compressor *lossy* de vídeos.

H.265 Compressor *lossy* de vídeos.

ICT Transformada Inteira do Cosseno (*Integer Cosine Transform*): versão da DCT que gera coeficientes inteiros e permite compressão *lossless*.

IDBC Agrupamento Iterativo Baseado em Distância (*Iterative Distance-Based Clustering*): algoritmos de agrupamento baseados nas distâncias entre as características das amostras.

IDHT Transformada Haar Discreta Inversa (*Inverse Discrete Haar Transform*): transformada inversa à DHT.

IDWT Transformada *Wavelet* Discreta Inversa (*Inverse Discrete Wavelet Transform*): transformada inversa à DWT.

IWT Transformada *Wavelet* de Coeficientes Inteiros (*Integer Wavelet Transform*): transformada *wavelet* que gera coeficientes inteiros e permite compressão *lossless*.

IWT+SPIHT Método de compressão *lossless* apresentado neste trabalho.

JPEG *Joint Photographics Experts Group*: método de codificação *lossy* de imagens.

JPEG 2000 Compressor *lossless* e *lossy* de imagens.

K Contexto máximo de um PPM.

KLT Transformada de Karhunen-Loève (*Karhunen-Loève Transform*): transformada que descorrelaciona perfeitamente os dados de entrada.

LWT Transformada *Wavelet* Preguiçosa (*Lazy Wavelet Transform*): operação de separação (*split*) de que compõe uma transformada *wavelet* usando o *Lifting Scheme*.

LZW *Lempel-Ziv-Welch*: compressor de dados *lossless* baseado em dicionários.

MIT/BIH PSGDB MIT/BIH *Polysomnographic Database*: banco de dados de sinais biomédicos do *Massachusetts Institute of Technology* e do Laboratório do Sono no Hospital Boston Beth Israel.

MSE Erro Médio Quadrático (*Mean Square Error*): medida de distorção objetiva largamente utilizada para a avaliação da reconstrução de sinais após compressão *lossy*.

MTF *Move To Front*: transformada aplicada comumente após uma BWT.

NPRD Percentual da Raiz Média Quadrática das Diferenças Normalizada (*Normalized Percent Root-mean-square Difference*): variação normalizada da métrica de erros PRD.

O_2S Saturação de Oxigênio (*Oxygen Saturation*): sinal biomédico que detecta variações na saturação de oxigênio.

P0 Preditor Linear de ordem 0.

P1 Preditor Linear de ordem 1.

P2 Preditor Linear de ordem 2.

P3 Preditor Linear de ordem 3.

PL Predição Linear: técnica de predição contextual de símbolos numa mensagem baseada na extrapolação de amostras.

PL+BitGol Método de compressão *lossless* apresentado neste trabalho.

PL+BPPM Método de compressão *lossless* apresentado neste trabalho.

PL+PPM-C Método de compressão *lossless* apresentado neste trabalho.

PPM Predição por Casamento Parcial (*Prediction by Partial Matching*): compressor de dados de propósitos gerais baseado na análise contextual de símbolos.

PPM* Variação do algoritmo do PPM.

PPM-C Variação do algoritmo do PPM.

PRD Percentual da Raiz Média Quadrática das Diferenças (*Percent Root-mean-square Difference*): métrica objetiva de erros usada comumente em sinais biomédicos.

PSG Sinal Polissonográfico: conjunto de sinais biomédicos comumente usado para a detecção de apneia noturna.

PTB *Physikalisch-Technische Bundesanstalt*: banco de dados diagnóstico de ECGs.

QRS Algumas ondas semi-periódicas que compõem sinais de ECG.

R Taxa de *Bits* (*Rate*): quantidade de *bits* utilizada para representar uma mensagem.

RC Razão de Compressão: medida comumente utilizada para avaliação de eficácia de compressão.

RLE Codificação por Comprimento de Sequência (*Run Length Encoding*): método de codificação de entropia baseado na repetição consecutiva de símbolos iguais comumente utilizado em conjunto com codificadores de Golomb.

RMSE Raiz do Erro Médio Quadrático (*Root Mean Square Error*): raiz quadrada da medida MSE.

RMSV Raiz do Valor Médio Quadrático (*Root Mean Square Value*): métrica usada para avaliar a variabilidade das amostras de sinais.

RSP Sinal de Respiração: sinal biomédico que detecta variações na respiração.

S Base *wavelet*.

S+P Base *wavelet*.

SAQ Quantização por Aproximações Sucessivas (*Successive Approximation Quantization*): algoritmos de quantização vetorial para coeficientes derivados de *wavelets* executados em conjunto com a codificação de entropia.

SOT Árvore de Orientação Espacial *Wavelet* (*Spatial Orientation Tree*): estrutura de dados usada nos algoritmos SPIHT e EZW para sinais espaciais.

SPIHT Particionamento de Conjuntos em Árvores Hierárquicas (*Set Partitioning in Hierarchical Trees*): algoritmo de codificação de entropia próprio para a utilização sobre coeficientes derivados de transformadas *wavelet*.

SV Volume Cardíaco (*Stroke Volume*): sinal biomédico que detecta variações na quantidade de sangue bombeada pelo coração.

TOT Árvore de Orientação Temporal *Wavelet* (*Temporal Orientation Tree*): estrutura de dados usada nos algoritmos SPIHT e EZW para sinais temporais.

TQC Transformação-Quantização-Codificação: paradigma de compressão *lossy* de sinais.

TS Base *wavelet*.

WDD Distorção Diagnóstica Ponderada (*Weighted Diagnostic Distortion*): métrica objetiva de erros específica para ECGs.

Sumário

Capítulo 1

Introdução

Os avanços nas tecnologias de transmissão de dados tornaram a capacidade de se comunicar um dos alicerces da sociedade atual. A rede mundial de computadores é um exemplo de uma fonte de informações democrática e completa, provendo oportunidades de negócios, cultura e entretenimento aos seus usuários. Porém, o acesso à Internet em banda larga, apesar dos recentes avanços tecnológicos, ainda continua a ser um serviço acessível a poucos, principalmente em países de terceiro mundo. Da mesma forma, uma grande quantidade de espaço em certos meios de armazenamento ou uma transmissão intensiva de dados podem ter custos elevados. Todos esses modos de armazenamento ou transmissão de dados são referidos como canais de dados pela Teoria de Informação. Uma forma de amenizar a sobrecarga sobre os canais de comunicação é diminuir a necessidade do espaço de armazenamento e/ou de transmissão de dados, mas preservando a informação que se deseja passar.

O conjunto de métodos utilizado numa mensagem visando a diminuição do espaço necessário para seu armazenamento é um processo chamado de compressão de dados, e esse tipo especial de processamento de dados é dividido em compressão com perdas e compressão sem perdas. Um processo de compressão e descompressão sem perdas de um sinal resulta num sinal reconstruído com exatamente as mesmas informações

do sinal original. Por sua vez, a compressão com perdas resulta numa aproximação do sinal original que pode ser satisfatoriamente fiel, dependendo da escolha das técnicas de compressão utilizadas e dos parâmetros passados para essas técnicas. Normalmente as técnicas de compressão com perdas resultam numa maior eficácia de compressão, embora não possam ser usadas em todos os casos. A Seção 3.1.1 detalha melhor os tipos de compressores de dados de acordo com a perda de informação.

Os termos compactação de dados e compressão de dados podem ser definidos de forma diferente, dependendo do material bibliográfico que seja seguido. Neste trabalho, esses termos serão tratados como equivalentes, indicando qualquer tipo de operação de codificação de informações que tenha como objetivo reduzir a quantidade de dados.

Dentre os vários tipos de sinais sobre os quais podem ser aplicadas técnicas de compressão com perdas estão os sinais biológicos. Existem diversos tipos desses sinais, cada um responsável por capturar uma ou mais características do ser vivo que está sendo analisado. A maioria dos sinais biológicos são usados para auxílio no diagnóstico médico ou biometria e, em ambos os casos, uma quantidade muito pequena de erros – ou até nenhuma quantidade – é aceitável após a compressão desses sinais.

A coleta de sinais biológicos através de polígrafos permite a realização de uma série de exames e experimentos fisiológicos e farmacológicos. Os polígrafos são analógicos por natureza e, tipicamente, registram em papel os dados coletados. Estes registros são posteriormente analisados por um profissional, que se encarrega de extrair parâmetros importantes para o experimento em questão, o que frequentemente constitui uma tarefa árdua, repetitiva e propensa a erros (Batista, 2002). Para eletroencefalogramas (EEGs), por exemplo, longos registros em papel, de até 55 e 60 metros de comprimento, são comuns (Pradhan e Dutt, 1994). Dessa forma, a digitalização desse tipo de exame é algo necessário para o armazenamento a longo prazo dos sinais de pacientes e para a garantia de fidelidade dos resultados em possíveis consultas futuras ao conteúdo dos sinais, já que o armazenamento em folhas de papel enfrenta tanto um problema de

espaço quanto de suscetibilidade a umidade, traças, ratos e outras intempéries físicas e químicas. A digitalização também pode tornar possível a criação de bancos de dados que centralizem o armazenamento de exames de hospitais de uma região, por exemplo, tornando os sinais acessíveis a qualquer estabelecimento de saúde de uma área.

A digitalização desse tipo de sinal, no entanto, não é computacionalmente barata. Um típico sinal de EEG normalmente usa 12 dígitos binários (*binary digits – bits*) por amostra e tem uma amostragem de 250 amostras por segundo (Ichimaru e Moody, 1999), o que resulta em 3000 *bits*/s – ou 375 *bytes*/s. Para armazenar uma hora de exames dessa natureza são necessários 1,29 MB, aproximadamente. Normalmente, além do EEG, também se deseja capturar sinais fisiológicos de outras naturezas e, unidos, esses exames baseados em polígrafos podem consumir até 9 MB/h. Os conjuntos de sinais capturados por polígrafos é denominado como o conjunto de sinais Polissonográficos (PSGs), os quais serão explicados em detalhes no Capítulo 2. A longo prazo, tais exames se tornam onerosos em demasia para se manter em computadores com poder de armazenamento comum. Para amenizar esse problema, técnicas de compressão de dados podem ser usadas nos sinais, conseguindo comprimi-los ao custo de pouca ou nenhuma eliminação de informação.

1.1 Trabalhos Relacionados

Diversos trabalhos descrevem métodos de compressão para eletrocardiogramas (ECGs) e EEGs, mas, em sua maioria, esses trabalhos são voltados para a codificação de um único tipo de exame, usando características intrínsecas aos seus respectivos sinais para obter melhores resultados de compactação. Nesta seção serão apresentados os principais trabalhos relacionados à compressão de sinais biomédicos encontrados na literatura. O único método autoadaptativo anterior ao presente trabalho que se propõe a abranger todos os sinais biomédicos é o apresentado por Oliveira et al. (2014), portanto, além

dele, apenas técnicas tratando da compressão de ECGs e EEGs serão descritas.

Batista (2002) propõe uma série de métodos semelhantes ao descrito neste trabalho, usando uma minimização de multiplicadores de Lagrange para encontrar os vetores ótimos de quantização e de zona-morta, mas considera apenas ECGs em seus testes. A principal diferença dos métodos apresentados por Batista (2002) para a estratégia de compressão com perdas de informação do presente trabalho está na etapa de codificação de entropia, que considerou principalmente codificadores de Golomb-Rice (Golomb, 1966) e codificadores aritméticos (Bell et al., 1990). Batista (2002) realizou testes com o MIT/BIH *Arrhythmia Database* (Moody e Mark, 2001), reportando razões de compressão médias entre 6,2 : 1 e 13,3 : 1, dependendo da distorção escolhida.

Zou e Gallagher (1993) usaram uma combinação de transformadas trigonométricas e *wavelets* para codificar áreas diferentes de sinais de ECG. As partes do sinal que contêm complexos QRS foram compactadas usando *wavelets* e as áreas correspondentes às ondas P, T e U foram comprimidas com a quantização de coeficientes gerados por transformações baseadas na função cosseno. Os sinais estudados nesse trabalho foram divididos em blocos de 512 amostras, transformados e, posteriormente, subamostrados, de forma que apenas 91 amostras eram mantidas. Dessa forma, o trabalho conseguia diminuições nos arquivos da ordem de 5,6 : 1 com uma boa qualidade de reconstrução.

Ranjeet et al. (2013) usam uma estratégia de Corte e Alinhamento (*Cut and Align* – CAB) para seccionar ECGs e reorganizá-los para executar transformadas *wavelet* bidimensionais com blocos de tamanho 180×20. Os coeficientes transformados são passados para um codificador de Huffman (Huffman, 1952), atingindo ganhos de 65% em eficácia de compressão e uma correlação de 0,999 entre o registro original e o comprimido. Os testes dessa técnica de compressão quase sem perdas foram novamente realizados com o MIT/BIH *Arrhythmia Database* (Moody e Mark, 2001).

O método descrito por Mukhopadhyay et al. (2011) usa uma técnica de diferenciação para detectar todos os picos de ondas R em ECGs, permitindo que o algoritmo aplique

métodos de predição linear nessas regiões, favorecendo a compressão. As áreas próximas a ondas R são passadas para um codificador sem perdas de informação, enquanto as áreas entre ondas R são codificadas com perdas de informação. Mukhopadhyay et al. (2011) usaram o banco de dados diagnóstico de ECGs *Physikalisch-Technische Bundesanstalt* (PTB) (Oeff et al., 2012) em seus testes, reportando ganhos de espaço de armazenamento na ordem de 23,10 : 1 com métricas de distorção baseadas no Percentual da Raiz Média Quadrática das Diferenças (*Percent Root-mean-square Difference* – PRD) de 7,55%.

O trabalho apresentado por Lai et al. (2013) usa variações da transformada discreta do cosseno para a compressão de ECGs. Lai et al. (2013) reportam razões de compressão com perdas de informação de 5,25 : 1 usando distorções baseadas na PRD de 0,19% em arquivos do MIT/BIH *Arrhythmia Database* (Moody e Mark, 2001).

Antoniol e Tonella (1997) descrevem vários métodos de compressão sem perdas ou quase sem perdas de EEGs, variando principalmente a etapa de predição dos sinais. Para os testes do método, os autores usaram um banco de dados próprio capturado no S. Chiara Hospital. Bons resultados são obtidos com o uso de um preditor linear alimentando um codificador de entropia, o que resultou em razões de compressão de aproximadamente 2,51 : 1. Os melhores resultados foram obtidos ao se usar uma quantização vetorial seguida de um codificador de Huffman (Huffman, 1952), chegando a eliminar 62% do sinal original.

Memon et al. (1999) comparam diversas técnicas de compressão sem perdas existentes para a compressão de EEGs. Os melhores resultados são obtidos ao se modelar o EEG usando um processo autorregressivo, o que resultou em arquivos comprimidos com tamanhos próximos a $1/3$ dos originais.

Srinivasan et al. (2013) propõem um método para a codificação de EEGs com canais múltiplos baseado em transformadas *wavelet* bidimensionais e tridimensionais. Essa estratégia permite que o algoritmo explore tanto as similaridades dos canais individuais

quanto as similaridades entre os diversos canais do registro de EEG. Esse trabalho reporta razões de compressão de 6,63 : 1 com distorções de 9,21% baseadas na PRD.

Oliveira et al. (2014) descrevem uma técnica de compressão com perdas de informação para PSGs, baseando-se na minimização de multiplicadores de Lagrange para a otimização de uma quantização vetorial de coeficientes no domínio da frequência. Os coeficientes transformados e quantizados são passados em uma ordem específica dependendo do índice para um codificador preditivo contextual. O método foi testado usando o MIT/BIH *Polysomnographic Database* (Ichimaru e Moody, 1999) e reportou resultados de razão de compressão entre 2,16 : 1 e 67,48 : 1, dependendo do tipo de registro biomédico e da quantidade de distorção selecionada.

1.2 Objetivos

Este trabalho tem por objetivo desenvolver métodos inovadores de compressão de sinais biomédicos e testar a eficácia de métodos de compressão comumente usados para a compactação de outros tipos de dados – como imagens e vídeos – no contexto dos sinais biomédicos.

1.2.1 Objetivos Gerais

O principal objetivo desse trabalho é avançar o estado da arte da área de compressão de sinais biomédicos. Para isso, serão realizados testes de compressão com perdas e sem perdas de informação usando tanto técnicas de compressão de sinais biomédicos quanto métodos originários de outras áreas associadas à Teoria da Informação. Dessa forma, serão analisadas as eficácias de compactação nos resultados obtidos após a execução desses métodos, possibilitando uma melhor aplicação dos mesmos em futuros trabalhos.

1.2.2 Objetivos Específicos

Visando obter uma análise mais completa sobre os métodos de compactação que apresentam os melhores resultados de compressão de sinais biomédicos unidimensionais, diversos tipos de combinação de técnicas serão avaliados e testados. Primeiramente, vários tipos de transformadas que aumentem a suscetibilidade dos sinais à compressão serão avaliadas. Entre as transformadas que se pretende avaliar estão:

1. Transformadas trigonométricas (Britanak et al., 2006);
2. Transformadas baseadas em *wavelets* (Calderbank et al., 1998).

Em uma segunda etapa, diversos métodos de codificação de entropia serão executados tanto como um pós-processamento para as transformadas quanto sozinhos para avaliar o desempenho de técnicas de compressão clássicas (de propósitos gerais) nos sinais em questão. Nesse trabalho, o foco será direcionado para codificadores de entropia baseado na modelagem estatística direta dos símbolos da mensagem. Os métodos clássicos de codificação de entropia que serão avaliados são:

1. Codificadores de Golomb (Golomb, 1966);
2. O algoritmo de Predição por Casamento Parcial (Cleary e Witten, 1984);
3. O algoritmo de Predição por Casamento Parcial Binário (Brasileiro e Cavalcanti, 2012);
4. Codificadores de Particionamento de Conjuntos em Árvores Hierárquicas (Said e Pearlman, 1996).

Por último, pretende-se testar métodos de pré e pós-processamento que auxiliem na compressão de sinais biomédicos como:

1. Preditores lineares como pré-processamento para algoritmos de codificação de entropia;

2. Estratégias de quantização;

3. Minimização Lagrangiana para a otimização de parâmetros de compressão, de acordo com Ratnakar e Livny (1995) e Ratnakar e Livny (1996);

4. Técnicas diferenciais, como a Modulação por Código de Pulso Diferencial (*Differential Pulse-Code Modulation* – DPCM) (Meares, 1974).

1.3 Conteúdo do Trabalho

O restante deste trabalho está organizado da seguinte forma: o Capítulo 2 introduz os sinais biológicos estudados e um banco de dados de sinais biomédicos; o Capítulo 3 introduz vários conceitos importantes da Teoria da Informação e do Processamento de Sinais; no Capítulo 4 estão descritas técnicas do estado da arte da Compressão de Dados; no Capítulo 5 estão descritas as técnicas apresentadas nos capítulos anteriores que foram usadas na prática para a compressão de sinais biomédicos; o Capítulo 6 apresenta os resultados de compressão e reconstrução dos PSGs obtidos após a aplicação dos codificadores propostos neste trabalho; e, por fim, o Capítulo 7 apresenta as discussões e conclusões que foram tiradas dos resultados e contém as considerações finais do trabalho. O Apêndice A demonstra distribuições estatísticas de sinais biomédicos após transformações trigonométricas. O Apêndice B mostra as reconstruções de vários sinais biomédicos após passarem por um processo de decodificação posterior a uma compressão com perdas de informação. O Apêndice C contém resultados de comparação complementares aos do Capítulo 6. Por último, o Anexo D destaca algumas funções de base para transformadas de domínio usadas neste trabalho.

Capítulo 2

Sinais Polissonográficos

Dentre os vários sinais biológicos usados para diagnóstico médico, os PSGs englobam alguns dos mais comuns em um único conjunto de exames. Os PSGs envolvem diversos tipos de sinais fisiológicos, dependendo do que se deseja avaliar da condição física do paciente. Na maioria dos casos esses sinais são capturados durante os horários de sono, pois, em geral, são usados no diagnóstico e acompanhamento de casos de apneia. Assim, esses exames necessitam muitas vezes de uma instalação própria dentro de uma clínica para a coleta dos dados.

Šušmáková (2004) define a polissonografia como um método de avaliação do sono humano composto de eletroencefalogramas, eletro-oculogramas (EOGs) e eletromiogramas (EMGs). Essa definição, porém, se foca nos exames polissonográficos que visam o estudo dos estados do sono, não levando em conta, por exemplo, a avaliação das condições cardíacas do paciente. Dessa forma, uma grande variedade de tipos de PSGs, além dos três apresentados por Šušmáková (2004), pode ser utilizada na prática. Esses tipos diferentes de sinais num único exame são chamados de canais de dados no contexto do Processamento de Sinais.

Esses exames são realizados em larga escala por clínicas no mundo inteiro, muitas vezes para diagnosticar condições que não podem ser vistas durante as horas em que o

paciente está acordado, como a apneia noturna. Esta seção apresentará os principais PSGs usados para diagnóstico médico, explanando as principais características de cada tipo de registro.

2.1 Eletrocardiograma

Muitas anomalias cardíacas podem ser diagnosticadas através de ECGs, o que faz desse exame um dos mais comuns dentre os presentes nos PSGs.

2.1.1 Morfologia de um ECG

O ECG é normalmente composto por uma onda P; um complexo QRS, formado pelas ondas Q, R e S; e uma onda T. Pode haver ainda, após a onda T, uma onda de baixa amplitude, denominada onda U. A periodicidade do ECG é uma peculiaridade que pode trazer complicações na compressão desses sinais. O complexo QRS tem uma distribuição estatística de símbolos bastante diferente do resto do sinal, portanto, deve ser manipulado de forma especial. Uma primeira ideia é simplesmente separar o conteúdo contendo QRSs do conteúdo do sinal que não contém QRSs, executando duas técnicas de compressão paralelamente. A Figura 2.1 apresenta um ciclo típico de um traçado de ECG, juntamente com suas partes constituintes. Já a Figura 2.2 exemplifica um sinal de ECG real.

2.2 Eletroencefalograma

Um EEG é um registro da atividade cerebral e reflete o estado funcional do cérebro. Na área de investigação neurofisiológica experimental e clínica, o EEG é considerado um fenômeno resultante dos processos metabólicos integrados do cérebro. O EEG reflete as atividades da estrutura cerebral interna e, particularmente, do córtex cerebral abaixo da superfície do couro cabeludo (Pradhan e Dutt, 1994).

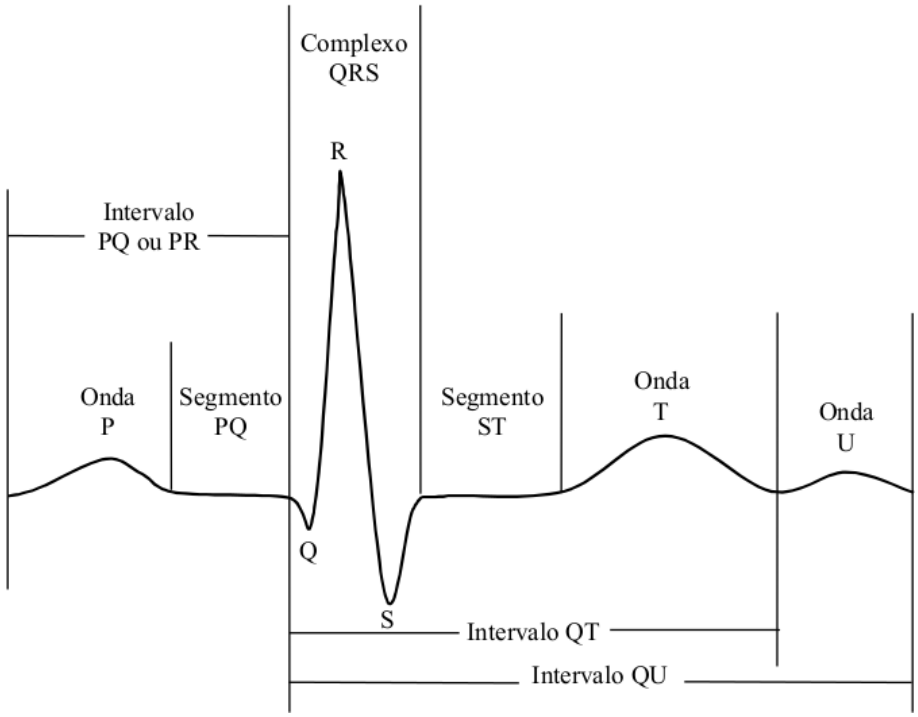


Figura 2.1: Morfologia de um ECG. Fonte: Batista (2002).

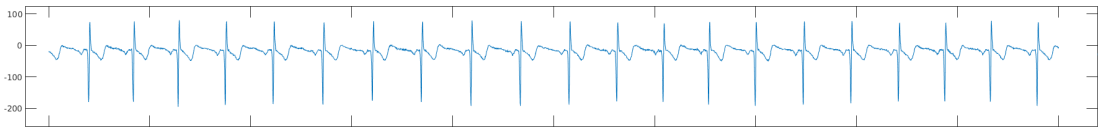


Figura 2.2: Seção de 20 segundos de um sinal de ECG.

EEGs, assim como o ECGs, contêm algumas características que os diferenciam dos outros PSGs, principalmente na forma de lidar com os dados desse tipo de exame. Em alguns aspectos os EEGs se comportam visualmente de forma aleatória e, se não forem manipulados da forma correta, podem apresentar uma perda de qualidade que comprometa um posterior exame do sinal. A Figura 2.3 demonstra o aspecto de um EEG típico.

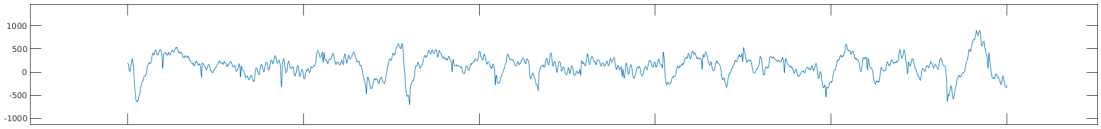


Figura 2.3: Seção de 20 segundos de um sinal de EEG.

2.3 Pressão Sanguínea

Além dos ECGs e EEGs, os PSGs contam muitas vezes com sinais de pressão sanguínea (*Blood Pressure* – BP) invasiva ou não invasiva. Assim como os ECGs, os BPs são periódicos, contendo uma onda que se assemelha a uma onda R do complexo QRS de um ECG. Esse desnível periódico no sinal acontece a cada batimento cardíaco, com um atraso temporal – se comparado com a onda R – já que as ondas de pressão geradas pelos batimentos cardíacos levam um certo tempo – determinado pela velocidade do som no sangue – para chegar ao local em que o BP está sendo amostrado. A Figura 2.4 mostra um exemplo de BP.

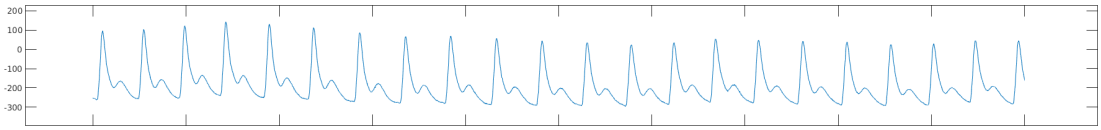


Figura 2.4: Seção de 20 segundos de um sinal de BP.

2.4 Respiração

Os sinais de respiração (RSPs), assim como os ECGs e BPs, têm períodos notadamente regulares. Isso se dá porque geralmente os PSGs são colhidos durante o sono dos pacientes, quando a respiração é mais constante. Esses exames, porém, podem sofrer alterações de acordo com o estado de saúde das pessoas examinadas. De acordo com Ichimaru e Moody (1999), os RSPs são monitorados usando termístores nasais e um pletismógrafo indutivo. Como os PSGs fazem parte muitas vezes de exames ligados ao diagnóstico de apneia noturna nos pacientes, os RSPs podem sofrer uma pausa brusca

durante alguns segundos. Essa pausa também interfere nos outros sinais, embora não tão significativamente quanto nos RSPs. Um sinal típico de respiração é visto na Figura 2.5.

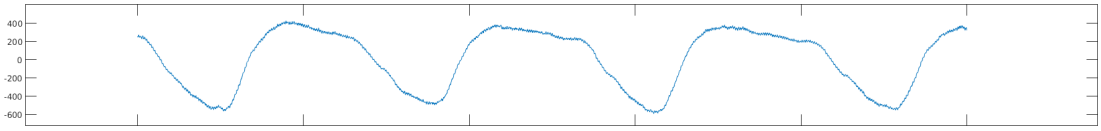


Figura 2.5: Seção de 20 segundos de um sinal de RSP.

2.5 Eletromiograma

Muitos PSGs contêm também EMGs, que são colhidos de acordo com a extensão e distensão de músculos voluntários. Esse tipo de sinal é quase imprevisível, já que a intensidade e frequência do movimento dos músculos voluntários são determinadas pelo estado do sono do paciente. Essa característica torna os EMGs um dos sinais mais entrópicos entre os PSGs. Um EMG pode ser visto na Figura 2.6.

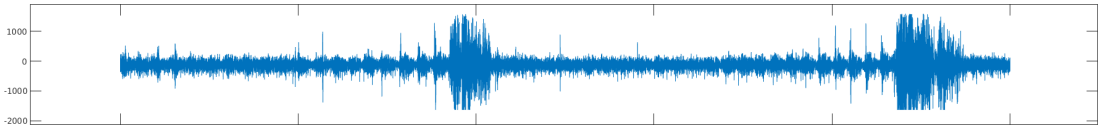


Figura 2.6: Seção de 20 segundos de um sinal de EMG.

2.6 Eletro-oculograma

Outros sinais muitas vezes presentes em exames com PSGs são os EOGs. Esse tipo de registro mede os impulsos elétricos da retina e pode detectar doenças oculares relacionadas ao envelhecimento, como o glaucoma. Um EOG pode ser visto na Figura 2.7.

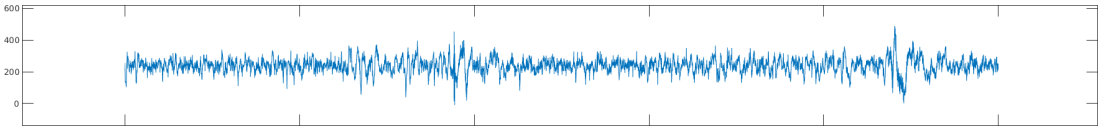


Figura 2.7: Seção de 20 segundos de um sinal de EOG.

2.7 Volume Cardíaco

Canais de dados contendo sinais de volume cardíaco (*Stroke Volume* – SV) também estão presentes em muitos exames envolvendo PSGs. Esses sinais detectam a quantidade de sangue bombeada pelo coração. Um SV comum pode ser visto na Figura 2.8.

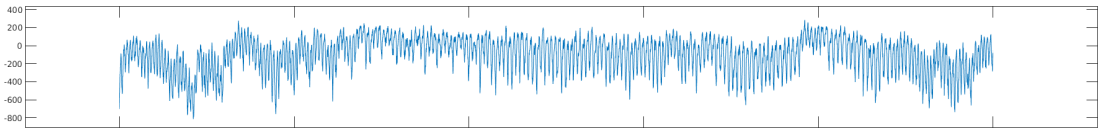


Figura 2.8: Seção de 20 segundos de um sinal de SV.

2.8 Saturação de Oxigênio

Sinais de saturação de oxigênio (*Oxygen Saturation* – O_2S) medem a quantidade de oxigênio que é dissolvida ou transportada pelo sangue de um paciente. Esse valor é proporcional à quantidade de hemoglobina presente no sangue. A medição desse tipo de sinal muitas vezes presente nos PSGs é feita usando um oxímetro de pulso ou de dedo. A Figura 2.9 mostra o aspecto de um O_2S típico.

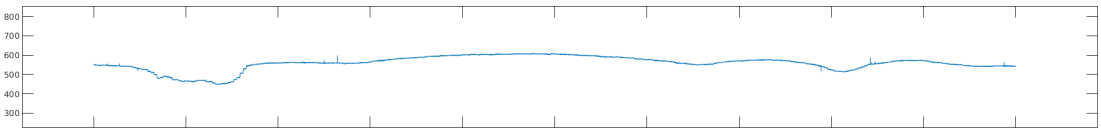


Figura 2.9: Seção de 20 segundos de um sinal de O_2S.

2.9 Banco de Dados da Physionet

Moody e Mark (2001) desenvolveram um banco de dados de arritmia cardíaca do MIT/BIH para CD-ROM. Esse banco de dados tornou-se padrão para a avaliação de algoritmos computadorizados de classificação de arritmias. Porém, a literatura da área de sinais biomédicos apresentava uma carência de bancos de dados de PSGs, o que levou

Ichimaru e Moody (1999) a criarem o banco de dados de sinais fisiológicos denominado de MIT/BIH *Polysomnographic Database*, referido a partir desse ponto como MIT/BIH PSGDB.

Os pacientes foram monitorados no Laboratório do Sono no Hospital Boston Beth Israel para a avaliação de síndrome de apneia obstrutiva crônica e para testar os efeitos da pressão constante positiva nas vias aéreas, uma intervenção terapêutica comum que normalmente previne ou reduz substancialmente a obstrução das vias aéreas nesses pacientes (Moody et al., 2001).

Segundo Ichimaru e Moody (1999), o tempo de captura dos sinais varia entre 2h e 7h, dependendo do paciente. Foram capturados ECGs, EEGs, EOGs, EMGs do músculo do queixo, BPs invasivas, O_2Ss, RSPs e SVs. Esses sinais fisiológicos foram digitalizados em um intervalo de amostragem de 250 Hz e com 12 *bits*/amostra. O conjunto completo dos registros compreende 18 ECGs, 18 EEGs, 5 EOGs, 5 EMGs, 18 BPs, 27 RSPs, 5 SVs e 5 O_2Ss, totalizando 101 canais de dados diferentes. Os registros biomédicos no MIT/BIH PSGDB estão organizados em 18 arquivos diferentes: slp01a, slp01b, slp02a, slp02b, slp03, slp04, slp14, slp16, slp32, slp37, slp41, slp45, slp48, slp59, slp60, slp61, slp66 e slp67x.

Além dos arquivos com as amostras, o banco de dados possui arquivos de cabeçalho contendo metadados sobre os sinais, arquivos de anotação sobre os batimentos cardíacos e arquivos com informações sobre os estágios de sono dos pacientes. A Figura 2.10 apresenta traçados de um dos sinais no banco de dados de PSGs.

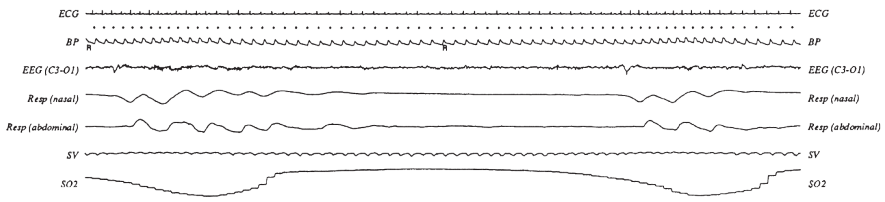


Figura 2.10: Sete sinais de um arquivo do banco de dados de PSGs do MIT/BIH PSGDB. Fonte: Ichimaru e Moody (1999).

As Tabelas 2.1, 2.2 e 2.3 contêm as organizações dos arquivos do MIT/BIH PSGDB. As siglas RSP(sm), RSP(ns), RSP(ab), RSP(pt) e EOG(dr) significam respectivamente: RSP(soma), RSP(nasal), RSP(abdominal), RSP(peitoral) e EOG(olho direito); e servem para especificar os pontos de amostragem específicos de cada canal de dados no corpo do paciente, quando necessário. O mesmo acontece para os EEGs, que podem ter as variações: EEG(C4-A1), EEG(O2-A1) e EEG(C3-O1). Os locais de captura dos EEGs podem ser vistos na Figura 2.11.

Tabela 2.1: Registros do MIT/BIH PSGDB que contêm 4 canais de dados.

Registro	**Canais**			
slp01a	ECG	BP	EEG(C4-A1)	RSP(sm)
slp01b	ECG	BP	EEG(C4-A1)	RSP(sm)
slp02a	ECG	BP	EEG(O2-A1)	RSP(ns)
slp02b	ECG	BP	EEG(O2-A1)	RSP(ns)
slp03	ECG	BP	EEG(C3-O1)	RSP(ns)
slp04	ECG	BP	EEG(C3-O1)	RSP(ns)
slp14	ECG	BP	EEG(C3-O1)	RSP(ns)
slp16	ECG	BP	EEG(C3-O1)	RSP(ns)

Tabela 2.2: Registro do MIT/BIH PSGDB que contém 6 canais de dados.

Registro	**Canais**					
slp61	ECG	BP	EEG(C3-O1)	RSP(ab)	SV	O_2S

Tabela 2.3: Registros do MIT/BIH PSGDB que contêm 7 canais de dados.

Registro	**Canais**						
slp32	ECG	BP	EEG(C4-A1)	RSP(ns)	RSP(pt)	EOG	EMG
slp37	ECG	BP	EEG(C4-A1)	RSP(ns)	RSP(ab)	EOG(dr)	EMG
slp41	ECG	BP	EEG(C4-A1)	RSP(ns)	RSP(ab)	EOG(dr)	EMG
slp45	ECG	BP	EEG(C3-O1)	RSP(ns)	RSP(ab)	EOG(dr)	EMG
slp48	ECG	BP	EEG(C3-O1)	RSP(ns)	RSP(pt)	EOG(dr)	EMG
slp59	ECG	BP	EEG(C3-O1)	RSP(ns)	RSP(ab)	SV	O_2S
slp60	ECG	BP	EEG(C3-O1)	RSP(ab)	RSP(ns)	SV	O_2S
slp66	ECG	BP	EEG(C3-O1)	RSP(ns)	RSP(ab)	SV	O_2S
slp67x	ECG	BP	EEG(C3-O1)	RSP(ns)	RSP(pt)	SV	O_2S

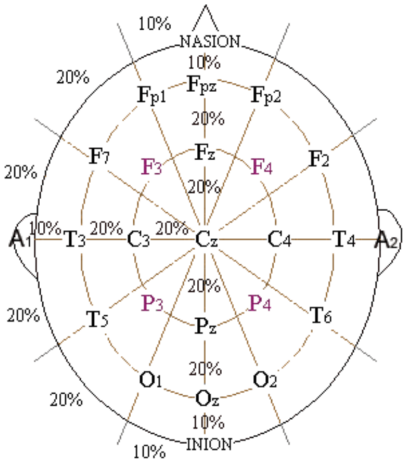


Figura 2.11: Locais de captura das variações dos EEGs. Fonte: Šušmáková (2004).

Mais informações sobre o banco de dados, incluindo os parâmetros de digitalização de cada tipo de sinal, podem ser encontradas em Ichimaru e Moody (1999) e no *website* da PhysioNet (`http://physionet.org/cgi-bin/atm/ATM`).

2.10 Conclusão do Capítulo

Esse capítulo apresentou os oito tipos de sinais e o banco que foram usados para os testes do presente trabalho. Foram descritas as características básicas dos sinais biomédicos em questão, bem como mostrados os traçados de exemplos desses sinais. Dando continuidade, os Capítulos 3 e 4 descreverão a base teórica necessária para o entendimento dos métodos de compressão implementados.

Capítulo 3

Conceitos da Teoria da Informação

Dados são comprimidos quando se reduz sua redundância, mas isso também os torna menos confiáveis, mais suscetíveis a erros. O aumento da integridade dos dados, por outro lado, é feito pela adição de *bits* de checagem e *bits* de paridade, um processo que aumenta o tamanho dos dados, aumentando a redundância. Compressão de dados e confiabilidade de dados são, portanto, opostos (Salomon, 2006). Tanto os processos de compressão de dados quanto os processos de aumento na confiabilidade de dados são formas de codificação. Essas formas de representação de informação podem ser usadas com vários propósitos diferentes, o que requer processos de codificação distintos. Por exemplo, um processo bastante específico de codificação de dados é a criptografia, na qual se deseja mudar a representação de uma certa fonte de informação para que apenas os destinos escolhidos consigam interpretar os dados com sucesso.

Já a compressão de dados foca na criação de uma codificação específica que diminua a quantidade de dados que se deseja enviar ou armazenar através da redução da redundância na informação. O termo descorrelação significa que os valores pós-processados são independentes uns dos outros. Como resultado, esses valores podem ser codificados

independentemente, o que torna o processo de criação de um modelo estatístico mais simples (Salomon, 2006). Um esquema genérico para um compressor de dados pode ser visto na Figura 3.1. O esquema define um compressor qualquer que opera sobre um sinal x (com n *bits*), resultando na mensagem $\bar{x}$ (com m *bits*), a qual é transmitida por um canal até chegar ao descompressor. Esse descompressor realiza a operação de decodificação do sinal, recuperando um sinal aproximado $\tilde{x}$ com os n *bits* originais.

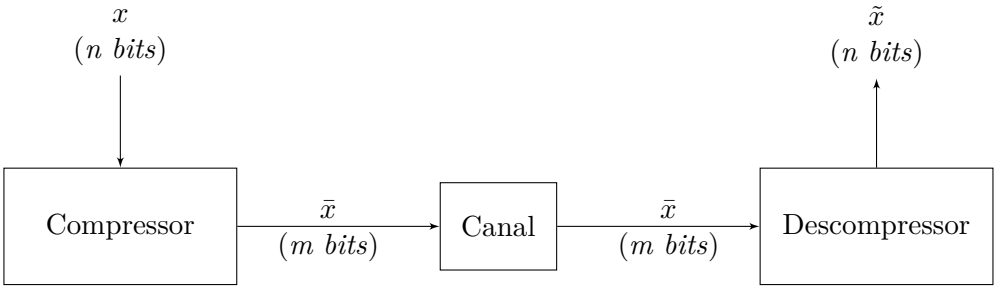


Figura 3.1: Um conjunto compressor/descompressor ligado por um canal de transmissão. Fontes: Batista (2002) e Shannon (1948).

Uma medida usada frequentemente na literatura para avaliar o grau de compressão obtido é a Razão de Compressão (RC), dada pela Equação 3.1 de acordo com as quantidades de *bits* n e m definidas na Figura 3.1:

$$RC = \frac{n}{m}. \tag{3.1}$$

Como define Batista (2002), para uma dada mensagem, um compressor não parametrizado produz invariavelmente a mesma RC e distorção. Os compressores parametrizados, por outro lado, oferecem um maior grau de controle sobre os resultados, uma vez que a RC e a distorção podem ser ajustadas pelos parâmetros passados para o método de compressão. Em muitos compressores que descartam informação da mensagem original deve haver um balanceamento entre a quantidade de distorção aceitável no sinal reconstruído e a quantidade de *bits* que o sinal comprimido ocupará. No caso de um compressor parametrizado, esse balanceamento pode ser atingido por uma busca

no espaço de parâmetros possíveis do compressor, procurando atingir a melhor taxa de *bits* para uma certa distorção fixa ou a menor distorção possível para uma taxa de *bits* fixa. Essa taxa de *bits* e distorção serão representadas respectivamente por $R(\phi)$ e $D(\phi)$, em que ϕ representa o conjunto de todas as combinações lineares dos parâmetros que podem ser modificados no compressor. Tanto no caso da otimização feita pela taxa de *bits* quanto no da otimização pela distorção, o problema pode ser formulado como um problema genérico de minimização com restrição para um compressor com perdas parametrizável seguindo a Formulação 3.1 a seguir:

Formulação 3.1 : Dada uma distorção D_t, encontrar uma instância de parâmetros do compressor que minimize R, com $D = D_t$.

Devido às diferentes naturezas e comportamentos dos diversos tipos de sinais que são estudados, várias técnicas de compressão distintas são necessárias para comprimi-los de forma eficiente. As técnicas de compressão estão divididas, de acordo com a revertibilidade da codificação, em duas categorias: técnicas de compressão com perdas de informação e técnicas de compressão sem perdas de informação. Outras classificações importantes sobre os compressores também existem, como a característica de serem não adaptativos, semiadaptativos ou adaptativos; ou ainda se são parametrizáveis ou não parametrizáveis. A Seção 3.1 tratará de explicar as particularidades de cada uma dessas classificações e como elas podem se comportar melhor atuando sobre certos tipos de mensagem de entrada. Em seguida, a Seção 3.2 introduzirá várias medidas de distorção usadas comumente para medir objetivamente a quantidade de erros gerados por um processo de compressão com perdas de informação. Posteriormente, a Seção 3.3 introduzirá os conceitos de informação e entropia, descrevendo seus cálculos e implicações para a compressão de sinais. Por último, a Seção 3.4 introduzirá a fundamentação matemática por trás da Teoria Taxa-Distorção e, por fim, descreverá um método de otimização baseado na Minimização Lagrangiana aplicado à compressão de dados.

3.1 Tipos de Compressores de Dados

Dada a vasta gama de tipos de sinais de entrada que podem ser submetidos a uma etapa de compressão, é natural que não exista um compressor ótimo para todos. Dessa forma, diversas classificações de tipos de algoritmos de compressão surgiram, cada uma com diferentes propriedades. A Seção 3.1.1 tratará da diferença entre compressores sem perdas de informação (*lossless*) e compressores com perda de informação (*lossy*).

3.1.1 Compressores *Lossless* e *Lossy*

As técnicas de compressão de dados sem perdas são limitadas pela entropia do sinal e, geralmente, são usadas nos casos nos quais apenas um único *bit* de diferença do sinal original interfere de forma grotesca na sua interpretação. Por exemplo, deseja-se comprimir um arquivo executável contendo o código de máquina de um processador e os valores binários 0b01001010 e 0b01001011 (em valores decimais, 74 e 75) representam duas instruções responsáveis, respectivamente, pela soma e multiplicação de dois números. Caso seja passado um compressor *lossy* nesse arquivo, por mais que as perdas sejam tão pequenas quanto um único *bit* no sinal inteiro, uma operação de soma pode ser decodificada erroneamente como uma operação de multiplicação, o que tem uma grande chance de mudar o funcionamento do programa inteiro e/ou causar um erro na execução. Grande parte dos sinais criados artificialmente ou não capturados diretamente da natureza – por exemplo, arquivos executáveis e arquivos de texto – tendem a ter uma distribuição estatística de símbolos totalmente diferente dos sinais naturais e uma dependência maior de cada *bit* que contém informação.

As técnicas de compressão de dados *lossy* são melhor empregadas em sinais capturados por sensores diretamente de um meio natural como por exemplo, imagens, áudio, vídeo e sinais biológicos. Outra característica importante desses sinais é que se consegue perceber os erros neles presentes apenas até um certo limiar delimitado pelos

sentidos humanos. Aproveitando-se dessa característica, são criados compressores de dados que conseguem, além de eliminar a correlação entre os símbolos, desfazer-se da parte da informação não perceptível pelo ser humano, resultando em RCs maiores que compressores *lossless*. Muitos compressores *lossy* contêm mecanismos de controle dos níveis de distorção, o que permite que o usuário escolha a quantidade de erros máxima que deve estar presente no sinal reconstruído.

Grande parte dos compressores *lossy* contêm ao menos um módulo que usa técnicas de compressão *lossless*, o qual geralmente é responsável pela codificação final do sinal após a introdução de perdas por algum outro módulo. Se implementados de forma correta, os compressores *lossy* podem chavear facilmente a etapa final de codificação *lossless* entre várias técnicas diferentes.

3.1.2 Classificação de Acordo com a Adaptabilidade do Modelo

Além de serem classificados de acordo com a perda ou não de informação, os compressores de dados podem ser definidos de acordo com sua adaptabilidade às mensagens emitidas pela fonte de informação. Compressores não adaptativos não levam em conta os símbolos compactados previamente para mudar as probabilidades de símbolos futuros, de forma que a compressão fica amarrada a uma determinada distribuição estatística predefinida. Esse tipo de compactador é muito utilizado como uma subetapa de um outro método de compressão, principalmente quando um dos principais requisitos do sistema é a baixa complexidade computacional, já que essa categoria de compactadores é a mais rápida. Compressores de vídeo em tempo real (ITU-T264, 2002) usam frequentemente métodos não adaptativos nos quais a compressão se resume apenas a um conjunto de tabelas de tradução de símbolos chaveadas pelo compressor em tempo de execução, representando uma intersecção entre um método de compressão não adaptativo e adaptativo. Essa categoria de compressores normalmente usa uma série de códigos fixos previamente calculados para codificar um conjunto de combinações de

símbolos, muitas vezes se limitando apenas a uma etapa de codificação de entropia.

Compressores adaptativos são os que levam em conta a distribuição da mensagem inteira desde o início da compressão, precisando ler o sinal de entrada duas vezes durante sua execução. Isso normalmente resulta num ganho pequeno comparado com métodos semiadaptativos. O lado negativo dos modelos adaptativos de compressão é que, para a descompressão, faz-se necessária a codificação do modelo estatístico da mensagem além da codificação da própria mensagem, podendo resultar num aumento significativo da informação que precisa ser transmitida.

Os métodos semiadaptativos são a categoria mais moderna de compressores, eliminando o problema dos compressores adaptativos de salvar a tabela de símbolos para a etapa de descompressão, bem como desfazendo-se da necessidade de ler duas vezes os símbolos da mensagem. O surgimento dessa categoria de compressores possibilitou a criação de métodos contextuais, considerados mais poderosos em termos de performance de compressão. Compressores da família do Preditor por Casamento Parcial (Cleary e Witten, 1984) e do *Lempel-Ziv-Welch* (LZW) (Welch, 1984) treinam seus modelos estatísticos e dicionários muito rapidamente, de forma que as versões semiadaptativas apresentaram RCs muito próximas às adaptativas, mesmo sem levar em conta a tabela de símbolos daquelas.

3.2 Medidas de Distorção

As medidas de distorção são usadas em conjunto com as técnicas de compressão de dados *lossy*, de modo que as perdas de informação sejam objetivamente mensuráveis. Tais medidas podem ser integradas a compressores com perdas com a finalidade de achar parâmetros ótimos de utilização desses compactadores, minimizando a perda de informação e preservando a qualidade do sinal após a descompressão.

Sejam o sinal original S_n e o sinal reconstruído após uma compressão *lossy* S'_n –

$n = 0, 1, ..., N-1$. Dentre as diversas medidas que permitem aferir a distorção entre S_n e S'_n, uma das mais populares é o Erro Médio Quadrático (*Mean Square Error* – MSE):

$$\text{MSE} = \frac{1}{N} \sum_{n=0}^{N-1} (S_n - {S'}_n)^2. \tag{3.2}$$

Uma medida de distorção diretamente ligada ao MSE é a Raiz do Erro Médio Quadrático (*Root Mean Square Error* – RMSE), que representa apenas uma operação de radiciação na base 2 sobre o resultado da MSE:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} (S_n - {S'}_n)^2}. \tag{3.3}$$

Não é possível medir com precisão a quantidade de erros na reconstrução de um sinal sem saber também como se comportam as amostras originais desse sinal. A medida conhecida como Raiz do Valor Médio Quadrático (*Root Mean Square Value* – RMSV) é uma métrica objetiva para a quantidade de energia nas amostras de um sinal, dada por:

$$\text{RMSV} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} (S_n)^2}. \tag{3.4}$$

Como ressalta Abenstein e Tompkins (1982), a distorção na reconstrução dos ECGs é frequentemente medida pela PRD:

$$\text{PRD} = \frac{\text{RMSE}}{\text{RMSV}} \times 100\% = \sqrt{\frac{\frac{1}{N} \sum_{n=0}^{N-1} (S_n - {S'}_n)^2}{\frac{1}{N} \sum_{n=0}^{N-1} (S_n)^2}} \times 100\%. \tag{3.5}$$

Entretanto, a PRD é uma medida muito sensível ao nível médio do sinal original (Zigel et al., 2000), portanto uma segunda definição para o seu cálculo é encontrada em alguns artigos. Neste trabalho, essa medida será chamada de PRD Normalizada (*Normalized PRD* – NPRD) e é calculada pela fórmula:

$$\text{NPRD} = \frac{\text{RMSE}}{\sigma} \times 100\% = \sqrt{\frac{\frac{1}{N}\sum_{n=0}^{N-1}\left(S_n - S'_n\right)^2}{\frac{1}{N}\sum_{n=0}^{N-1}\left(S_n - \overline{S}\right)^2}} \times 100\%. \tag{3.6}$$

Nesse caso σ é o desvio padrão do sinal original e $\overline{S}$ é a média desse sinal. Um sinal com uma linha de base muito alta interfere no cálculo da PRD, já que o valor do RMSV cresce com relação ao valor das amostras em ordem quadrática, possibilitando níveis mais altos de RMSE para uma mesma PRD. A NPRD, portanto, é mais confiável para medir a quantidade de erros na reconstrução de sinais biológicos, já que ela desconsidera a linha de base presente nas amostras.

A PRD é um valor normalizado que indica o erro entre dois sinais (Abenstein e Tompkins, 1982), porém, segundo Zigel e Cohen (1999), apesar de largamente utilizadas como medidas de distorção, a PRD e a NPRD não indicam com precisão a qualidade da reconstrução de um sinal. Uma alternativa mais confiável é a Distorção Diagnóstica Ponderada (*Weighted Diagnostic Distortion* – WDD), proposta por Zigel et al. (1997). Todavia, essa medida pode ser apenas aplicada a sinais eletrocardiográficos, já que é baseada na preservação das características das ondas PQRST, presentes apenas em ECGs. Além disso, a elevada complexidade envolvida em seu cálculo limita sua aplicabilidade (Zigel e Cohen, 1999). Diferentes registros biomédicos podem apresentar características extremamente distintas e uma medida de distorção adequada deve levar em conta a preservação do traçado e das características relevantes para diagnóstico médico do sinal original, sendo assim, a criação de uma medida universal de distorção objetiva adequada para todos esses sinais é altamente improvável.

3.3 Informação e Entropia

Segundo Shannon (1948), o problema fundamental da comunicação é o de reproduzir em um ponto, exatamente ou aproximadamente, uma mensagem gerada em outro

ponto. Qualquer sistema que trate da transmissão de informação segue o diagrama da Figura 3.2, composto por 5 elementos principais: a fonte de informação, responsável pela criação da mensagem; o transmissor da informação, responsável pela codificação da mensagem e pela emissão da mesma; uma fonte de ruído, que age sobre o canal de transmissão; o receptor da mensagem, que decodifica os dados recebidos do canal de transmissão para uma linguagem conhecida pelo destino; e o destino da mensagem.

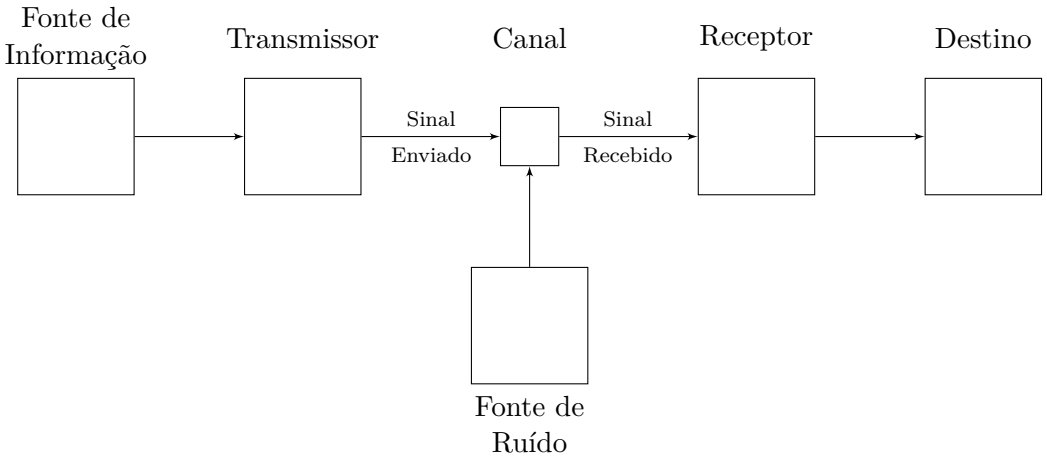


Figura 3.2: Diagrama de um esquema geral de comunicação. Fonte: Shannon (1948).

De acordo com a Teoria da Informação proposta por Shannon (1948), os aspectos semânticos da informação são irrelevantes para o estudo do ponto de vista da engenharia. Em outras palavras, a informação é influenciada apenas pelas características sintáticas e léxicas de uma mensagem, e não pelo seu significado – pela sua semântica.

A informação pode ser entendida como o grau de liberdade de escolha que o emissor tem quando emite uma mensagem (Shannon e Weaver, 1998). Esse grau de liberdade é uma função logarítmica da probabilidade do surgimento do símbolo numa mensagem emitida pela fonte de informação. A base desse logaritmo é definida pela forma de representação escolhida para o símbolo. Por exemplo, caso se deseje expressar um algarismo na base decimal, o logaritmo terá base 10. Comumente a informação é expressa em *bits*, logo o cálculo de informação é feito com logaritmos na base 2 e a unidade da

informação é dada em *bits*. A Equação 3.7 demonstra o cálculo da auto-informação contida num símbolo n, que tem probabilidade p_n de ser emitido por uma fonte de informação que emite símbolos na base b:

$$I_n = \log_b \left(\frac{1}{p_n} \right) \quad \text{dígitos na base } b. \tag{3.7}$$

Como o cálculo da informação é normalmente feito em termos de dígitos binários, uma definição mais específica é dada por:

$$I_n = \log_2 \left(\frac{1}{p_n} \right) \quad \text{dígitos na base 2 } (\textit{bits}). \tag{3.8}$$

Se $S_1, S_2, ..., S_{N-1}$ são os elementos gerados por uma fonte de informação S, a entropia da fonte é definida de acordo com a Equação 3.9:

$$H(S) = \lim_{N \to \infty} \frac{1}{N} G_N \quad \textit{bits} \text{ / símbolo}, \tag{3.9}$$

na qual G_N é dado por:

$$G_N = - \sum_{i_0=0}^{M-1} \sum_{i_1=0}^{M-1} ... \sum_{i_{N-1}=0}^{M-1} p_s \log_2 (p_s), \tag{3.10}$$

M representa a quantidade de símbolos no alfabeto e a probabilidade p_s é descrita pela fórmula:

$$p_s = p\left(x_0 = a_{i_0},\, x_1 = a_{i_1},\, ...,\, x_{N-1} = a_{i_{N-1}}\right). \tag{3.11}$$

Se as variáveis aleatórias que constituem a mensagem são independentes e identicamente distribuídas, a Equação 3.9 pode ser simplificada para expressar a entropia de ordem 1 de S (Sayood, 1996):

$$H(S) = - \sum_{i=0}^{M-1} p(x = a_i) \log_2 p(x = a_i) \quad \textit{bits}/\text{símbolo}. \tag{3.12}$$

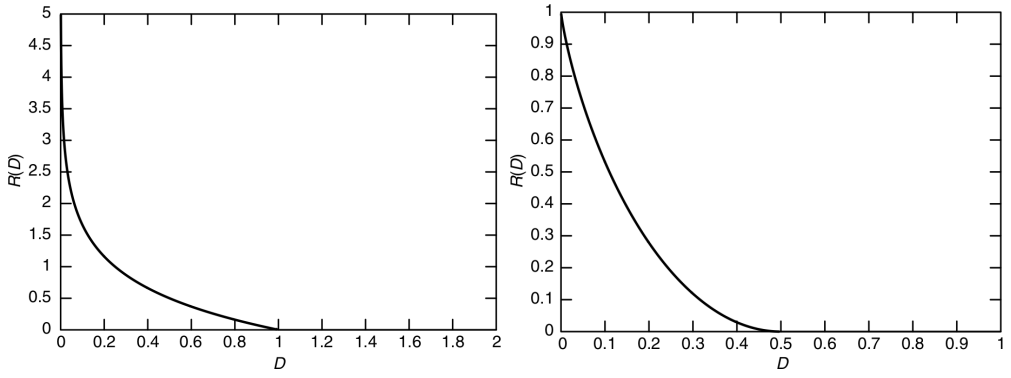
Como define Shannon (1948), é possível codificar sem perdas a saída de uma fonte de informação com um número médio de *bits* por símbolo arbitrariamente próximo à entropia, mas não inferior a ela. A entropia é entendida também como uma medida de avaliação para uma compressão efetuada numa mensagem, de forma que o código ótimo tenha o comprimento médio arbitrariamente próximo à entropia, mas nunca menor.

3.4 Função Taxa-Distorção

De acordo com Cover e Thomas (2012), o principal problema na Teoria Taxa-Distorção pode ser definido de duas formas:

- Dada a distribuição de uma fonte de informação e uma medida de distorção, qual é a menor distorção esperada atingível para uma certa taxa de *bits*?
- Ou, equivalentemente, qual é a menor taxa de *bits* requerida para se atingir uma certa distorção?

Como explicado por Hoang (1997), a definição clássica da Teoria Taxa-Distorção para compressores *lossy* baseia-se em um plano cartesiano de pontos que representam os valores da taxa de *bits* (*Rate* – R) do sinal comprimido e os valores de algum critério de distorção D. Como descrita por Shannon (1948), uma função $R(D)$ – denominada função Taxa-Distorção – pode ser definida como a borda inferior teórica da melhor compressão atingível como uma função da distorção D para uma fonte de informação dada. O critério de distorção, em geral, pode ser qualquer métrica válida. Na prática, costuma-se usar o MSE por sua facilidade de cálculo e proporcionalidade direta à quantidade de erros. Para fontes discretas, $R(0)$ é a entropia da fonte e corresponde à compressão sem perdas ($D = 0$). Exemplos de bordas inferiores teóricas podem ser vistas na Figura 3.3 para fontes de informação com distribuições Gaussianas e distribuições de Bernoulli.



(a) (b)

Figura 3.3: Funções Taxa-Distorção teóricas para distribuições (a) Gaussianas. (b) de Bernoulli. Fonte: Cover e Thomas (2012).

Para fontes com funções densidade de probabilidade bem definidas, podem ser calculadas formas analíticas para a função $R(D)$. Por exemplo, Hoang (1997) apresenta a função densidade de probabilidade usando a MSE como métrica de distorção de uma fonte Gaussiana com variância σ^2 como sendo:

$$R(D) = \begin{cases} \frac{1}{2}\log_2 \frac{\sigma^2}{D}, & \text{para} \quad 0 \leq D \leq \sigma^2 \\ 0 \qquad\quad , & \text{para} \quad D > \sigma^2 \end{cases} . \tag{3.13}$$

3.5 Taxa-Distorção Operacional

Segundo Batista (2002), em situações práticas, a aplicação da Teoria Taxa-Distorção nem sempre produz resultados exatos, pois as fontes de informação reais podem desviar-se significativamente das hipóteses simplificadoras necessárias para obtenção da função Taxa-Distorção – tais quais elementos estatisticamente independentes, função densidade de probabilidade analiticamente tratável, estacionaridade, etc. Uma alternativa para a abordagem analítica é a medição das taxas e distorções reais atingidas pelo compressor para um conjunto finito, ainda que potencialmente grande, de instâncias dos parâmetros utilizados na compressão.

3.5.1 O Plano Taxa-Distorção Operacional

Cada instância de parâmetros ϕ gera um ponto no Plano Taxa-Distorção (Plano *Rate-Distortion*, ou Plano R-D) Operacional, denominado de ponto operacional. Dessa forma, cada ponto P_i gerado no Plano R-D Operacional corresponde a uma instância ϕ_i e está localizado num plano cartesiano nas coordenadas (D_i, R_i). A Figura 3.4 mostra alguns pontos hipotéticos num Plano R-D Operacional.

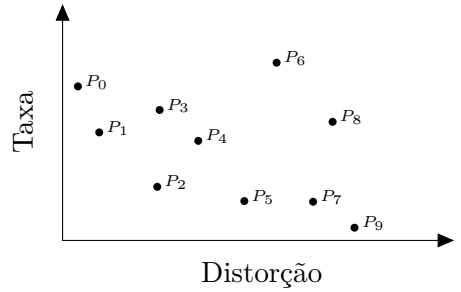


Figura 3.4: Exemplo de Plano R-D Operacional. Fonte: Batista (2002).

Pode ser vista nesse gráfico a grande influência da escolha dos parâmetros de compressão sobre o resultado final de taxa e distorção. Tomando como exemplo os pontos P_2 e P_3, percebe-se que ambos têm mais ou menos o mesmo valor de distorção, mas o ponto P_2 tem menos da metade da taxa de *bits* necessária para representar os coeficientes comprimidos, logo o conjunto de parâmetros ϕ_2 é mais adequado para se usar no caso de uma distorção desejada $D = D_2$. Já os pontos P_5 e P_7 têm taxas de *bits* R_5 e R_7 bastante parecidas, mas os parâmetros ϕ_7 ligados ao ponto P_7 geram uma distorção D_7 bem maior que a distorção D_5 gerada pelos parâmetros ϕ_5 do ponto P_5, portanto o conjunto de parâmetros ϕ_5 é indicado para ser usado com a finalidade de atingir a taxa de compressão $R = R_5$. Uma comparação de pontos como P_0 e P_9 não é conclusiva, já que P_0 apresenta uma baixa distorção ao custo de uma alta taxa de *bits*, enquanto P_9 gera uma taxa de *bits* menor, mas ao custo de uma distorção maior. Comparando P_2 e P_8 percebe-se que P_2 gera uma taxa de *bits* menor com uma menor

distorção do sinal reconstruído, o que faz dos parâmetros ϕ_2 uma escolha melhor que ϕ_8, tanto da perspectiva de taxa de *bits* quanto de distorção.

Como afirma Batista (2002), devido à natureza discreta do Plano R-D Operacional, a Formulação 3.1 só tem solução quando existe ao menos um ponto operacional com distorção igual à distorção alvo D_t. A Formulação 3.2, apresentada a seguir, é mais apropriada para o caso discreto, já que aceita soluções com uma tolerância de distorção de valor menor ou igual a τ. Essa tolerância τ pode ser definida empiricamente, já que depende do sinal que está sendo codificado e da quantidade de erros que pode ser tolerada na compressão.

Formulação 3.2 : Dada uma distorção D_T e uma tolerância τ, encontrar uma instância de parâmetros do compressor que minimize R, com $D_T - \tau \leq D \leq D_T + \tau$.

3.5.2 Conjunto Convexo Gerado pelos Pontos Operacionais

Como definido por Batista (2002), um polígono convexo gerado por m pontos $P_0, ..., P_{m-1}$ em $\mathbb{R}^2$ é o conjunto produzido por todas as combinações convexas dos pontos geradores. Os pontos geradores que não podem ser produzidos por combinação convexa de outros pontos formam os extremos (vértices) do polígono.

As instâncias de parâmetros que formam o Plano R-D Operacional são finitas e têm valores tanto de taxa de *bits* quanto de compressão finitos, logo, um polígono convexo é formado pelo conjunto de pontos no plano. Um polígono convexo é caracterizado por não ter reentrâncias – regiões côncavas – ou áreas internas vazias. A Figura 3.5 mostra os pontos de um Plano R-D Operacional e o polígono convexo por eles gerado.

A convexidade de um polígono lhe fornece uma série de propriedades especiais, entre elas, a característica de sempre formar uma Borda Convexa Inferior (BCI). Essa borda inferior pode ser usada numa técnica de compactação como uma forma de diminuir a complexidade computacional do cálculo dos parâmetros ótimos passados para um compressor *lossy*. Isso se dá porque todos os pontos pertencentes à BCI têm a menor taxa

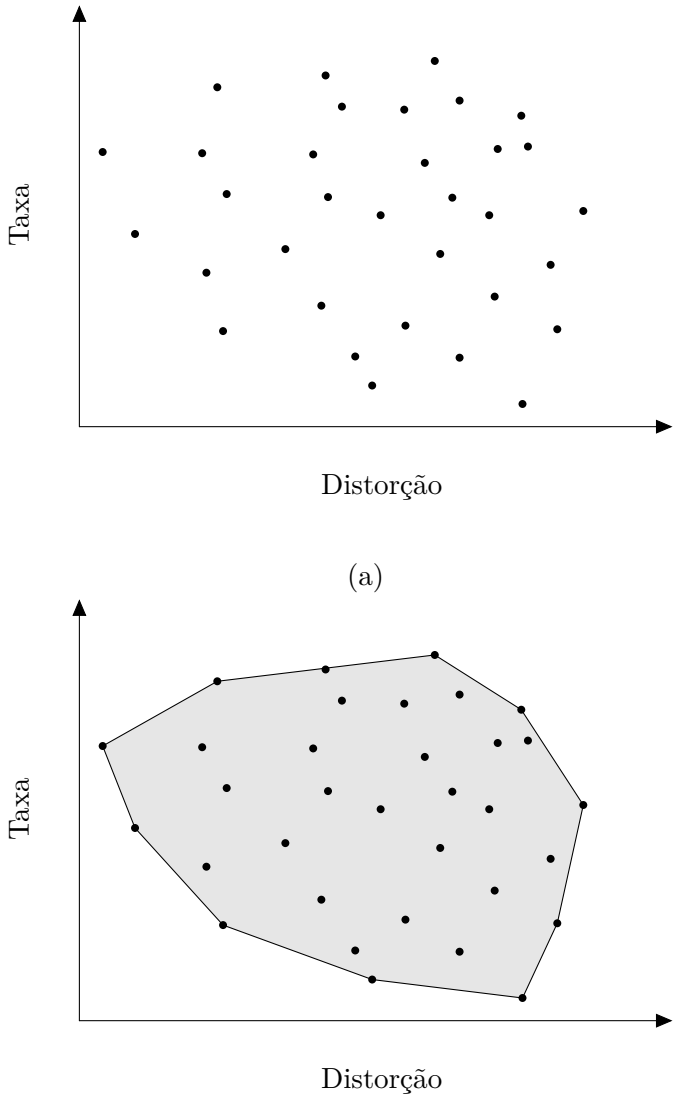


Figura 3.5: Exemplo de Plano R-D Operacional. (a) Pontos do Plano R-D Operacional. (b) Polígono convexo gerado pelos pontos do Plano R-D Operacional.

de *bits* possível para sua respectiva distorção e, analogamente, a menor distorção para sua taxa de *bits*. Essa característica permite restringir a busca da melhor combinação de parâmetros que solucione a Formulação 3.2 aos pontos pertencentes à BCI. A Figura 3.6 mostra a BCI gerada pelo polígono convexo da Figura 3.5b.

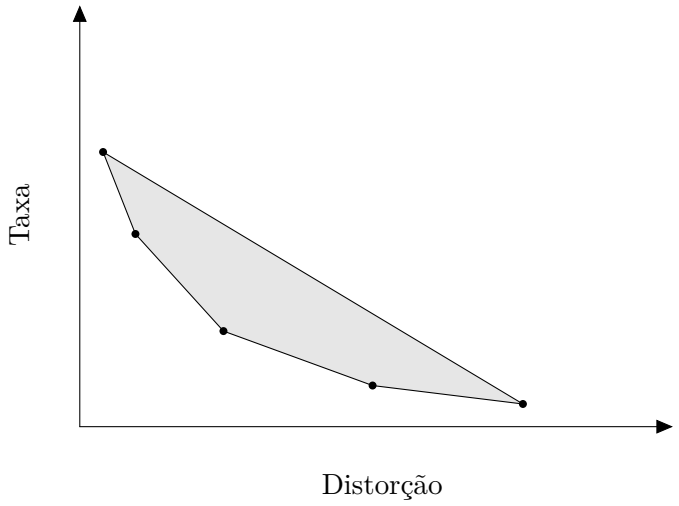


Figura 3.6: Exemplo de BCI de um Plano R-D Operacional.

Segundo Ratnakar e Livny (1996), sabendo-se que qualquer ponto operacional $P_i = (D_i, R_i)$ localizado na BCI é ótimo para a distorção $D = D_i$ e que buscas convexas podem ser eficientemente efetuadas, muitos métodos práticos de busca se restringem à BCI. Um fator que deve ser levado em consideração para garantir que se ache uma solução para a Formulação 3.2 na BCI é a densidade dos pontos obtidos no Plano R-D Operacional. Essa densidade deve ser suficiente para garantir que um baixo valor de tolerância τ seja usado durante a busca no espaço de soluções, o que deve ser investigado empiricamente para cada tipo de sinal sobre o qual se deseja otimizar o processo de compressão.

3.5.3 Minimização Lagrangiana

Como demonstrado por Batista (2002) e Ratnakar e Livny (1996), para buscas sobre a BCI, o problema de otimização da taxa com distorção restrita pode ser transformado em um problema de otimização sem restrição usando-se Minimizações Lagrangianas:

Formulação 3.3 : Dado um parâmetro real $\lambda \geq 0$, encontrar uma instância de parâmetros do compressor que minimize o lagrangiano $J(\phi) = R(\phi) + \lambda D(\phi)$.

O parâmetro λ é denominado multiplicador lagrangiano. Para λ fixo, $J = R + \lambda D$ representa uma reta no Plano R-D Operacional com declividade $-\lambda$, interceptando o eixo R em J e o eixo D em J/λ. O método de busca usando multiplicadores de Lagrange pode ser entendido como uma busca pelo último ponto operacional interceptado numa varredura em direção à origem realizada pela reta $J = R + \lambda D$. Essa afirmação é representada graficamente na Figura 3.7, na qual a varredura é feita começando do valor J_0 e achando o último ponto no valor J_3. A busca no espaço de parâmetros é realizada, então, iterando sobre diferentes valores de λ. Dessa forma, a busca pode ser feita com uma complexidade logarítmica ao se selecionar valores intermediários de lambda em cada iteração do algoritmo, diminuindo consideravelmente o tempo de procura por uma solução que satisfaça a Formulação 3.2, dado que a densidade do Plano R-D Operacional seja grande o bastante.

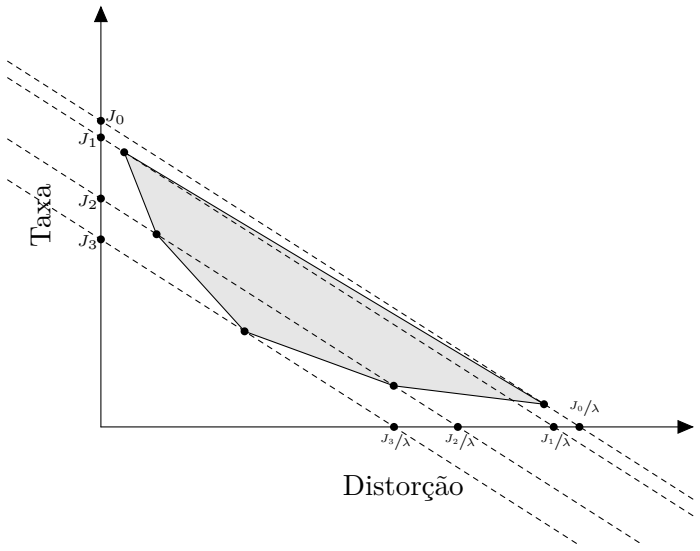


Figura 3.7: Varredura da busca para um valor λ fixo do multiplicador de Lagrange.

3.6 Conclusão do Capítulo

O capítulo atual descreveu em detalhes alguns conceitos importantes para a área de compressão de dados e processamento de sinais. Esses conceitos serão usados em conjunto com as técnicas apresentadas no Capítulo 4, dando origem aos compressores de testes apresentados no Capítulo 5. A Seção 5.2.2 apresenta uma aplicação da minimização lagrangiana no contexto da Quantização Vetorial.

Capítulo 4

Técnicas de Compressão de Dados

Este capítulo introduzirá diversos métodos e estratégias de compactação usados por codificadores do estado da arte da literatura da Compressão de Dados. Os diferentes métodos apresentados serão organizados em: transformações, Seção 4.1; estratégias de quantização, Seções 4.2.1 e 4.2.2; codificadores de entropia, Seção 4.3. A Seção 4.4 explicará como as técnicas de transformação, quantização e codificação de Entropia se organizam num compressor *lossy*. Por último, as Seções 4.5 e 4.6 tratarão de explicar as técnicas de compressão baseadas em preditores lineares e no algoritmo K-Médias.

4.1 Transformações

De acordo com Salomon (2006), transformações – ou transformadas, como também são chamadas – são ferramentas que operam ao mudar uma entidade matemática (um número, um vetor, uma função, etc) para um outro domínio, no qual ela pode se tornar irreconhecível, mas apresentar propriedades úteis para a operação que se deseja realizar. Essa entidade transformada é usada para resolver um problema ou realizar um cálculo,

e o resultado, então, transformado para o domínio original.

Transformações são usadas nas mais diversas subáreas no Processamento Digital de Sinais, incluindo o reconhecimento de padrões, diagnóstico médico, biometria e – especialmente para o contexto desse trabalho – compressão de dados. Algumas transformadas apresentam propriedades especiais que facilitam a compactação de dados, se usadas de forma inteligente. Transformadas trigonométricas baseadas na função cosseno, por exemplo, concentram grande parte da informação importante de sinais contínuos digitalizados nos coeficientes transformados de menor frequência. *Wavelets*, por sua vez, separam os coeficientes de alta e baixa frequência dos sinais em diferentes camadas organizadas hierarquicamente, permitindo que sejam codificadas apenas as frequências que minimizem erros de reconstrução.

Britanak et al. (2006) generalizaram as características desejáveis para uma transformada usada na compressão de imagens – apresentadas por Rabbani e Jones (1991) – para o domínio mais abrangente da compressão de sinais. Assim, como descrevem Britanak et al. (2006), o seguinte conjunto de propriedades desejáveis para transformadas foi proposto:

- **Descorrelação dos Dados:** a transformada ideal descorrelaciona completamente os dados de uma sequência/bloco, ou seja, ela concentra uma grande quantidade de energia (informação) numa quantidade pequena de coeficientes. Dessa forma, vários coeficientes podem ser descartados após uma quantização e antes da codificação de entropia. É importante notar que a operação da transformada em si não atinge nenhum grau de compressão. Ela simplesmente deve descorrelacionar os dados originais e compactar uma grande parte da energia do sinal em uma quantidade relativamente pequena de coeficientes transformados.

- **Função-Base Independente dos Dados:** devido à grande variação estatística dos dados, a transformada ótima normalmente depende dos dados analisados,

tornando o processo de cálculo das funções-base dessas transformadas computacionalmente caro. Isso é particularmente um problema se os blocos de dados forem altamente não estacionários, o que pode levar à necessidade de usar mais de um conjunto de funções-base para atingir altos níveis de descorrelação. Assim, é necessário um compromisso entre *performance* ótima por uma transformada que tenha funções-base independentes dos dados a serem transformados.

- **Implementação Rápida:** o número de operações requeridas por uma transformada em n amostras é geralmente de ordem $\mathcal{O}(n^2)$. Algumas transformadas têm implementações rápidas que reduzem o número de operações para $\mathcal{O}(n \log n)$. Para uma transformada bidimensional separável executada em $n \times n$ amostras, calcular transformadas sequenciais unidimensionais nas linhas e colunas do sinal bidimensional reduz o número de operações de $\mathcal{O}(n^4)$ para $\mathcal{O}(2n^2 \log n)$.

A Transformada de Karhunen-Loève (*Karhunen-Loève Transform* – KLT) (Karhunen, 1947; Loève, 1948) atinge otimamente a primeira característica desejável, mas é penalizada nas duas últimas. Ainda segundo Britanak et al. (2006), a KLT é uma transformada ótima para a compressão de dados do ponto de vista estatístico, porque ela descorrelaciona um sinal no domínio transformado, junta a maior parte da informação em uma quantidade pequena de coeficientes e, posteriormente, minimiza o MSE entre o sinal reconstruído e o original. Porém, a KLT é construída a partir dos autovalores (*eigenvalues*) e dos seus respectivos autovetores (*eigenvectors*) de uma matriz de covariância do sinal a ser transformado. Ela é uma transformada dependente do sinal e não há um algoritmo geral para o seu cálculo acelerado.

As transformadas trigonométricas e as baseadas em *wavelets* atingem de forma satisfatória os três critérios apresentados por Britanak et al. (2006), constituindo uma grande parte da literatura do estado da arte da compressão *lossy* atualmente. As próximas seções detalharão algumas transformadas usadas para compressão.

4.1.1 Transformadas Trigonométricas

A Transformada Discreta do Cosseno (*Discrete Cosine Transform* – DCT) e a Transformada Discreta do Seno (*Discrete Sine Transform* – DST) são membros da família de transformadas unitárias senoidais. Elas têm aplicações no processamento de imagens e de sinais digitais e particularmente em sistemas de codificação baseados em transformadas para compressão/descompressão (Britanak et al., 2006). A DCT tende a apresentar desempenho ótimo de compressão para sinais estacionários Markov-1 quando o coeficiente de correlação entre amostras adjacentes tende a 1 (Rao e Yip, 2014). Um símbolo pertencente a uma mensagem gerada por uma fonte Markov-1 pode ser predita apenas ao se observar o último símbolo gerado. Como disserta Batista (2002), na prática, a DCT geralmente se comporta de forma satisfatória mesmo em situações em que o sinal desvia-se consideravelmente das condições apresentadas previamente.

O conjunto completo de DCTs e DSTs foi primeiramente descrito por Wang (1984). Essas transformadas trigonométricas discretas consistem em oito versões de DCTs e oito versões correspondentes de DSTs (Martucci, 1992). Cada transformada pode ser classificada como par ou ímpar e pertencente aos tipos I, II, III e IV. Todas as aplicações presentes no processamento de sinais e imagens (principalmente a codificação baseada em transformadas e os filtros digitais) envolvem apenas os tipos pares da DCT e DST (Britanak et al., 2006). Exemplos de compressores de imagens e vídeos baseados em transformadas trigonométricas são o JPEG (*Joint Photographic Experts Group*) (Wallace, 1991), o H.264 (ITU-T264, 2002) e o H.265 (Sullivan et al., 2012). Como definem Britanak et al. (2006), as formas matriciais das DCTs pares de tipos I, II, III e IV são encontradas nas Equações 4.1, 4.2, 4.3 e 4.4.

$$DCT-I: \quad \left[C^{I}{}_{N+1}\right]_{nk} = \sqrt{\frac{2}{N}}\left[\epsilon_n \epsilon_k \cos\frac{\pi nk}{N}\right], \quad n,k = 0,\ 1,\ ...,\ N, \tag{4.1}$$

$$DCT-II: \quad \left[C^{II}{}_N\right]_{nk} = \sqrt{\frac{2}{N}}\left[\epsilon_k \cos\frac{\pi\,(2n+1)\,k}{2N}\right], \quad n,k = 0,\ 1,\ ...,\ N-1, \tag{4.2}$$

$$DCT-III: \quad \left[C^{III}{}_N\right]_{nk} = \sqrt{\frac{2}{N}}\left[\epsilon_n \cos\frac{\pi\,(2k+1)\,n}{2N}\right], \quad n,k = 0,\ 1,\ ...,\ N-1, \tag{4.3}$$

$$DCT-IV: \quad \left[C^{IV}{}_N\right]_{nk} = \sqrt{\frac{2}{N}}\left[\cos\frac{\pi\,(2n+1)\,(2k+1)}{4N}\right], \quad n,k = 0,\ 1,\ ...,\ N-1, \tag{4.4}$$

nas quais ϵ_n e ϵ_k se comportam de acordo com:

$$\epsilon_p = \begin{cases} \frac{1}{\sqrt{2}}, & \text{para } p = 0 \text{ ou } p = N \\ 1, & \text{caso contrário} \end{cases} . \tag{4.5}$$

A DCT tem a capacidade de concentrar a parte mais importante da informação nos primeiros elementos, os quais representam as frequências mais baixas de símbolos no sinal original. O primeiro coeficiente é chamado de coeficiente DC e os outros são referidos por coeficientes AC (esses termos foram herdados da engenharia elétrica e significam "corrente contínua" e "corrente alternada") (Salomon, 2006). O DC normalmente é o coeficiente que concentra a maior quantidade de informação e representa uma média da linha de base (*baseline*) do sinal. Na Figura 4.1 pode ser visto um ECG e na parte inferior a DCT do mesmo sinal dividido em blocos demarcados pelas linhas verticais. Percebe-se a concentração de informação em alguns coeficientes do sinal transformado, principalmente nos de baixa frequência – ou seja, os mais próximos dos inícios dos blocos. Essas áreas de concentração, se codificadas de forma correta, podem resultar em boas RCs com baixos níveis de erros de reconstrução no sinal decodificado. A Figura 4.2 facilita a visualização da concentração de informação, apresentando uma DCT bidimensional sobre uma imagem, na qual a concetração de informação está representada pela

cor vermelha e aparece principalmente na borda superior esquerda da figura.

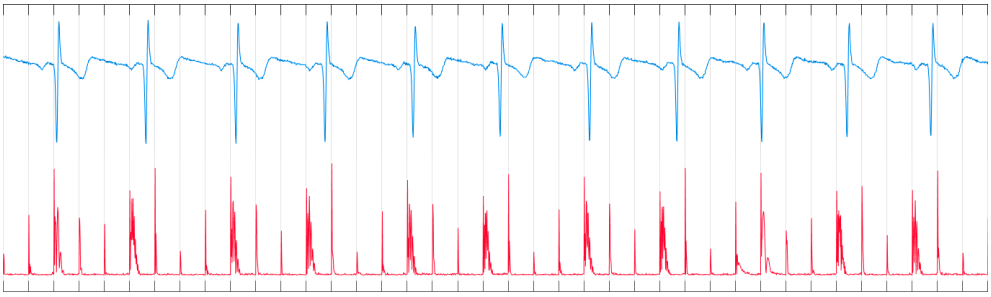


Figura 4.1: Registro de ECG original e seu correspondente transformado por DCTs nos blocos delimitados pelas linhas verticais.

(a) (b)

Figura 4.2: Imagem (a) antes da DCT. (b) após a DCT.

A concentração da informação em certos coeficientes dos sinais transformados por DCTs, porém, pode ter o seu custo em termos de qualidade de reconstrução. Como uma frequência presente no sinal original inteiro é representada por apenas um coeficiente no domínio gerado pela DCT, uma quantização aplicada num coeficiente importante para a reconstrução pode gerar artefatos indesejáveis no sinal reconstruído inteiro. Dessa forma, compressores que utilizem DCTs devem saber identificar quais frequências são mais importantes para a reconstrução do sinal. No caso da compressão de imagens, por exemplo, muitas altas frequências geradas pela DCT, além de produzirem normalmente

coeficientes de magnitude pequena em imagens lineares, podem sofrer uma quantização maior do que níveis que representem baixas frequências. Isso se dá pela dificuldade de percepção de altas frequências pelo olho humano, que tende a não identificar quantizações – mesmo que sejam pesadas – nesses coeficientes. Já no caso de EEGs, as altas frequências devem sofrer uma ação menor das etapas de quantização pós-transformada, já que esses sinais tendem a conter desníveis mais repentinos e acentuados, e, portanto, coeficientes de alta frequência mais importantes.

Dentre os vários tipos de DCT, a DCT-II e a DCT-III têm recebido muita atenção na área de processamento de sinais. Além de serem ortogonais, separáveis e operarem em números reais, suas propriedades são relevantes para a compressão de dados e algoritmos rápidos para seu cálculo têm sido aplicados na prática (Britanak et al., 2006). Porém, a DCT não é aplicada comumente no sinal inteiro de uma vez, já que, além de ter uma complexidade computacional elevada, a transformada produz muitos coeficientes de alta frequência importantes, o que não é bom para propósitos de compressão. A DCT é aplicada consecutivamente em blocos de poucas amostras – como demonstrado pela Figura 4.1 – o que gera uma concentração bem maior de informação nos primeiros elementos dos blocos. Esse método tem o lado negativo de produzir artefatos potencialmente danosos à qualidade visual do sinal nas regiões de fronteiras de blocos.

No caso de compressão de imagens, assume-se que os coeficientes AC seguem distribuições Laplacianas (Rao e Yip, 2014; Jayant e Noll, 1990), enquanto o coeficiente DC segue uma distribuição Gaussiana ou Rayleigh (Rao e Yip, 2014). Assim, codificadores de entropia que comprimam de forma eficiente mensagens com essas distribuições estatísticas devem ser utilizados.

O mesmo comportamento das Funções de Distribuição Cumulativas (*Cumulative Distribution Function* – CDF) observado em imagens por Rao e Yip (2014) e Jayant e Noll (1990) pode ser visto para a maioria dos sinais biomédicos introduzidos no Capítulo 2, como apresentado no Apêndice A.

Apesar das DSTs usarem uma função (o seno) que é apenas uma mudança de fase da função utilizada pela DCT (o cosseno), ela só agora está começando a ser utilizada com sucesso nos métodos de compressão de dados. Como apontado por Han et al. (2010), sinais que apresentem uma concentração de informação na área central de suas amostras – seja devido a artefatos gerados por predições ou simplesmente pela natureza do sinal – terão uma representação compacta após serem transformados por DSTs. Isso se dá pelo comportamento dos períodos da função seno, que começam com valores baixos e crescem, enquanto os períodos da função cosseno começam com valores altos e decrescem.

4.1.2 Transformadas *Wavelet*

Como afirma Salomon (2006), transformadas *wavelet* são uma abordagem adequada para analisar um sinal nos domínios naturais – tempo ou espaço – e no domínio da frequência. Dado um sinal que varie na dimensão temporal, é possível selecionar um intervalo de tempo e aplicar uma transformada *wavelet* com a finalidade de isolar as frequências que constituem o sinal naquele intervalo. Se o intervalo selecionado for grande, é dito que a análise está sendo feita em uma larga escala. À medida que o intervalo diminui, considera-se que a análise passa a ser feita numa escala cada vez mais curta. Uma análise em larga escala ilustra o comportamento global do sinal, enquanto cada análise numa curta escala demonstra como o sinal se comporta num curto intervalo; é como se fosse dado um zoom no sinal. Assim, transformadas *wavelet* fazem análises de funções de acordo com a escala.

Wavelets são funções-base que representam uma dada função em múltiplos níveis de detalhe (Schröder e Sweldens, 1995). Uma transformada *wavelet* consiste de suscetivas aproximações de um sinal de entrada de acordo com o traçado de uma dessas funções-base – chamada de *Wavelet*-mãe – gerando um sinal de saída com um domínio determinado pela função-base. As sucessivas operações de aproximação realizadas no

sinal são chamadas de passos da *wavelet* e, à medida que esses passos vão sendo executados, análises mais gerais do sinal – ou seja, de larga escala – são realizadas. Como esclarece Salomon (2006), sinais transformados por *wavelets* em aplicações práticas, como sons e imagens digitais, são discretos, consistindo de números individuais. É por isso que a forma discreta, e não contínua, da transformada *wavelet* é usada na prática, sendo conhecida como a Transformada *Wavelet* Discreta (*Discrete Wavelet Transform* – DWT).

DWTs vêm sendo usadas nas últimas décadas para a compressão de sinais de ECG (Lu et al., 2000; Istepanian e Petrosian, 2000; Miaou e Chao, 2005; Kim et al., 2006), EEG (Higgins et al., 2010; Srinivasan et al., 2013; Kedir-Talha e Amer, 2013) e EMG (Wellig et al., 1998; de A.Berger et al., 2007; Brechet et al., 2007), caracterizando as *wavelets* como um dos métodos mais popularmente utilizados na literatura da área de compressão de sinais biomédicos.

A primeira etapa a ser executada para o cálculo de uma DWT é a escolha da função que servirá como *wavelet*-mãe – uma função que é não nula em um pequeno intervalo (Salomon, 2006) – de modo que as propriedades do sinal que se deseja transformar sejam exploradas na transformação. Após cada passo da execução de uma transformada DWT, uma operação de subamostragem pelo fator 2 é realizada, diminuindo o tamanho da janela de tempo do sinal transformado. Essas operações consecutivas de subamostragem, após algumas iterações do algoritmo, podem chegar a resultar em apenas um coeficiente, que representará as frequências do sinal inteiro. A soma da quantidade de coeficientes em todos os níveis do algoritmo, porém, deve ser constante e igual ao tamanho do sinal original.

A DWT é uma transformada que gera um sinal transformado em níveis hierárquicos de forma recursiva, logo os coeficientes transformados são relacionados entre si, como demonstrado pelas setas ligando os coeficientes na Figura 4.3.

Como ilustrado na Figura 4.3a, S é um sinal que contém $N = 8$ amostras. O

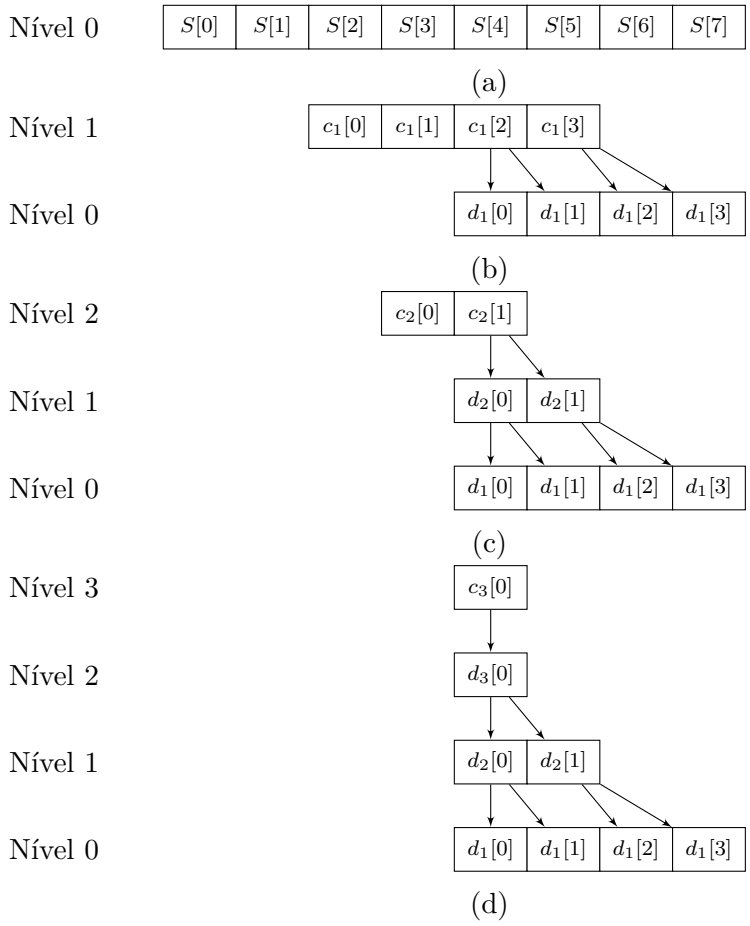


Figura 4.3: Diagrama dos 3 passos de uma DWT num bloco de 8 amostras. (a) Sinal inicial. (b) Sinal após o primeiro passo da DWT. (c) Sinal após o segundo passo da DWT. (d) Sinal após o terceiro passo da DWT.

primeiro passo da DWT é realizado nas N amostras do sinal S, gerando dois sinais c_1 e d_1, cada um com $N/2 = 4$ coeficientes. O sinal subamostrado c_1 – chamado de sinal bruto (*coarse*) – representa as baixas frequências do sinal S, enquanto o sinal subamostrado d_1 – nomeado de sinal de detalhes (*detail*) – representa as altas frequências. O sinal S é, então, substituído pelos coeficientes de c_1 e d_1, mantendo o tamanho original $N = 8$, como pode ser visto na Figura 4.3b.

Analogamente ao que acontece no primeiro passo, a cada passo subsequente de índice i do algoritmo, um nível mais alto de coeficientes é criado, gerando os sinais

c_i e d_i, ou seja, os sinais *coarse* e *detail* referentes ao passo i. A diferença é que no nível i a janela na qual a DWT é aplicada não engloba mais as N amostras do sinal S, mas sim o sinal *coarse* do passo anterior, ou seja, c_{i-1}. No caso na Figura 4.3b, a DWT é aplicada no seu segundo passo de execução sobre o sinal c_1, que é substituído posteriormente pelos sinais c_2 e d_2, de tamanho $N/4 = 2$ cada. Na etapa ilustrada pela Figura 4.3c o sinal contém três níveis: nível 0 (d_1), contendo 4 coeficientes; nível 1 (d_2), com 2 coeficientes; e nível 2 (c_2), que possui também 2 coeficientes. Nota-se que o somatório da quantidade de coeficientes nos níveis novamente permanece inalterado após o segundo passo do algoritmo, totalizando $N = 8$ coeficientes.

No terceiro e último passo da transformada, os 2 coeficientes *coarse* gerados pelo segundo passo – ou seja, $c_2[0]$ e $c_2[1]$ – são transformados novamente, gerando os sinais *coarse* (c_3) e *detail* (d_3), cada um com apenas 1 coeficiente. Esse passo resulta no sinal mostrado na Figura 4.3d. Os níveis 2 e 3, como contêm o número mínimo de coeficientes permitido por uma DWT, representam, respectivamente, baixas e altas frequências que inicialmente estavam espalhadas pelo sinal S inteiro. Isso significa que a análise feita por esses níveis é de larga escala, em contraste às análises feitas pelo nível 0 (d_1) e nível 1 (c_1) da Figura 4.3b, ambas de curta escala. O coeficiente $d_1[1]$, por exemplo, representa os detalhes referentes apenas às amostras $S[2]$ e $S[3]$.

A Transformada *Wavelet* Discreta Inversa (*Inverse Discrete Wavelet Transform* – IDWT) é definida pelos passos inversos da DWT. Novamente a Figura 4.3 será usada para ilustrar o funcionamento do cálculo. Na IDWT, os níveis mais altos são transformados primeiro, sendo agrupados em níveis menores com uma quantidade maior de coeficientes, ou seja, análises em menor escala do sinal são realizadas a cada iteração. No primeiro passo da IDWT, os coeficientes dos sinais c_3 e d_3 são agrupados e interpolados, passando de dois sinais de tamanho $N/8 = 1$ para um único sinal c_2 de tamanho $N/4 = 2$. Posteriormente, os coeficientes de c_2 e d_2 são mesclados e interpolados, produzindo os valores de c_1. Por último, c_1 e d_1 – cada um com $N/2 = 4$ coeficientes – são agrupados

e interpolados, resultando no sinal original S com 8 amostras.

Para melhor ilustrar as iterações executadas no cálculo de uma DWT, a Seção 4.1.2.1 introduzirá a transformada *wavelet* mais simples, a Transformada Haar Discreta (*Discrete Haar Transform* – DHT).

4.1.2.1 A Transformada Haar

Assim como na Seção 4.1.2, a explicação da Transformada Haar será guiada por um exemplo com $N = 8$ amostras iniciais transformadas por uma DWT de 3 passos, como mostrado na Figura 4.4.

Stollnitz et al. (1996) propuseram uma transformação baseada nas funções de Haar. Na prática geralmente se usa a forma não normalizada da DHT, envolvendo simplesmente médias e diferenças calculadas em pares (Calderbank et al., 1998). O conceito dessa transformada parte do pressuposto de que os sinais digitais que estão sendo transformados são contínuos, ao menos em boa parte de suas amostras. Como é possível expressar qualquer par de coeficientes numéricos pelo par média e diferença desses coeficientes, a DHT se aproveita dessa propriedade para compactar a informação das amostras. Caso os valores das duas amostras sejam numericamente próximos, o valor da média do par será próximo do valor de ambas as amostras e a diferença terá uma magnitude pequena.

Em DWTs baseadas na DHT, os valores de média e diferença calculados representam, respectivamente, o *coarse* (c_i) e o *detail* (d_i) da transformada *wavelet* no passo i. O cálculo dos valores *coarse* é regido pela Equação 4.6, enquanto a Equação 4.7 descreve como são calculados os valores dos coeficientes *detail*:

$$c_i[l] = \frac{c_{i-1}[2l] + c_{i-1}[2l+1]}{2}. \tag{4.6}$$

$$d_i[l] = c_{i-1}[2l+1] - c_{i-1}[2l]. \tag{4.7}$$

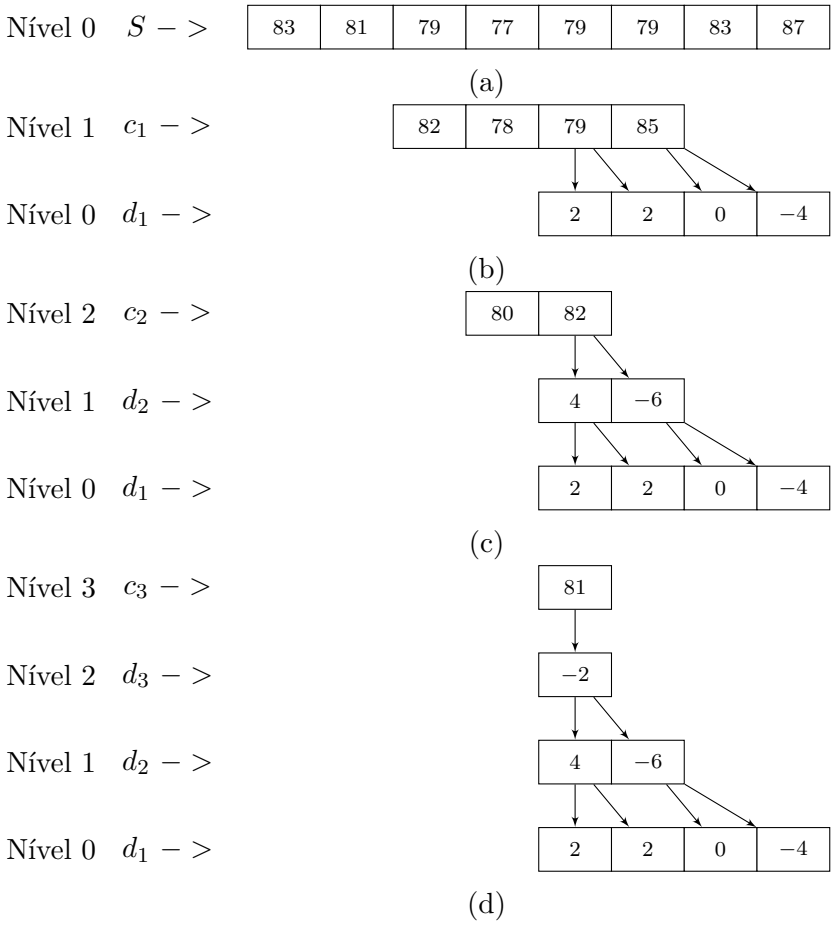


Figura 4.4: Diagrama dos 3 passos de uma DHT num bloco de 8 amostras. (a) Sinal inicial. (b) Sinal após o primeiro passo da DHT. (c) Sinal após o segundo passo da DHT. (d) Sinal após o terceiro passo da DHT.

O *coarse* indica o valor bruto das amostras do sinal original S – ou seja, uma aproximação para ambos os valores do par de amostras original – enquanto o *detail* representa um refinamento para esses valores, proporcionando um maior nível de detalhes, pois especifica cada uma das amostras originais do par. Se fosse necessária apenas uma aproximação para o sinal original, seria possível armazenar apenas os coeficientes *coarse*, resultando numa aproximação baseada num filtro passa baixa executado em pares de amostras. Por isso é correto afirmar que o resultado do filtro passa baixa da DHT, ou seja, os coeficientes *coarse*, concentram a maior parte da informação da mensagem.

Como se assume que o sinal é contínuo, ainda é seguro dizer que existe uma correlação entre coeficientes *coarse* vizinhos, já que as médias consecutivas foram calculadas de acordo com coeficientes com valores numéricos próximos, o que permite que um segundo passo da DWT seja aplicado nesses coeficientes. O sinal c_1, agora processado pela segunda etapa da DHT, gerará os sinais c_2 e d_2 de baixa e alta frequência, respectivamente. Novamente é esperado que os valores dos coeficientes *coarse* sejam altos e próximos dos dois valores que os originaram, enquanto os valores *detail* devem ter uma magnitude baixa. Por último, o terceiro passo da transformada é executado, transformando os $N/4 = 2$ coeficientes de c_2 em d_3 e c_3. Na Figura 4.4d pode ser vista a concentração de informação no nível 3 (coeficiente c_3), que contém o valor 81, enquanto todos os outros níveis contêm valores de pequena magnitude.

A operação da Transformada Haar Discreta Inversa (*Inverse Discrete Haar Transform* – IDHT) é executada, assim como mostrado previamente nesta seção, dos níveis mais altos gerados pela transformada direta até chegar no nível 0. O agrupamento de coeficientes é calculado pelas fórmulas da Equação 4.9 e da Equação 4.8:

$$c_i\,[2l] = c_{i+1}\,[l] + \frac{d_{i+1}\,[l]}{2}. \tag{4.8}$$

$$c_i\,[2l+1] = c_{i+1}\,[l] - \frac{d_{i+1}\,[l]}{2}. \tag{4.9}$$

Através do mecanismo de concentração de informação pelo uso de médias e diferenças, a DHT consegue eliminar a redundância de informação entre amostras parecidas, o que se assemelha a uma codificação diferencial de ordem 1, na qual o primeiro coeficiente é codificado inteiramente e os subsequentes são codificados diferencialmente. Essa definição da DHT baseada em predição de coeficientes é facilmente estendida para qualquer DWT, já que todas contam com filtros passa baixa e passa alta aplicados de forma hierárquica. A DHT, portanto, é considerada a DWT mais simples, já que conta com um modelo de predição que só considera o coeficiente anterior mais próximo para

prever o subsequente.

Agora que a DHT foi explicada em detalhes, as próximas seções tratarão de conceitos relacionados às transformadas *wavelet* usando a Haar como base.

4.1.2.2 Uma Abordagem Matricial para a DWT e Ortonormalidade

Como mostrado por Mulcahy (1996, 1997), é possível criar uma matriz que, quando multiplicada por um vetor, resulte no cálculo de diferenças – níveis *detail* – e médias – níveis *coarse* – de suas amostras. Salomon (2006) demonstra as três matrizes relacionadas aos cálculos do primeiro (A_1), segundo (A_2) e terceiro (A_3) passos de uma transformada *wavelet* para um vetor de tamanho $N = 8$, expressas nas Equações 4.10, 4.11 e 4.12. Usar-se-á o vetor V da Equação 4.13 para ilustrar as operações de multiplicação de matrizes que resultam no vetor transformado T.

$$A_1 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{pmatrix}, \tag{4.10}$$

$$A_2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.11}$$

$$A_3 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.12}$$

$$V = \begin{pmatrix} 255 \\ 224 \\ 192 \\ 159 \\ 127 \\ 95 \\ 63 \\ 32 \end{pmatrix}, \; T = A_3.(A_2.(A_1.V)) = \begin{pmatrix} 143{,}375 \\ 64{,}125 \\ 32 \\ 31{,}75 \\ 15{,}5 \\ 16{,}5 \\ 16 \\ 15{,}5 \end{pmatrix}. \tag{4.13}$$

Essa, porém, é uma forma ineficiente de realizar o cálculo da transformada, já que três multiplicações matriciais são executadas no processo. Como demonstra Salomon (2006), não é preciso calcular médias e diferenças ou mesmo realizar as três operações matriciais da Equação 4.13. É necessário apenas construir as matrizes A_1, A_2 e A_3, multiplicá-las para obter a matriz $W = A_3 \; . \; A_2 \; . \; A_1$ e realizar a operação $T = W \; . \; V$ demonstrada na Equação 4.14. A vantagem dessa abordagem – que à primeira vista é tão custosa quanto o cálculo de T de acordo com a Equação 4.13 – é que a matriz W precisa ser calculada apenas uma vez, e pode ser reutilizada diversas vezes no decorrer de uma aplicação.

$$W = \begin{pmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ \frac{1}{4} & \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{pmatrix}, \; T = W.V = \begin{pmatrix} 143{,}375 \\ 64{,}125 \\ 32 \\ 31{,}75 \\ 15{,}5 \\ 16{,}5 \\ 16 \\ 15{,}5 \end{pmatrix}. \tag{4.14}$$

O cálculo da IDHT é dado pela multiplicação matricial $V = (T' \; . \; W^{-1})'$, como demonstrado na Equação 4.15:

$$W^{-1} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix}, \; V = \left(T'.W^{-1}\right)' = \begin{pmatrix} 255 \\ 224 \\ 192 \\ 159 \\ 127 \\ 95 \\ 63 \\ 32 \end{pmatrix}. \tag{4.15}$$

O cálculo da matriz inversa W^{-1} é, porém, assintoticamente tão custoso computacionalmente quanto uma multiplicação de matrizes, como provado por Cormen et al. (2009). O algoritmo recursivo de Strassen (1969), originalmente usado para multiplicação de matrizes quadradas, pode ser executado para realizar o cálculo de matrizes quadradas inversas num tempo menor que o método tradicional, porém esse algoritmo ainda é executado apenas na ordem $\mathcal{O}\left(n^{\lg(7)}\right) \cong \mathcal{O}\left(n^{2{,}81}\right)$. Nesse ponto, a normalização dos coeficientes não só de DHTs, mas de todas as DWTs, torna-se importante. A Seção 4.1.2.1 apresentou apenas a DHT não normalizada, então mostrar-se-á a versão da transformada de forma que ela gere uma matriz W ortonormal. Como explanado por Salomon (2006), matrizes ortonormais têm a propriedade de que sua matriz inversa é igual à sua matriz transposta, simplificando o cálculo da IDHT para $V = (T' \; . \; W')'$.

O cálculo de matrizes ortonormais se dá pela incorporação do cálculo da raiz quadrada da resolução em cada passo de uma DWT. A resolução de uma DWT num determinado passo é dada pela quantidade de coeficientes *coarse* restantes no final da iteração. Dessa forma, o cálculo da resolução r, após o passo i com $i = 1, 2..., p$, em que p é o número de passos da transformada e N representa o tamanho total do sinal, é dado por:

$$r = \frac{N}{2^i}. \tag{4.16}$$

Assim, após o primeiro passo da transformada, metade do tamanho do sinal original será composto de níveis *coarse*, logo, $r = {}^{N}/_{2^1}$. A raiz quadrada desse valor de resolução é incorporada ao cálculo da transformada, dividindo os valores dos coeficientes *detail* gerados após a execução do passo i. Dessa forma, após o primeiro passo, todos os coeficientes *detail* devem ser divididos por $\sqrt{{}^{N}/_{2^1}}$; após o segundo passo, os níveis *detail* são divididos por $\sqrt{{}^{N}/_{2^2}}$ e assim por diante. No último passo, porém, os níveis *coarse* também são divididos por $\sqrt{r}$. Como descreve Salomon (2006), para as matrizes de normalização da DHT funcionarem, é necessário dividir o sinal original V pela raiz da quantidade $N = 8$ total de suas amostras, ou seja, o valor $\sqrt{8}$. As matrizes A_1, A_2, A_3 e W ortonormais para a DHT e o vetor final T são as expressas nas Equações 4.17, 4.18, 4.19, 4.20 e 4.21:

$$A_1 = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}, \tag{4.17}$$

$$A_2 = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.18}$$

$$A_3 = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.19}$$

$$W = \begin{pmatrix} \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \\ \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} & -\frac{1}{\sqrt{8}} \\ \frac{1}{\sqrt{4}} & \frac{1}{\sqrt{4}} & -\frac{1}{\sqrt{4}} & -\frac{1}{\sqrt{4}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{4}} & \frac{1}{\sqrt{4}} & -\frac{1}{\sqrt{4}} & -\frac{1}{\sqrt{4}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}, \tag{4.20}$$

$$T = W.\frac{V}{\sqrt{8}} = \begin{pmatrix} 143{,}375 \\ 64{,}125 \\ 22{,}6274 \\ 22{,}4506 \\ 7{,}75 \\ 8{,}25 \\ 8 \\ 7{,}75 \end{pmatrix}. \tag{4.21}$$

Se a DWT normalizada for usada, é possível provar formalmente que ignorar as menores diferenças é a melhor escolha para compressão com perdas usando *wavelets*, já que isso causa uma menor perda de informação em imagens (Salomon, 2006). A mesma prova formal apresentada por Salomon (2006) para compressão *lossy* de imagens é válida para sinais n-dimensionais, já que a multidimensionalidade das DWTs é atingida por meio da execução de consecutivas transformadas unidimensionais.

A criação de matrizes W para o cálculo de DHTs, porém, gera um custo de memória de ordem $\Theta\left(n^2\right)$, o que é desnecessário, como será explicado nas próximas seções.

4.1.2.3 Transformação de Haar usando Bancos de Filtros

Principalmente em dispositivos embarcados e casos de sinais multidimensionais com uma grande quantidade de amostras, um custo em memória de ordem $\Theta\left(n^2\right)$ para o cálculo de DWTs usando operações matriciais pode ser proibitivo. Esta seção descreve como uma DWT pode ser escrita no formato de um Banco de Filtros.

Segundo Salomon (2006), um filtro é um operador linear definido de acordo com seus coeficientes $h\left(0\right), h\left(1\right), h\left(2\right), ..., h\left(N-1\right)$. Um filtro com N coeficientes é chamado de um filtro com N *taps*. Ainda de acordo com Salomon (2006), um filtro pode ser aplicado a um sinal digital de entrada x e produzir um sinal digital de saída y de acordo com a Equação 4.22, na qual o operador $\star$ indica uma convolução:

$$y\left[n\right] = \sum_{k} h\left[k\right] x\left[n-k\right] = h \star x. \tag{4.22}$$

Os vetores $x[\tau]$, $h[\tau]$ e $y[\tau]$ são vistos como funções discretas temporais com infinitos valores, nas quais a amostra central representa o tempo $\tau = 0$. Na prática, as entradas são sempre finitas, então o vetor infinito x terá apenas um número finito de elementos não nulos (Salomon, 2006) que precisam ser armazenados.

Um melhor entendimento do comportamento de um filtro linear pode ser obtido ao

se considerar a simples entrada $x = [..., 0, 0, 1, 0, 0, ...]$. Essa entrada é zero em todos os tempos, exceto em $\tau = 0$. Ela é chamada de pulso unitário ou impulso unitário (Salomon, 2006). Nota-se que $y[n] = h[n]\,x[0] = h[n]$ para essa entrada. A saída $y[n] = h[n]$ no tempo $\tau = n$ é a resposta no tempo n ao impulso $x[0] = 1$. Como $h[i]$ é finito, esse filtro é uma Resposta de Impulso Finita (*Finite Impulse Response* – FIR).

Um Banco de Filtros de Análise (*Analysis Filter Bank* – AFB) é definido, então, como uma coleção de N filtros $H = h_0, h_1, ..., h_{N-1}$ que, aplicados a um sinal de entrada $x[n]$ com n amostras, produzem N saídas $Y = y_0[n], y_1[n], ..., y_{N-1}[n]$, cada uma com n amostras. Um Banco de Filtros de Síntese (*Synthesis Filter Bank* – SFB), analogamente, é um Banco de Filtros composto por M filtros $F = f_0, f_1, ..., f_{M-1}$ que – aplicados a um conjunto de entradas $X = x_0[m], x_1[m], ..., x_{M-1}[m]$, cada uma com m amostras – gera uma única saída $y[m]$, também com m amostras. A Figura 4.5 demonstra o funcionamento de um AFB e a Figura 4.6 exemplifica um SFB.

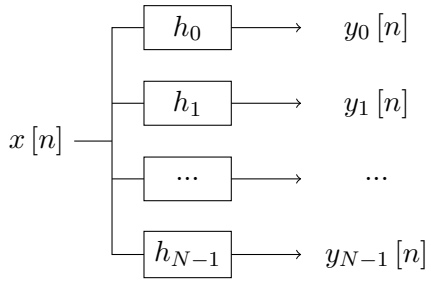


Figura 4.5: Exemplo de um AFB com N filtros.

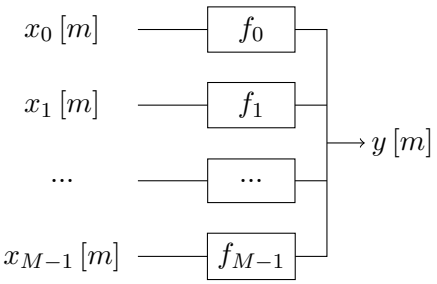


Figura 4.6: Exemplo de um SFB com M filtros.

No escopo das DHTs, os AFBs e os SFBs são compostos por dois filtros h_0 e h_1. Os AFBs incorporam passos de subamostragem por um fator 2 ($\downarrow 2$) após as operações de convolução, enquanto os SFBs executam passos de interpolação por um fator 2 ($\uparrow 2$) após seu cálculo. O filtro h_0 no AFB de uma DWT é um filtro passa-baixa, ou seja, ele deixa apenas as frequências de baixo valor serem expressas no sinal de saída $y_0[n]$, que, seguindo a convenção das últimas seções, chamar-se-á de c (*coarse*). O filtro h_1, por sua vez, permite a passagem apenas das altas frequências para o sinal $y_1[n]$, chamado de d (*detail*). Os filtros usados no AFB de uma DHT ortonormal são $h_0 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$ e $h_1 = \left(-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$, os quais operam sobre o sinal de entrada e produzem respectivamente os coeficientes que compõem c e d. A Figura 4.7 demonstra um AFB de uma DHT.

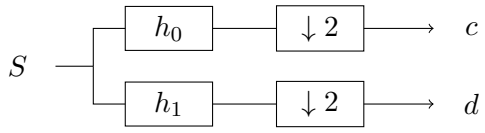


Figura 4.7: Exemplo de um AFB executando o cálculo de um passo de uma DWT.

Os vetores c e d podem passar por uma etapa de codificação de entropia (ver Seção 4.3) e serem armazenados num arquivo ou transmitidos por um canal de informação que tenha uma operação de IDWT incorporada à sua recepção. Os filtros usados no SFB de uma IDHT ortonormal são $f_0 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$ e $f_1 = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$. f_0 age sobre os coeficientes de c e f_1 age sobre os coeficientes de d, compondo o sinal reconstruído S. Um passo de uma operação de IDWT é modelado usando, além dos filtros, dois passos de interpolação, como mostra a Figura 4.8.

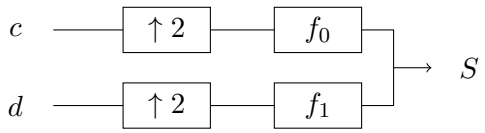


Figura 4.8: Exemplo de um SFB executando o cálculo de um passo de uma IDWT.

As operações de interpolação num SFB de uma IDWT acontecem da forma mais primitiva possível, ou seja, inserindo $N/2$ coeficientes de valor zero entre os $N/2$ coefi-

cientes do sinal, que, somados, totalizam o tamanho n do sinal antes da transformação. Como mostrado por Salomon (2006), as operações de subamostragem e interpolação de um sinal y são definidas nas Equações 4.23 e 4.24.

$$(\downarrow y) = (..., y(-4), y(-2), y(0), y(2), y(4), ...), \tag{4.23}$$

$$(\uparrow y) = (..., y(-4), 0, y(-2), 0, y(0), 0, y(2), 0, y(4), 0, ...). \tag{4.24}$$

A subamostragem, porém, causa perda de dados potencialmente importantes para a reconstrução exata da mensagem, de modo que usar simplesmente uma interpolação não compensa essa perda. Esse fator limita a escolha de filtros a certas categorias que minimizem a perda de dados causada pela subamostragem, preservando a informação contida na mensagem. Uma característica que é comumente usada na construção de bons filtros é a ortogonalidade (Salomon, 2006). Outra categoria de filtros usada é o conjunto de filtros biortogonais, os quais também podem resultar numa reconstrução perfeita da mensagem S.

Como foi introduzido na Seção 4.1.2, DWTs geralmente são operações realizadas em vários níveis diferentes, transformando hierarquicamente os coeficientes *coarse* restantes. Uma DWT, então, é representada por um conjunto de AFBs que transferem a saída c de um AFB para outro AFB semelhante, o que pode ser visto na Figura 4.9. Analogamente, uma IDWT pode ser calculada por um conjunto de SFBs organizados hierarquicamente, como mostrado na Figura 4.10. Como cada nó de uma árvore de AFBs produz metade do número de saídas que o seu predecessor, essa árvore é chamada de Árvore Logarítmica (Salomon, 2006).

As regras completas de derivação para definir os filtros ortogonais para o cálculo de DWTs, usando AFBs e SFBs, são matematicamente densas e extensas, podendo ser encontradas em Daubechies (1988). No entanto os valores das respostas aos impulsos

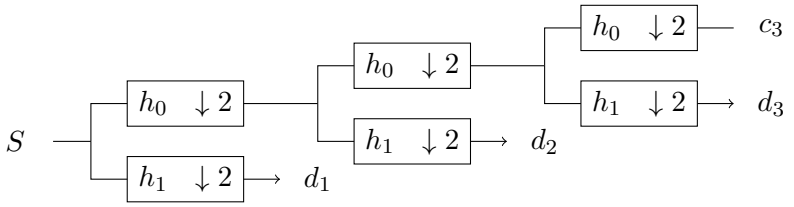


Figura 4.9: Exemplo de um AFB executando o cálculo de três passos de uma DWT.

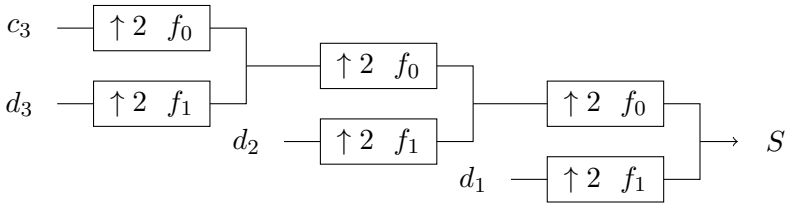


Figura 4.10: Exemplo de um SFB executando o cálculo de três passos de uma IDWT.

unitários de h_0 pré-calculados para alguns filtros ortogonais podem ser vistos em Daubechies (1988), Cohen et al. (1992) e Salomon (2006). É importante ressaltar que os coeficientes de h_0 são suficientes para calcular os filtros h_1, f_0 e f_1 – caso a base *wavelet* em questão seja ortogonal – como demonstrado pela Equação 4.25:

$$\begin{aligned} h_1 &= -h_0\left(N-1\right), ..., h_0\left(2\right), -h_0\left(1\right), h_0\left(0\right), \\ f_0 &= h_0\left(N-1\right), ..., h_0\left(2\right), h_0\left(1\right), h_0\left(0\right), \\ f_1 &= h_0\left(0\right), -h_0\left(1\right), h_0\left(2\right), ..., -h_0\left(N-1\right), \\ &\text{para } N \text{ par.} \end{aligned} \tag{4.25}$$

Todas as transformadas *wavelet* que foram apresentadas até esse momento são do tipo ortogonais. Isso implica que essas DWTs usam modificações de um filtro de entrada para gerar os outros filtros que compõem o AFB e o SFB. As *wavelets* biortogonais, por sua vez, não são ligadas a um único conjunto de coeficientes que forma os Bancos de Filtros, mas, sim, a dois. A transformada *wavelet* biortogonal 9/7 (Cohen et al., 1992), por exemplo, usa um filtro passa-baixa com 9 coeficientes e um filtro passa-alta com 7 coeficientes para gerar, respectivamente, os sinais transformados *coarse* e *detail*.

Esses tipos de filtros ganharam popularidade especialmente com a inclusão das bases biortogonais 9/7 e 5/3 nas variações *lossy* e *lossless* do compressor de imagens JPEG 2000 (JPEG 2000 Organization, 2000). A Figura 4.11 demonstra um nível de uma transformada *wavelet* biortogonal que usa os filtros H_0, H_1, F_0 e F_1 como bases.

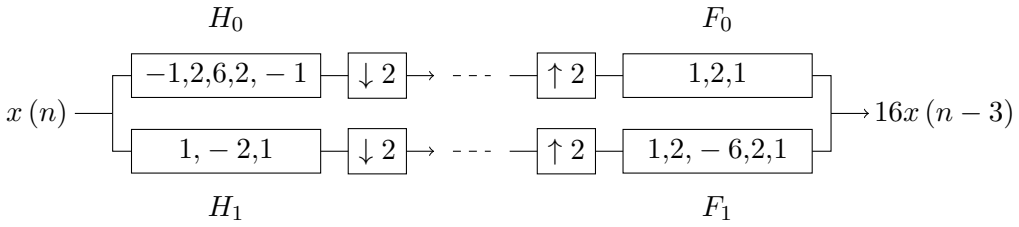


Figura 4.11: Exemplo de um passo de transformada *wavelet* biortogonal. Fonte: Salomon (2006).

É notável a semelhança do filtro H_0 com o F_1 e do filtro H_1 com o F_0. Isso se dá porque, em *wavelets* biortogonais, o filtro passa-alta da transformada direta gera o filtro passa-baixa da transformada inversa, enquanto o filtro passa-baixa da transformada direta gera o filtro passa-alta da transformada inversa.

4.1.2.4 *Lifting Scheme*

O método conhecido como *Lifting Scheme* (Stollnitz et al., 1996; Sweldens e Schröder, 1996) se propõe a transformar o pesado cálculo de DWTs usando matrizes – apresentado na Seção 4.1.2.2 – em simples operações usando Bancos de Filtros e subamostragens. A diferença para a aplicação normal de Bancos de Filtros está na forma na qual o *Lifting Scheme* simplifica o cálculo de DWTs, resumindo-as a um simples esquema de três passos executados *in place* – ou seja, sem necessitar de posições de memória adicionais. Dessa forma, apenas os coeficientes que não serão descartados após a subamostragem são calculados, o que diminui a quantidade de operações numéricas necessárias para o cálculo da DWT pela metade. Além disso, essa técnica possibilita a criação de esquemas gráficos de fácil entendimento e uma forma simples de implementação de *wavelets* para

compressão *lossless*, como explanado na Seção 4.1.2.5. Como afirmam Calderbank et al. (1998), o *Lifting* é uma técnica flexível que vem sendo usada em diferentes cenários, como os de simplificar a construção e implementação de *wavelets* tradicionais (Sweldens, 1996), *wavelets* de segunda geração (Sweldens, 1998) e *wavelets* esféricas (Schröder e Sweldens, 1995).

Novamente, utilizar-se-á a DHT para exemplificar o cálculo de uma DWT usando o *Lifting Scheme*, mas, dessa vez, será tomada como base a versão não normalizada da transformada. Sabe-se que os coeficientes $c_1[l]$ e $d_1[l]$ são calculados usando, respectivamente, as médias e diferenças do par de coeficientes $c_0[2l]$ e $c_0[2l+1]$. A operação de transformação de uma DHT *in place* para dois coeficientes $a = s_0[2l+1]$ e $b = s_0[2l]$ é calculada de acordo com as Equação 4.26 e 4.27:

$$b = b - a, \tag{4.26}$$

$$a = a + \frac{b}{2}. \tag{4.27}$$

Assim, os dados são representados na forma intercalada:

$$S_1[2l] = (c_1[0], d_1[0], c_1[1], d_1[1], ..., c_1[l-1], d_1[l-1]). \tag{4.28}$$

Cada par de amostras transformadas agora está ordenado para ocupar o mesmo espaço do par de amostras original no sinal, o que propicia uma fácil visualização de como a transformada pode ser calculada *in place*. Posteriormente, por conveniência, é possível separar as amostras para que elas sejam organizadas na forma:

$$S_1[2l] = (c_1[0], c_1[1], ..., c_1[l-1], d_1[0], d_1[1], ..., d_1[l-1]). \tag{4.29}$$

Qualquer DWT calculada por meio do *Lifting Scheme* segue três operações: dividir (*split*), predizer (*predict*) e atualizar (*update*); executadas nessa ordem. Como explica

Salomon (2006), a operação de *split*, também chamada de Transformada *Wavelet* Preguiçosa (*Lazy Wavelet Transform* – LWT), é uma simples separação entre as amostras pares e ímpares do sinal. A LWT cria os conjuntos $even_{j-1}$ e odd_{j-1} a partir do sinal de entrada s_j, como mostrado na Equação 4.30, na qual cada sinal contém 2^j amostras, indo de 0 a $2^j - 1$:

$$(even_{j-1}, odd_{j-1}) = Split\,(s_j)\,. \tag{4.30}$$

A operação *predict* usa o conjunto par $even_{j-1}$ para prever o conjunto ímpar odd_{j-1}. Isso se baseia no fato de que cada valor $s_j\,[2l+1]$ no conjunto ímpar é adjacente ao valor correspondente $s_j\,[2l]$ no conjunto par. Portanto, os dois valores são correlacionados e um pode ser usado para calcular o outro (Salomon, 2006). O operador de predição (P) calcula as diferenças do sinal – ou seja, os coeficientes *detail* – e é definido como:

$$d_{j-1} = odd_{j-1} - P\,(even_{j-1}) => odd_{j-1} - = P\,(even_{j-1})\,. \tag{4.31}$$

O operador *update* (U) segue o operador *predict*, calculando os níveis *coarse* do sinal de acordo com a Equação 4.32:

$$s_{j-1} = even_{j-1} + U\,(d_{j-1}) => even_{j-1} + = U\,(odd_{j-1})\,. \tag{4.32}$$

As IDWTs são obtidas apenas revertendo a ordem das operações e remanejando os termos das Equação 4.30, 4.31 e 4.32, o que resulta nas operações *undo update* (UU), *undo predict* (UP) e *merge*, respectivamente dadas por:

$$even_{j-1} - = UU\,(odd_{j-1})\,, \tag{4.33}$$

$$odd_{j-1} + = UP\,(even_{j-1})\,, \tag{4.34}$$

$$s_j = Merge\,(odd_{j-1}, even_{j-1})\,. \tag{4.35}$$

De acordo com Calderbank et al. (1998), a implementação dos passos *predict* e *update* da DHT usando *Lifting* é dada pelas Equações 4.36 e 4.37:

$$d_1[l] = s_0[2l+1] - s_0[2l], \tag{4.36}$$

$$s_1[l] = s_0[2l] + \frac{d_1[l]}{2}. \tag{4.37}$$

Os passos *undo update* e *undo predict* de uma IDHT implementada de acordo com o *Lifting Scheme* são descritos por:

$$s_0[2l] = s_1[l] - \frac{d_1[l]}{2}, \tag{4.38}$$

$$s_0[2l+1] = d_1[l] + s_0[2l]. \tag{4.39}$$

O *Lifting Scheme* é normalmente apresentado no formato de diagramas esquemáticos, como mostrado nas Figuras 4.12 e 4.13. A Figura 4.12 demonstra um exemplo de *Lifting* de uma transformada *wavelet* direta – ou seja, uma DWT – enquanto a Figura 4.13 demonstra o esquema de processamento de uma IDWT. Essas figuras apresentam os diagramas dos esquemas de *Lifting* de *wavelets* mais simples, como a DHT. Transformadas mais complexas exigem passos adicionais de predição (P) e atualização (U), criando um diagrama hierárquico no qual a saída de um par de operações $\left(P^{(i)}, U^{(i)}\right)$ é passada como entrada para um outro par de operações $\left(P^{(i+1)}, U^{(i+1)}\right)$.

Além dos ganhos em *performance* computacional, da facilidade de implementação e da possibilidade de paralelização do cálculo de DWTs usando o *Lifting Scheme*, ele também desempenha um papel importante na implementação de *wavelets* com coeficientes inteiros. Tais implementações inteiras de *wavelets* são usadas com frequência para a compressão *lossless* de sinais e imagens, como será abordado na próxima seção.

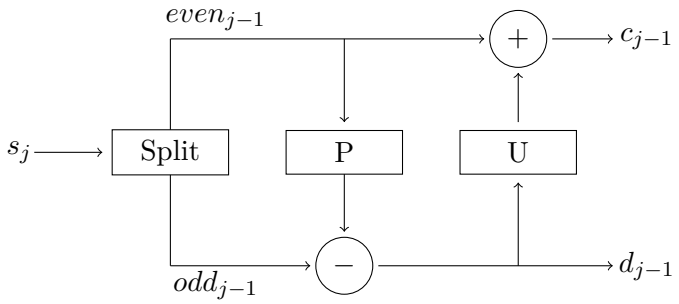


Figura 4.12: Diagrama de um passo de uma DWT usando o *Lifting Scheme*. Fonte: Salomon (2006).

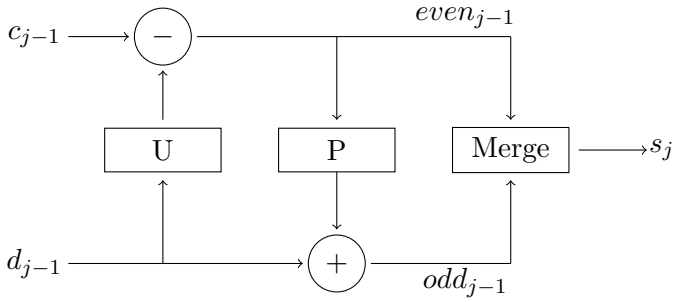


Figura 4.13: Diagrama de um passo de uma IDWT usando o *Lifting Scheme*. Fonte: Salomon (2006).

4.1.2.5 Transformadas *Wavelet* com Coeficientes Inteiros

Todas as formas de DWT apresentadas até agora geram versões aproximadas do sinal original após a etapa de reconstrução. Isso se dá porque, em geral, os valores dos coeficientes gerados pelas DWTs são arredondados para números inteiros numa etapa prévia à codificação de entropia, dada a ineficácia da maioria das técnicas de compressão sobre números de ponto flutuante. Quando os sinais devem ser reconstruídos sem nenhuma perda de informação, é necessário seguir outra abordagem para a DWT. Assim como todas as técnicas de compressão *lossless*, os coeficientes gerados pela aplicação de Transformadas *Wavelet* de Coeficientes Inteiros (*Integer Wavelet Transforms* – IWT) (Calderbank et al., 1998) tendem a acumular uma maior quantidade de energia quando comparados com métodos *lossy*, implicando em mensagens finais que precisam de uma

maior quantidade de *bits*.

Implementações de DWTs não normalizadas usando o *Lifting Scheme* podem ser facilmente modificadas para realizarem o cálculo das respectivas IWTs dessas transformadas com uma quantidade mínima de mudanças. Em geral, apenas uma operação de arredondamento para o valor inteiro imediatamente menor que um certo número real – expressa matematicamente pelo operador *floor* ($\lfloor\ \rfloor$) – precisa ser adicionada para transformar uma DWT numa IWT. No exemplo da DHT, as transformações que resultam nos elementos *detail* e *coarse* são dadas respectivamente por:

$$d_1[l] = s_0[2l+1] - s_0[2l], \tag{4.40}$$

$$s_1[l] = s_0[2l] + \left\lfloor \frac{d_1[l]}{2} \right\rfloor. \tag{4.41}$$

Como afirmam Calderbank et al. (1998), à primeira vista o arredondamento na definição de $s_1[l]$ parece descartar informação, porém a soma e a diferença de dois inteiros são ambos valores pares ou valores ímpares. É possível, então, omitir o último *bit* da soma – o qual indica a paridade – já que ele é igual ao último *bit* da diferença. Essa propriedade confere a característica da reversibilidade às Equação 4.40 e 4.41. A IDHT inteira, igualmente a qualquer outra transformação inversa de IWT, é calculada invertendo-se a ordem das operações e remanejando as variáveis das equações, o que resulta em:

$$s_0[2l] = s_1[l] - \left\lfloor \frac{d_1[l]}{2} \right\rfloor, \tag{4.42}$$

$$s_0[2l+1] = d_1[l] + s_0[2l]. \tag{4.43}$$

4.2 Quantização Escalar

Muitos sinais de interesse são, por natureza, de amplitude e tempo contínuos. Os conversores A/D, através das operações de amostragem e quantização, permitem a

representação discreta destas formas de onda. Neste contexto, a quantização é o processo de transformar amplitudes analógicas em discretas (Jayant e Noll, 1990). A quantização, porém, pode ser usada em sinais naturalmente digitais ou que tenham sido digitalizados previamente. Tal operação pode ser usada para diminuir a quantidade de informação de um sinal digital dividindo as suas amostras já digitais por um número real, o qual é chamado de passo de quantização. Deve ser possível representar o sinal quantizado com menos *bits* que o sinal original, já que a faixa de valores possíveis é menor (Richardson, 2010).

4.2.1 Quantização Escalar

Uma operação de quantização de um valor x por um passo de quantização escalar q, resultando num valor quantizado $\hat{x}$, é vista na Equação 4.44, na qual o operador // representa uma divisão seguida por um arredondamento para o número inteiro mais próximo:

$$\hat{x} = x//q. \tag{4.44}$$

A Equação 4.45 mostra uma operação de dequantização, na qual o valor quantizado $\hat{x}$ é dequantizado pelo passo de quantização q, gerando o valor aproximado $\tilde{x}$:

$$\tilde{x} = \hat{x} \times q. \tag{4.45}$$

Alguns métodos de quantização também concentram a informação com o uso de um limiar, zerando todos os valores de magnitude insignificantes de acordo com esse valor. A essa técnica é dado o nome de quantização com zona-morta. Ratnakar (1997) expõe consideráveis ganhos de qualidade de reconstrução em imagens com o uso de quantização com zona-morta em comparação com a quantização normal para sinais com distribuições Laplacianas. A operação de quantização com zona-morta de um valor x com um passo de quantização q e um limiar (*threshold*) t é expressa pela Equação 4.46.

A fórmula usada para realizar a dequantização da quantização com zona-morta é a mesma da quantização sem zona-morta, demonstrada pela Equação 4.45:

$$\hat{x} = \begin{cases} 0, & \text{para } |x| < t, \\ x//q, & \text{caso contrário.} \end{cases} \tag{4.46}$$

A quantização deve ser utilizada de forma inteligente ou pode gerar erros que comprometem a qualidade da reconstrução de um sinal. Essa necessidade de um pré-processamento antes da quantização de sinais digitais pode ser vista mais claramente em imagens. Padrões de compressão *lossy* de imagens digitais, tais como o, JPEG (Wallace, 1991) e o JPEG 2000 (JPEG 2000 Organization, 2000) quantizam as imagens de forma que a qualidade perdida com a compressão seja aceitável, ou até imperceptível para o observador. Simplesmente realizar uma operação de quantização nos *pixels* de uma imagem resulta numa reconstrução com uma degradação visual altamente perceptível. A Figura 4.14 demonstra a degradação gradual da qualidade de uma imagem à medida que o passo de quantização aumenta para uma quantização escalar realizada no domínio do espaço.

4.2.2 Quantização Vetorial

Segundo Richardson (2010), de forma similar à quantização escalar, a quantização vetorial busca diminuir o domínio de valores que precisam ser codificados, eliminando parte da informação intrínseca do sinal original, o que resulta numa impossibilidade de reconstrução perfeita por uma etapa de dequantização. Uma quantização escalar mapeia uma amostra do sinal de entrada em um valor quantizado de saída, enquanto uma quantização vetorial mapeia um grupo de amostras de entrada – um vetor – em um grupo de valores quantizados. Em outras palavras, a quantização escalar leva em conta apenas um passo de quantização q para todas as suas amostras, enquanto a quantização

(a) (b)

(c) (d)

Figura 4.14: Imagem (a) original (8 *bits*/amostra). (b) quantizada com passo de quantização 8 (5 *bits*/amostra). (c) quantizada com passo de quantização 32 (3 *bits*/amostra). (d) quantizada com passo de quantização 128 (1 *bit*/amostra).

vetorial utiliza um vetor Q de tamanho N composto por uma sequência de coeficientes q_i, para $i = 0, 1, ..., N-1$. Dessa forma, uma sequência de valores do sinal original X composta por amostras x_i, para $i = 0, 1, ..., N-1$, após uma etapa de quantização vetorial resulta numa sequência de amostras $\hat{x}_i$, compondo o sinal quantizado $\hat{X}$. A Equação 4.47 demonstra como se dá a operação de quantização vetorial e a Equação 4.48

demonstra o cálculo usado no processo de dequantização vetorial:

$$\hat{x}_i = x_i // q_i, \tag{4.47}$$

$$\tilde{x}_i = \hat{x}_i \times q_i. \tag{4.48}$$

Analogamente, a quantização vetorial com zona-morta – ao invés de usar apenas um limiar – usa um vetor T de tamanho N composto de limiares. A Equação 4.49 define matematicamente a operação de quantização vetorial com zona-morta:

$$\hat{x}_i = \begin{cases} 0, & \text{para } |x_i| < t_i, \\ x_i // q_i, & \text{caso contrário.} \end{cases} \tag{4.49}$$

O uso da quantização vetorial se faz ideal quando os pesos das amostras em diferentes índices do sinal são diferentes. A alta correlação entre amostras no domínio do espaço ou do tempo impossibilita a aplicação da quantização vetorial nos domínios originais da maioria dos sinais. Transformadas como a DCT e a DWT, porém, apresentam uma variação na entropia de seus coeficientes de acordo com seu índice, já que essas transformações têm a propriedade de descorrelacionar os coeficientes transformados. Assim, como será explicado na Seção 4.4, um dos paradigmas de compressão com perdas de informação mais usados atualmente envolve uma etapa prévia de transformação seguida por uma operação de quantização, sendo muitas vezes escolhida a quantização vetorial.

Apesar da eficácia da quantização vetorial usada em conjunto com transformadas de domínio, seu cálculo é em geral oneroso devido à alta complexidade computacional envolvida na sua computação. Técnicas como as Quantizações por Aproximações Sucessivas – apresentadas na Seção 4.3.6 – além de serem codificadores de entropia, podem ser adaptadas para servirem como quantizadores vetoriais. A Seção 5.2.2 apresenta um

outro método de otimização para o cálculo da quantização vetorial baseado na teoria Taxa-Distorção, descrita na Seção 3.4.

4.3 Codificação de Entropia

Desde os primórdios do estudo da Teoria da Informação, diversos algoritmos de codificação de entropia foram formulados, alguns mais eficientes que outros. A eficácia de um algoritmo de codificação pode ser medida analisando-se o tamanho médio das palavras-código por ele geradas e comparando esse valor com a entropia da mensagem.

Os algoritmos clássicos de Codificação de Entropia propostos ainda nas décadas de 1940 a 1960 são a Codificação de Shannon-Fano (Fano, 1949), a Árvore de Huffman (Huffman, 1952) e o codificador de Golomb (Golomb, 1966). Diversas variações desses algoritmos foram propostas durante as décadas subsequentes, com destaque para a Codificação de Golomb-Rice (Rice, 1979), que possibilitou uma computação mais rápida de códigos de Golomb com pouca perda de eficácia de compressão. O outro grande passo na área de codificação de entropia subsequente à Codificação de Golomb só seria dado na década de 1990, quando Bell et al. (1990) propuseram o codificador aritmético. Esse método consegue codificar símbolos na entropia da informação em praticamente todos os casos, já que elimina a limitação do codificador de Huffman de apenas codificar símbolos com tamanhos de *bits* inteiros.

As próximas subseções tratarão da descrição dos principais métodos de codificação de entropia encontrados na literatura e usados nesse trabalho.

4.3.1 Codificação de Huffman

Huffman (1952) apresentou um método ideal de codificação com palavras de tamanhos variáveis para códigos com números inteiros de *bits*. A obtenção do código de tamanho variável em um codificador de Huffman é baseada na criação de uma árvore binária

ordenada pelas probabilidades dos símbolos emitidos pela fonte de informação seguindo um algoritmo guloso. Com o aumento da probabilidade de um símbolo – ou seja, quanto mais perto da raiz da árvore esse símbolo estiver – menos *bits* devem ser atribuídos a ele. Esse método, apesar de facilmente adaptável para um ambiente computacional, pode ser bastante caro dependendo da quantidade N de símbolos no alfabeto que podem ser emitidos pela fonte de informação, já que uma ordenação desses valores deve ser feita entre cada intervalo I de iterações do algoritmo. Apesar de produzir uma eficácia de compressão razoável a um custo computacional aceitável, o codificador de Huffman é limitado por poder apenas escrever palavras com *bits* inteiros. Dessa forma, esse codificador consegue alcançar um valor próximo da entropia da mensagem apenas nos casos em que as probabilidades forem potências negativas na base 2, ou seja, frações do tipo $1/2^n$, tal que $n \in \mathbb{N}$.

Para gerar o código C_i de um símbolo i numa iteração de um algoritmo de Huffman semiadaptativo, os seguintes passos devem ser seguidos:

1. Executar uma etapa de ordenação decrescente sobre o conjunto de símbolos S de tamanho N de acordo com a frequência que eles apareçam, ou seja, os símbolos mais frequentes são alocados no início do vetor;

2. Substituir no nível superior da árvore os nós K_{N-1} e K_{N-2} correspondentes aos dois símbolos S_{N-1} e S_{N-2} de menor probabilidade por um só nó K'_{N-2}, criando uma hierarquia de forma que K'_{N-2} seja o nó pai de K_{N-1} e K_{N-2}. K_{N-2} deve ser o nó da esquerda e K_{N-1} o nó da direita;

3. Testar se há mais que um único nó no nível superior da árvore. Se houver, voltar para o passo 1 considerando agora apenas os nós presentes no nível superior da árvore. Se não houver, seguir para o passo 4;

4. Partindo do nó K_i do símbolo i que se deseja codificar, subir iterativamente na árvore até chegar à raiz. Se o nó filho na iteração atual for o filho esquerdo do nó

pai, concatenar o *bit* 1 ao código C_i do símbolo i. Caso contrário, concatenar o *bit* 0 a C_i;

5. Executar uma operação de inversão nos *bits* do código C_i, gerando o código C_i';

6. Mandar o código C_i' para o *output* do codificador;

7. Atualizar o nó K_i contendo a probabilidade do símbolo i, já que ele foi codificado mais uma vez.

No primeiro passo do algoritmo é possível definir diversas políticas de desempate para ordenar símbolos com probabilidades iguais, desde que as mesmas políticas sejam seguidas pelo decodificador.

A Figura 4.15 demonstra duas árvores construídas pelo algoritmo de Huffman em etapas diferentes da codificação da mensagem "BABCABBCBABBADBD".

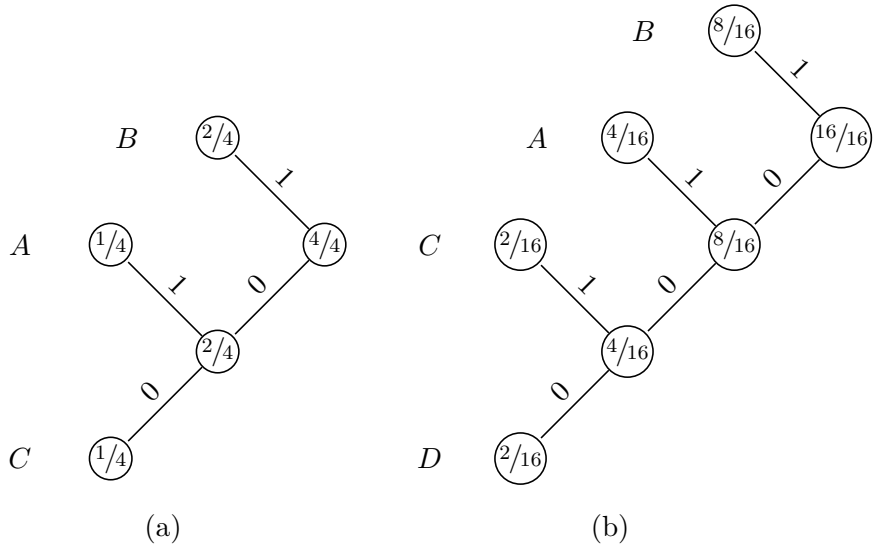


Figura 4.15: Árvore de Huffman após a codificação dos (a) 4 primeiros símbolos da mensagem "BABCABBCBABBADBD". (b) 16 símbolos da mensagem "BABCABBCBABBADBD".

4.3.2 Codificação de Golomb

Uma sequência na qual a razão entre elementos consecutivos é uma constante q é chamada de geométrica. Uma sequência desse tipo tem elementos a, aq, aq^2, aq^3,... (Salomon, 2006). Segundo Salomon (2006), Códigos de Golomb (Golomb, 1966) são os melhores códigos de prefixo para a compressão de dados distribuídos geometricamente e Codificações por Comprimento de Sequência (*Run Length Encoding* – RLE) geram distribuições estatísticas naturalmente geométricas. A probabilidade ${p_v}^n$ de que uma fonte de informação gere uma sequência de n valores v iguais que têm, cada um, a probabilidade p é dada pela Equação 4.50:

$${p_v}^n = p^n. \tag{4.50}$$

É perceptível, então, que a Equação 4.50 gera uma distribuição geométrica com razão p. Nota-se que quando a probabilidade p é alta, ${p_v}^n$ decresce vagarosamente à medida que n aumenta, característica intrínseca às distribuições geométricas. RLEs e métodos baseados em Códigos de Golomb, então, são aplicados em conjunto, principalmente em dados previamente quantizados que contam com uma grande quantidade de coeficientes de valor nulo.

Códigos de Golomb são derivados de Códigos de Huffman para distribuições estatísticas geométricas, gerando os mesmos resultados para esse tipo de dados, mas seu cálculo é muito mais eficiente devido à alta complexidade exigida pela construção de árvores de Huffman. Uma baixa complexidade é desejável principalmente para sistemas que exijam uma maior eficiência computacional, como codificadores de sinais em tempo real. Muitas vezes também não se sabe de antemão a quantidade de símbolos para os quais devem ser gerados códigos próprios por não se conhecer a variabilidade da fonte de informação, o que impossibilita a criação de uma tabela de códigos pelo algoritmo de Huffman para ser usada durante toda a execução do codificador. Um codificador

de Golomb, por outro lado, gera códigos para tamanhos de símbolos de entrada potencialmente infinitos, ou seja, o codificador gerará um código para qualquer palavra independentemente de sua quantidade de *bits*.

Um codificador de Golomb recebe um parâmetro m, que pode ser calculado de acordo com $p\,(0)$. Como explica Salomon (2006), para valores pequenos de m, os Códigos de Golomb começam curtos e aumentam rapidamente de tamanho. Eles são apropriados para RLEs em casos nos quais a probabilidade $p\,(0)$ de zero é pequena, implicando em poucas sequências longas. Para valores maiores de m, os códigos iniciais (para $n = 1, 2, ...$) são longos, mas seus tamanhos aumentam vagarosamente. Tais códigos fazem sentido para RLEs nas quais $p\,(0)$ é grande, resultando em sequências longas.

A primeira etapa para o cálculo do Código Golomb para o inteiro não-negativo n é computar o quociente q, o resto r e o valor c, dados, respectivamente, pelas Equações 4.51, 4.52 e 4.53:

$$q = \left\lfloor \frac{n}{m} \right\rfloor, \tag{4.51}$$

$$r = n - qm, \tag{4.52}$$

$$c = \lceil \log_2 m \rceil, \tag{4.53}$$

nas quais o operador *ceil* ($\lceil\ \rceil$) representa o arredondamento para o valor inteiro imediatamente maior que um número real.

Posteriormente o valor de q é codificado de acordo com um código unário, ou seja, 11...10, tal que o número de 1's seja igual a q; formando a primeira parte do código de Golomb referente ao valor n. A segunda parte do código é concatenada à primeira parte e corresponde ao valor do resto da divisão (r) codificado com uma quantidade de *bits* c', tal que:

$$c' = \begin{cases} c-1, & \text{caso} \quad r < 2^c - m, \\ c, & \text{caso contrário.} \end{cases} \tag{4.54}$$

A eficácia de compressão de um codificador de Golomb é obtida, então, conseguindo um balanceamento entre os tamanhos dos códigos gerados pelo quociente da divisão q – codificado de forma unária – e pelo resto da divisão r – codificado em binário com a quantidade de *bits* c'. O valor ótimo de m, logo, é dependente do valor de $p(0)$, já que a probabilidade de ocorrência de um *bit* zero influencia no tamanho médio dos códigos que devem ser utilizados para a codificação da RLE. Como demonstrado por Gallager e van Voorhis (1975), a relação entre $p(0)$ e m é dada pela desigualdade da Equação 4.55:

$$p(0)^m + p(0)^{m+1} \leq 1 < p(0)^m + p(0)^{m-1}. \tag{4.55}$$

A desigualdade na Equação 4.55, ainda de acordo com Gallager e van Voorhis (1975), tem apenas uma solução para o inteiro m, de forma que pode ser manipulada para dar origem à igualdade 4.56:

$$m = \left\lceil -\frac{\log_2(1 + p(0))}{\log_2 p(0)} \right\rceil. \tag{4.56}$$

As Tabelas 4.1, 4.2 e 4.3 mostram Códigos de Golomb para $m = 1$, $m = 4$ e $m = 14$.

Tabela 4.1: Códigos gerados por um codificador de Golomb para $m = 1$ e $n = \{0, 1, ..., 9, 10\}$. Fonte: Golomb (1966).

n	$p(n)$	Código
0	$1/2$	0
1	$1/4$	10
2	$1/8$	110
3	$1/16$	1110
4	$1/32$	11110
5	$1/64$	111110
6	$1/128$	1111110
7	$1/256$	11111110
8	$1/512$	111111110
9	$1/1024$	1111111110
10	$1/2048$	11111111110

Tabela 4.2: Códigos gerados por um codificador de Golomb para $m = 4$ e $n = \{0, 1, ..., 9, 10\}$. Fonte: Golomb (1966).

n	$p(n)$	Código
0	0,151	000
1	0,128	001
2	0,109	010
3	0,092	011
4	0,078	1000
5	0,066	1001
6	0,056	1010
7	0,048	1011
8	0,040	11000
9	0,034	11001
10	0,029	11010

Tabela 4.3: Códigos gerados por um codificador de Golomb para $m = 14$ e $n = \{0, 1, ..., 16, 17\}$. Fonte: Golomb (1966).

n	Código
0	0000
1	0001
2	00100
3	00101
4	00110
5	00111
6	01000
7	01001
8	01010
9	01011
10	01100
11	01101
12	01110
13	01111
14	10000
15	10001
16	100100
17	100101

Uma variação possível dos codificadores de Golomb consiste em apenas executar as RLEs sequencialmente nos planos de *bits* do sinal, ao invés de fazê-lo amostra por amostra. Esse codificador modificado de Golomb – daqui para a frente chamado de BitGol – se aproveita da correlação entre os diferentes planos de *bits* de amostras vizinhas, obtendo resultados melhores que codificadores de Golomb normais em alguns casos.

4.3.3 Codificação Aritmética

Bell et al. (1990) resolveram o problema da codificação de entropia usando palavras-código de tamanho inteiro com a codificação aritmética. A codificação aritmética permite codificar símbolos que necessitem de números não inteiros de *bits*, de acordo com suas probabilidades. Essa técnica atinge a entropia da mensagem não apenas nos casos das potências negativas na base 2 – como as codificações com palavras-código inteiras – mas em qualquer caso de probabilidade p no intervalo $[0, 1)$. A codificação aritmética supera o problema da atribuição de códigos inteiros para os símbolos individuais da mensagem atribuindo um código (normalmente longo) para o arquivo de entrada inteiro (Salomon, 2006). Esse código único para o sinal de entrada está suscetível a uma perda máxima de 1 *bit* na sua codificação, o que na maioria dos casos é insignificante. O que é codificado, então, é a probabilidade de que o arquivo de entrada K com N símbolos tenha exatamente essa concatenação de símbolos.

O algoritmo de um codificador aritmético funciona basicamente seguindo três passos:

1. Comece definindo o intervalo atual como $[0, 1)$;

2. Repita os próximos dois passos para cada símbolo S_i no fluxo de entrada:

   (a) Divida o intervalo atual em subintervalos com tamanhos proporcionais ao conjunto de probabilidades p dos símbolos;

(b) Selecione o subintervalo p_i correspondente a S_i e defina-o como o intervalo atual;

3. Quando o fluxo de entrada inteiro for processado dessa forma, a saída deve ser qualquer número que identifique unicamente o intervalo atual – ou seja, qualquer número dentro do intervalo atual.

A codificação aritmética é limitada, porém, pela finidade da representação de números decimais nos computadores e pela alta carga computacional envolvida nas operações aritméticas sucessivas em números de ponto flutuante. Usa-se na prática uma aproximação discreta da codificação aritmética com números inteiros e com intervalos finitos entre os limites inferior e superior, o que permite o seu cálculo acelerado ao custo de uma compressão não tão próxima da entropia quando o caso contínuo poderia prover. Todavia, essa diferença entre a aproximação discreta e a codificação aritmética contínua proposta por Bell et al. (1990) é praticamente imperceptível, não afetando consideravelmente a eficácia de compactação de dados de um compressor que utilize a versão discreta para codificar seus símbolos.

4.3.4 Predição por Casamento Parcial

Os compressores de dados *lossless* normalmente eliminam a redundância de um sinal aproveitando-se de particularidades na sua distribuição estatística. Em alguns casos – como no algoritmo de Predição por Casamento Parcial (*Prediction by Partial Matching* – PPM), Árvore de Huffman e Codificação de Golomb – essa modelagem estatística é calculada diretamente. Em outros métodos de compressão – como no dos compressores baseados em dicionários – criam-se abstrações para a representação dessas distribuições probabilísticas dos dados que permitem um menor custo computacional e uma maior facilidade na etapa de implementação.

Dentre os métodos de compressão baseados na modelagem estatística direta, um dos

mais usados atualmente é o PPM. Grande parte das aplicações comerciais de compressão de dados *lossless* para arquivos de natureza geral são baseadas em modelagens feitas usando PPMs, já que os computadores modernos conseguem lidar razoavelmente bem com a complexidade computacional desse algoritmo. Um PPM pode ser definido como um codificador aritmético alimentado pela saída de um modelo estatístico adaptativo contextual parametrizável.

Proposto inicialmente por Cleary e Witten (1984), o PPM atualmente pode ser encontrado em diversas variações de seu algoritmo original. O *Fast* PPM, por exemplo, tenta isolar as propriedades do PPM que contribuem apenas marginalmente para o desempenho da compressão e trocá-las por aproximações (Salomon, 2006), o que resulta num algoritmo mais rápido, mas que gera uma taxa de compressão menor. Já o PPM* conta com um tamanho máximo de contexto ilimitado, possibilitando ganhos de compressão entre 5% e 6%, se comparado com o PPM-C – que é a implementação mais comum. O grande desafio do PPM* é tratar a potencial grande quantidade de símbolos de escape que pode ser gerada quando contextos muito grandes estão envolvidos, o que foi resolvido com um método de análise do determinismo da predição dos símbolos da mensagem. Outra variação do esquema do PPM – entitulada de Predição por Casamento Parcial Binário – é apresentada na Seção 4.3.5.

O tamanho de uma mensagem é determinado pelas suas características léxicas e sintáticas, de forma que certas partes da mensagem já são predefinidas pelas regras de criação às quais a fonte de informação está submetida. Essas regras de criação geram, muitas vezes, mensagens maiores do que o tamanho mínimo necessário para se representar as informações que se quer passar. O PPM se aproveita do contexto dos últimos símbolos lidos para calcular as probabilidades do próximo símbolo a ser codificado, o que resulta numa eliminação da redundância gerada pelas regras sintáticas da mensagem. Uma codificação gerada por um PPM, se parametrizada de forma correta, gera um tamanho médio de *bits* por símbolo muito próximo da entropia da mensagem.

Esses *bits* codificados representam os dados independentes da mensagem, ou seja, os dados que não estão presos por alguma regra sintática que modifique sua distribuição estatística. Dessa forma, os compressores de propósitos gerais baseados em PPMs são o estado da arte da compressão *lossless*.

Como apontado por Salomon (2006), o PPM muda para um contexto menor quando uma busca por um contexto maior resulta na probabilidade zero. Logo, um PPM começa com um contexto de ordem K e, caso não encontre o símbolo nesse contexto, procura iterativamente nos contextos menores $K - i$, tal que $1 \leq i \leq K$. Caso o símbolo não tenha sido encontrado quando $i = K$, ou seja, quando não foi levado em consideração nenhum contexto, o símbolo é codificado de acordo com um modelo de ignorância absoluta, ou seja, equiprobabilidade (chamado de $K = -1$). Assim que o símbolo é codificado, o PPM atualiza o modelo estatístico. A estrutura de dados que implementa um PPM pode ser entendida de forma abstrata como uma tabela, na qual as colunas representam os contextos de símbolos vistos. Por exemplo, para a mensagem "*abracadabra*", as Tabelas 4.4, 4.5, 4.6 e 4.7 representam os modelos de um PPM-C de contexto máximo $K = 2$ para o primeiro, segundo, terceiro e último símbolos lidos, respectivamente. É perceptível como a tabela de símbolos, contextos e probabilidades do PPM cresce rápido, logo uma estrutura de dados adequada deve ser usada.

Tabela 4.4: Tabela de um PPM-C na codificação da mensagem "a".

K=2			**K=1**			**K=0**		**K=-1**
Cont.	**Símb.**	**Prob**	**Cont.**	**Símb.**	**Prob.**	**Símb.**	**Prob.**	**Prob.**
						"a"	1/2	1/4
						esc.	1/2	

Tabela 4.5: Tabela de um PPM-C na codificação da mensagem "ab".

K=2			**K=1**			**K=0**		**K=-1**
Cont.	**Símb.**	**Prob.**	**Cont.**	**Símb.**	**Prob.**	**Símb.**	**Prob.**	**Prob.**
			"a"	"b"	1/2	"a"	1/4	1/3
				esc.	1/2	"b"	1/4	
						esc.	1/2	

Tabela 4.6: Tabela de um PPM-C na codificação da mensagem “abr”.

K=2			**K=1**			**K=0**		**K=-1**
Cont.	**Símb.**	**Prob.**	**Cont.**	**Símb.**	**Prob.**	**Símb.**	**Prob.**	**Prob.**
“ab”	“r”	1/2	“a”	“b”	1/2	“a”	1/6	1/2
	esc.	1/2		esc.	1/2	“b”	1/6	
			“b”	“r”	1/2	“r”	1/6	
				esc.	1/2	esc.	3/6	

Tabela 4.7: Tabela de um PPM-C na codificação da mensagem “abracadabra”.

K=2			**K=1**			**K=0**		**K=-1**
Cont.	**Símb.**	**Prob.**	**Cont.**	**Símb.**	**Prob.**	**Símb.**	**Prob.**	**Prob.**
“ab”	“r”	2/3	“a”	“b”‘	2/7	“a”	5/11	
	esc.	1/3		“c”	1/7	“b”	2/11	
“ac”	“a”	1/2		“d”	1/7	“c”	1/11	
	esc.	1/2		esc.	3/7	“d”	1/11	
“ad”	“a”	1/2	“b”	“r”	2/3	“r”	2/11	
	esc.	1/2		esc.	1/3			
“br”	“a”	2/3	“c”	“a”	1/2			
	esc.	1/3		esc.	1/2			
“ca”	“d”	1/2	“d”	“a”	1/2			
	esc.	1/2		esc.	1/2			
“da”	“b”	1/2	“r”	“a”	2/3			
	esc.	1/2		esc.	1/3			
“ra”	“c”	1/2						
	esc.	1/2						

4.3.5 PPM Binário por Planos de *Bits*

Uma variação do algoritmo do PPM otimizada para o uso no caso de poucos recursos computacionais é o PPM Binário (*Binary* PPM – BPPM) (Brasileiro e Cavalcanti, 2012). Esse caso especial do PPM trata os sinais como conjuntos de planos de *bits* independentes, permitindo que uma modelagem contextual seja executada para cada plano de *bits*. Além disso, o acesso linear necessário para procurar símbolos numa Estrutura de Dados de um PPM tradicional é substituído por um simples acesso aleatório num vetor pré-alocado de elementos num BPPM. O BPPM é especialmente adaptado para o uso em sistemas embarcados que contam com recursos de memória e processamento escassos, como é mostrado por Brasileiro e Cavalcanti (2012).

Brasileiro e Cavalcanti (2012) demonstram a eficiência do BPPM em comparação ao PPM em termos de quantidade de células de memória necessárias para a implementação em *hardware* de um esquema de compressão. Um BPPM com contexto igual a 3 ocupa apenas 30 posições de memória, enquanto um PPM tradicional com modelagem contextual de mesmo nível necessitaria de um total de 281543712968704 células de memória.

A abordagem binária do BPPM ainda pode ser melhorada com o uso de Códigos Gray (*Gray Code* – GC) – que também são computacionalmente baratos. Os GCs são uma codificação diferente para representar um certo número de *bits*, de forma que a diferença entre a representação de um número natural x qualquer para um outro número $x + 1$ ou $x - 1$ seja de apenas um *bit*, como mostrado na Tabela 4.8. Essa abordagem tende a aumentar a correlação entre os *bits* de amostras vizinhas em sinais contínuos, o que favorece a compressão. Como a implementação do BPPM usada nesse trabalho contém o cálculo de GCs, o uso desses dois algoritmos em conjunto será implicitamente referenciado a partir deste ponto apenas como BPPM.

Tabela 4.8: Transformação de um código binário tradicional para um GC de 4 *bits*.

Decimal	**Binário**	**Gray**	**Decimal**	**Binário**	**Gray**
0	*0b0000*	*0b0000*	8	*0b1000*	*0b1100*
1	*0b0001*	*0b0001*	9	*0b1001*	*0b1101*
2	*0b0010*	*0b0011*	10	*0b1010*	*0b1111*
3	*0b0011*	*0b0010*	11	*0b1011*	*0b1110*
4	*0b0100*	*0b0110*	12	*0b1100*	*0b1010*
5	*0b0101*	*0b0111*	13	*0b1101*	*0b1011*
6	*0b0110*	*0b0101*	14	*0b1110*	*0b1001*
7	*0b0111*	*0b0100*	15	*0b1111*	*0b1000*

4.3.6 Particionamento de Conjuntos em Árvores Hierárquicas

O método de Particionamento de Conjuntos em Árvores Hierárquicas (*Set Partitioning in Hierarchical Trees* – SPIHT) (Said e Pearlman, 1996) foi construído especificamente para codificar coeficientes transformados por *wavelets*, sendo uma modificação do algoritmo de Árvores Embarcadas de Transformadas *Wavelet* (*Embedded Zerotrees of Wavelet Transforms* – EZW) (Shapiro, 1993). O SPIHT se aproveita da correlação entre os vários níveis hierárquicos nos quais os coeficientes *wavelet* são organizados utilizando uma Árvore de Orientação Espacial (*Spatial Orientation Tree* – SOT) ou uma Árvore de Orientação Temporal (*Temporal Orientation Tree* – TOT). Essas árvores permitem que o SPIHT faça buscas por coeficientes significantes organizadas pelos níveis da DWT, acelerando e melhorando a eficiência do processo de codificação.

O SPIHT não codifica especificamente as amostras de um sinal, mas sim as posições de cada *bit* considerado significante em cada iteração do algoritmo. Os *bits* no SPIHT são escaneados numa ordem específica e predeterminada, de forma que as posições dos *bits* não precisam ser enviadas para o decodificador – o que necessitaria de uma grande quantidade de informação adicional. O que é na realidade codificado é a informação indicando a significância – ou não significância – de um certo coeficiente (ou conjunto de coeficientes) *wavelet* de acordo com um limiar que é variado nas iterações do SPIHT.

Como afirmam Rao e Yip (2010), transformadas *wavelet* são indicadas para codificações escaláveis (no domínio espacial ou temporal), facilitando a transmissão de dados em formato embarcado. Essa característica é definida como: se um codificador embarcado produz dois arquivos – um grande com M *bits* e outro pequeno de tamanho m – então o arquivo menor é igual aos primeiros m *bits* do arquivo maior (Salomon, 2006). Isso permite ao SPIHT finalizar a decodificação em qualquer ponto, podendo atender a diferentes demandas de qualidade com uma mesma codificação. Qualquer técnica que utilize essa codificação embarcada como um quantizador vetorial – permitindo que a taxa de distorção seja passada como parâmetro antes da codificação em

si – é classificada como uma Quantização por Aproximações Sucessivas (*Successive Approximation Quantization* – SAQ).

Apesar de fazerem parte de vários métodos do estado da arte da compressão de imagens, SAQs não são geralmente ótimas na perspectiva de taxa-distorção (da Silva et al., 2002). A Figura 4.16 exemplifica a não otimalidade dos métodos de SAQ usando as condições de Lloyd-Max (Gersho e Gray, 1992) para separar as áreas de codificação da função Gaussiana de acordo com o número de *bits*. Segundo da Silva et al. (2002), tomando-se uma sequência de números de uma distribuição normal $X \sim N(0,1)$, a descrição de 1 *bit* ótima para um valor localizado na função Gaussiana é aquela que especifica o sinal (positivo ou negativo) do valor a ser codificado – Figura 4.16a. A descrição ótima para um total de 2 *bits* é demonstrada na Figura 4.16b, na qual a subdivisão entre as áreas da função a serem codificadas é um subconjunto da divisão apresentada na Figura 4.16a. Até esse ponto a codificação é embarcada, já que todos os números que eram codificados na etapa prévia com o *bit* 0 agora são codificados com um código que começa com o *bit* 0. Porém, quando o terceiro *bit* é adicionado – Figura 4.16c – as divisões das áreas de codificação da função não são mais subconjuntos das divisões apresentadas na Figura 4.16b, com 2 *bits*. O valor -1, por exemplo, com 2 *bits* é codificado como 0b00 e com 3 *bits* produz a *string* binária 0b010.

O SPIHT, apesar da não otimalidade das SAQs, é um dos métodos de codificação embarcada mais sofisticados encontrados no estado da arte da compressão de dados, estando presente em uma vasta gama de estratégias de compressão envolvendo *wavelets*. O algoritmo do SPIHT conta com um conjunto de três listas para determinar quais coeficientes – ou conjuntos de coeficientes – *wavelet* são significantes para a iteração atual. Os elementos nessas listas indicam as posições dos coeficientes na TOT ou na SOT utilizada. Salomon (2006) e Said e Pearlman (1996) detalham a implementação do SPIHT para a compressão de sinais bidimensionais – por exemplo, imagens – enquanto Lu et al. (2000) descrevem a implementação em uma dimensão.

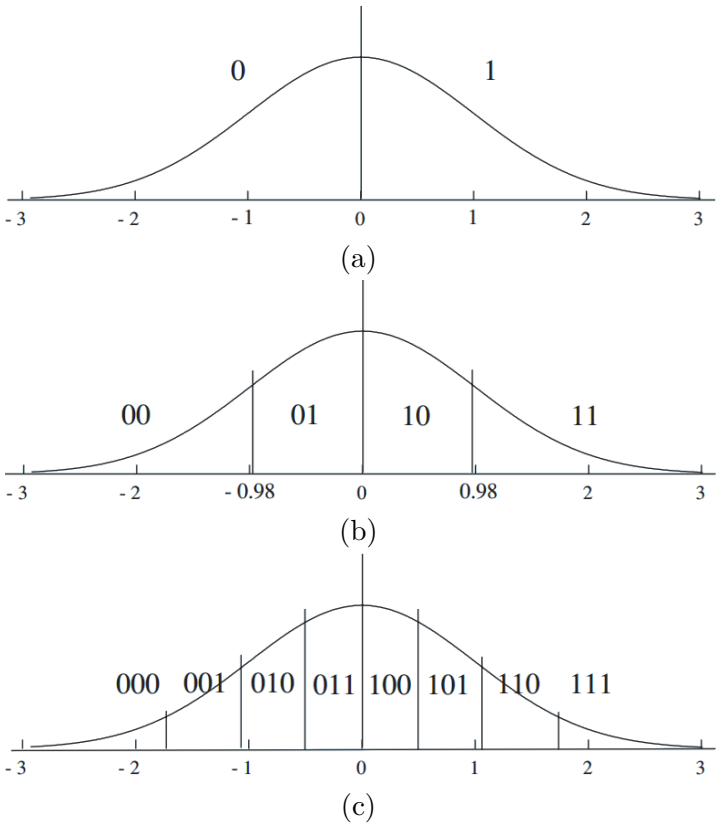


Figura 4.16: Codificando um valor aleatório numa fonte Gaussiana com (a) 1 *bit*. (b) 2 *bits*. (c) 3 *bits*. Fonte: da Silva et al. (2002).

4.4 O Paradigma Transformação-Quantização-Codificação

O paradigma de compressão Transformação-Quantização-Codificação (TQC), um dos mais usados nos compressores com perdas atuais, engloba três fases do processo de compressão *lossy* de uma mensagem.

Primeiro é realizada uma transformação na mensagem x que se deseja comprimir, deixando-a mais propensa a uma quantização com pouca perda de informações importantes para a reconstrução e concentrando a distribuição estatística dos dados. A transformação geralmente é realizada nos sub-blocos b_i de tamanho N, tal que $i = 0, 1, ..., M-1$ e $M = \frac{\text{tamanho}(x)}{N}$, que compõem a mensagem original – e não na mensagem inteira – dado que a complexidade computacional das transformadas pode

ser proibitivamente alta. Os blocos transformados serão denominados B_i. A quantização é a etapa que descarta a informação redundante e não importante para a reconstrução do sinal, deixando sua distribuição estatística mais densa e, consequentemente, diminuindo sua entropia. A quantização é realizada sobre os blocos transformados B_i, gerando os blocos quantizados $\hat{B}_i$. A codificação geralmente é uma etapa na qual se aplicam técnicas de compressão *lossless* nos blocos de amostras já quantizadas, gerando o sinal comprimido $\bar{x}$. Esse sinal pode ser transmitido por um canal de comunicação ou armazenado em algum tipo de memória não volátil.

Na etapa de decodificação do sinal $\bar{x}$, o descompressor segue os passos inversos aos seguidos pelo compressor: primeiro se decodificam os coeficientes do sinal $\bar{x}$; depois esses são dequantizados, gerando os blocos aproximados $\tilde{B}_i$; e, por fim, é aplicada a transformação inversa à usada na compressão. A descompressão recupera os blocos aproximados $\tilde{b}_i$ que compõem o sinal reconstruído $\tilde{x}$ no domínio do tempo ou espaço. A Figura 4.17 demonstra o funcionamento de um conjunto compressor/descompressor (*Compressor/Decompressor* – CODEC) que usa o paradigma TQC.

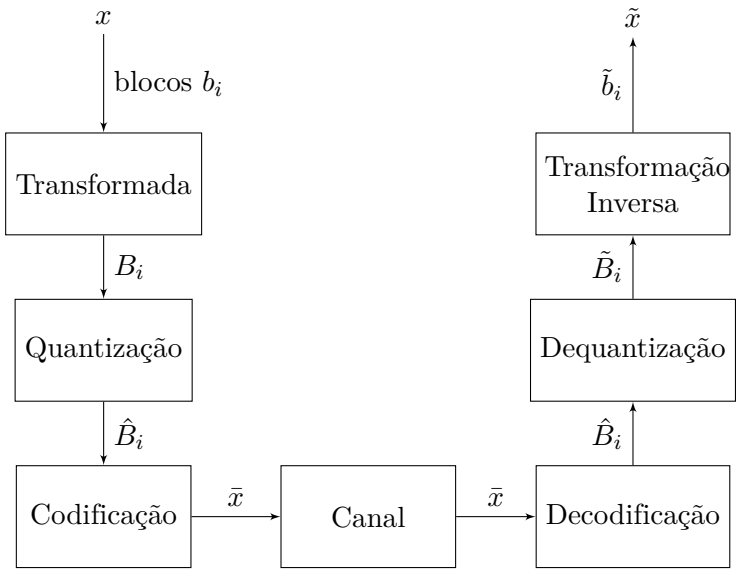


Figura 4.17: Diagrama em blocos de um CODEC baseado no paradigma TQC. Fonte: Batista (2002).

A eficácia do esquema em termos de taxa e distorção depende tanto da transformada quanto das estratégias de quantização e de codificação de entropia. Uma transformada apropriada gera coeficientes pouco correlacionados e concentra a maior parte da energia do bloco em um pequeno número de coeficientes (Batista, 2002; Britanak et al., 2006). Essa concentração de informação permite uma quantização mais intensa nos coeficientes que contêm menos informação, já que esses valores são menos importantes para a reconstrução do sinal. Os coeficientes transformados e quantizados são passados para o codificador, que se aproveita da concentração da informação para gerar modelos estatísticos compatíveis com a distribuição estatística dos dados. Tipicamente observa-se que, nos esquemas de compressão baseados em transformada e quantização, os blocos quantizados apresentam grandes sequências de coeficientes nulos, de maneira que a etapa final é muitas vezes uma combinação de RLE e codificação de entropia (Wallace, 1991).

4.5 Predição Linear

Em sinais nos quais os valores de amostras próximas são altamente correlacionados, métodos diferenciais baseados em predição linear podem transformar a distribuição da entropia da mensagem, favorecendo a compressão. Algumas DWTs incorporam esse efeito em seu cálculo por meio da adição de um ou mais passos de predição na sua computação. No entanto, as predições lineares podem ser calculadas diretamente nos domínios do espaço ou do tempo de forma menos custosa.

Esquemas de compactação de áudio como o ADPCM (*Adaptive Differential Pulse-Code Modulation*) (ITU-T, 1990) e o Shorten (Robinson, 1994) implementam esses métodos de predição visando reduzir a quantidade de informação necessária para codificar tais sinais. De forma parecida, codificadores de vídeo modernos como o H.264 (ITU-T264, 2002) e o H.265 (Sullivan et al., 2012) usam várias configurações de predi-

tores lineares – chamados preditores *intra-frame* – como pré-processamento para suas etapas de codificação baseadas no paradigma TQC. Os codificadores de vídeo ainda usam outra forma de predição, dessa vez codificando diferencialmente blocos de *pixels* de *frames* de vídeo vizinhos, o que permite realizar uma codificação diferencial sem propagação de erros devido à sua característica *lossy*.

Em geral, um preditor de ordem n computa um polinômio de ordem $n-1$ que passa pelos n pontos da amostra $s\left(t-n\right)$ até a amostra $s\left(t-1\right)$ e as extrapola para predizer $s\left(t\right)$ (Salomon, 2006). A Equação 4.57 define as aproximações $\hat{s}_0\left(t\right)$, $\hat{s}_1\left(t\right)$, $\hat{s}_2\left(t\right)$ e $\hat{s}_3\left(t\right)$ para a amostra $s\left(t\right)$ de acordo com preditores de ordem zero, um, dois e três:

$$\begin{aligned}
\hat{s}_0\left(t\right) &= 0 \\
\hat{s}_1\left(t\right) &= s\left(t-1\right) \\
\hat{s}_2\left(t\right) &= 2s\left(t-1\right) - s\left(t-2\right) \\
\hat{s}_3\left(t\right) &= 3s\left(t-1\right) - 3s\left(t-2\right) + s\left(t-3\right).
\end{aligned} \tag{4.57}$$

As predições resultam em aproximações para a amostra no índice atual t, possibilitando que o compressor codifique apenas o erro de predição de ordem n (e_n) de acordo com a Equação 4.58:

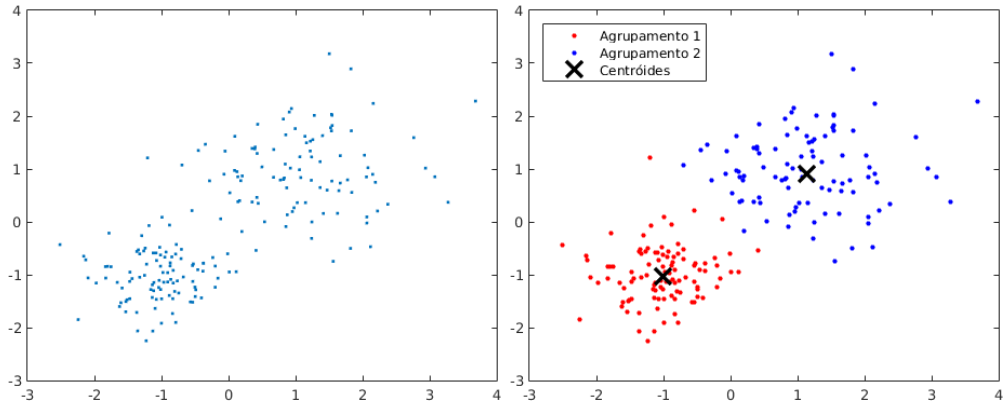
$$e_n\left(t\right) = s\left(t\right) - \hat{s}_n\left(t\right). \tag{4.58}$$

4.6 K-Médias

Como definem Witten et al. (2011), o K-Médias é um algoritmo de Agrupamento Iterativo Baseado em Distância (*Iterative Distance-Based Clustering* – IDBC). Em outras palavras, o K-Médias, assim como o algoritmo de Maximização de Expectativa (*Expectation Maximization* – EM) (Dempster et al., 1977) e o *Mean Shift* (Fukunaga e Hostetler, 1975), usam iterações consecutivas com a finalidade de realizar uma operação de agrupamento sobre um conjunto de dados. De acordo com Witten et al. (2011),

técnicas de agrupamento (*clustering*) são usadas quando não há nenhuma classe a ser predita, mas as instâncias estão divididas naturalmente em grupos distintos. Dessa forma, o K-Médias é dito uma técnica de classificação não-supervisionada.

Um exemplo de execução de um K-Médias sobre um espaço de atributos bidimensional pode ser visto na Figura 4.18. Nota-se que os agrupamentos naturais representados pelas amostras vermelhas e azuis foram separados corretamente pelo algoritmo, permitindo uma análise individual de cada conjunto de dados linearmente separável.



(a) (b)

Figura 4.18: Exemplo de K-Médias sobre um espaço de atributos bidimensional. (a) Amostras originais. (b) Amostras após o agrupamento com os centróides discriminados.

Como explanam Witten et al. (2011), algoritmo do K-Médias é composto dos seguintes passos:

1. Receber do usuário o número K de agrupamentos (*clusters*) nos quais as amostras serão divididas;

2. Escolher K pontos aleatoriamente no espaço de atributos e defini-los como centróides da primeira iteração do algoritmo;

3. Definir o centróide mais próximo de cada amostra por meio da distância euclidiana;

4. Agrupar as amostras dividindo-as por centróide;

5. Atualizar a posição dos centróides em cada agrupamento de acordo com a média das posições de todas as amostras nos seus respectivos *clusters*;

6. Caso as amostras de cada agrupamento não sejam alteradas após 2 iterações consecutivas, diz-se que o algoritmo convergiu e, portanto, é finalizado. Caso contrário, repetir o algoritmo começando do terceiro passo.

Como citado previamente, a DCT tende a apresentar desempenho ótimo de compressão para sinais estacionários Markov-1 quando o coeficiente de correlação entre amostras adjacentes tende a 1 (Rao e Yip, 2014). Dessa forma, visando aumentar a estacionariedade do sinal, o algoritmo do K-Médias pode ser aplicado previamente a uma etapa de quantização vetorial visando separar os blocos DCT em K grupos mais próximos de estacionários. Essa estratégia possibilita que sejam realizadas K etapas de quantização vetorial ótima, uma para cada tipo de bloco DCT.

4.7 Conclusão do Capítulo

Os Capítulos 3 e 4 apresentaram conceitos e técnicas comumente utilizadas na área de compressão de dados e processamento de sinais. O presente capítulo descreveu em detalhes as transformadas, as técnicas de pré- e pós-processamento e os codificadores de entropia utilizados nesse trabalho. O Capítulo 5 introduz a forma como essas técnicas foram organizados em métodos de compactação para os testes desse trabalho.

Capítulo 5

Metodologia

No decorrer desse trabalho, diversos métodos de compressão de dados e conceitos relacionados à teoria da informação e ao processamento de sinais foram apresentados. Este capítulo descreverá os compressores de dados *lossless* e *lossy* implementados com a finalidade de validar e verificar a eficácia das técnicas de compressão propostas. Todas as variações da aplicação de teste foram implementadas usando as linguagens C++ (Stroustrup, 1986) e Shell Script (Jargas, 2008). O banco de dados de sinais biológicos usado nos testes dos métodos propostos foi o MIT/BIH PSGDB (Ichimaru e Moody, 1999), apresentado na Seção 2.9.

5.1 Aplicações de Testes *Lossless*

As variações *lossless* das aplicações de testes propostas são baseadas em duas técnicas de compressão distintas. A primeira versão utiliza esquemas de predição linear e será referida a partir desse ponto como PL. Como codificadores de entropia para essa técnica foram testados os algoritmos PPM-C (Cleary e Witten, 1984), BPPM (Brasileiro e Cavalcanti, 2012) e BitGol (Golomb, 1966), apresentados nas Seções 4.3.2 e 4.3.4.

Uma predição linear pode gerar valores que não cabem na quantidade de *bits*

originalmente usada pelo sinal. Assim, uma codificação especial precisa ser realizada para assegurar que os erros de predição sejam passados para o codificador de entropia de maneira correta. O esquema usado foi o descrito no algoritmo:

Seja $s\,(t)$ uma amostra no tempo t e b o número de *bits* usado na digitalização do sinal s:

1. Calcular os limites superior $u_b = 2^{b-1} - 1$ e inferior $l_b = -2^{b-1}$ de representação de números inteiros com sinal para a quantidade de *bits* b;

2. Computar a predição $\hat{s}_n\,(t)$ de ordem n para a amostra $s\,(t)$;

3. Computar o erro $e_n\,(t)$ da predição de ordem n para a amostra $s\,(t)$;

4. Se $e_n\,(t)$ for um valor positivo ou zero, uma divisão em $c = \left\lfloor \frac{e_n(t)}{u_b} \right\rfloor + 1$ palavras de b *bits* será realizada, formando o conjunto C_k:

   4.1. As primeiras $c - 1$ palavras de C_k assumirão o valor u_b e a última palavra receberá o valor $e_n\,(t) - ((c-1) \times u_b)$;

5. Se $e_n\,(t)$ for um valor negativo, uma divisão em $c = \left\lfloor \frac{e_n(t)}{l_b} \right\rfloor + 1$ palavras de b *bits* será realizada, formando o conjunto C_k:

   5.1. As primeiras $c - 1$ palavras de C_k assumirão o valor l_b e a última palavra receberá o valor $e_n\,(t) + ((c-1) \times l_b)$;

O conjunto C_k com c palavras de b *bits*, todas no intervalo $[l_b..u_b]$, passa pela transformação mostrada na Equação 5.1 para que seus valores possam ser descritos em $\mathbb{N}$.

$$M\,(v) = \begin{cases} 2v, & \text{if } v \geq 0, \\ 2\,|v| - 1, & \text{if } v < 0. \end{cases} \tag{5.1}$$

O segundo método de compressão *lossless* – referido a partir deste ponto como IWT+SPIHT – é baseado em transformadas *wavelet*, mais especificamente nas IWTs (Calderbank et al., 1998), descritas na Seção 4.1.2.5. Diversas *wavelets* de coeficientes inteiros foram usadas nos testes, incluindo as transformadas: S-*Transform*, TS-*Transform*, S+P-*Transform*, (2,2), (2,4), (6,2), (2 + 2,2), e a *wavelet* biortogonal 9/7, cujas versões inteiras foram introduzidas por Calderbank et al. (1998). Nesse esquema os coeficientes transformados foram passados para um codificador SPIHT-1D (Said e Pearlman, 1996; Lu et al., 2000).

5.2 Testes *Lossy*

Esse trabalho apresenta três variações de compressores *lossy*, embora diversos testes exploratórios tenham sido feitos com uma variedade de outras técnicas. Algumas delas foram consideradas impróprias ou proibitivamente dispendiosas para a compressão de sinais biomédicos após os testes exploratórios. Os métodos descartados após os testes são mencionados na Seção 5.3. O primeiro compressor testado (DCT+MinLag) usa a DCT (Seção 4.1.1) e a Minimização Lagrangiana (Seção 3.5.3) como estratégia de Quantização Vetorial. A segunda estratégia (DCT+MinLag) é semelhante à DCT+MinLag, mas conta com a execução de um K-Médias (Seção 4.6) sobre os blocos DCT numa etapa prévia à Minimização Lagrangiana visando aumentar a estacionariedade do sinal. O terceiro método utiliza uma DWT (Seção 4.1.2) seguida de um quantizador vetorial SPIHT (Seção 4.3.6) – portanto, chamado de DWT+SPIHT. As próximas seções detalharão as abordagens de compressão *lossy*.

5.2.1 DCT+MinLag

O compressor DCT+MinLag implementado nesse trabalho usa apenas a DCT como transformada de domínio. O sinal transformado pela DCT é, então, quantizado otima-

mente por uma etapa de otimização da quantização vetorial por meio de Minimização Lagrangiana. A Minimização Lagrangiana aplicada ao contexto da quantização vetorial será explicada adiante na Seção 5.2.2.

Os coeficientes quantizados são passados, então, para codificadores de entropia numa ordem específica, visando se aproveitar da correlação entre as frequências de blocos vizinhos. Essa ordem segue o algoritmo seguinte:

1. Para cada índice de coeficiente $m = 0, 1, ..., L_b - 1$:

    1.1. Se o coeficiente atual é o DC:

        1.1.1. Codifique os coeficientes $\hat{B}_i[0]$ para todos os blocos $i = 0, 1, ..., \frac{N}{L_b} - 1$ diferencialmente e em ordem crescente;

    1.2. Se o coeficiente atual é algum AC:

        1.2.1. Codifique os coeficientes $\hat{B}_i[m]$ para todos os blocos $i = 0, 1, ..., \frac{N}{L_b} - 1$ em ordem crescente;

no qual N é o tamanho do sinal de entrada e L_b é o tamanho do bloco de transformação.

Embora a maioria dos coeficientes quantizados tenham magnitudes pequenas, alguns coeficientes podem ter uma vasta gama de valores possíveis, alguns deles precisando de até 12 ou 14 *bits* para serem codificados. Ainda há também o problema da codificação de valores negativos, que devem ser normalizados para intervalos positivos. Visando uniformizar a saída dos valores possíveis para caberem num número fixo de *bits* – no caso, 8 (um *byte*) – o seguinte esquema similar ao apresentado na Seção 5.1 é executado:

Seja $\hat{B}_i[m]$ o k-ésimo coeficiente lido:

1. Se $\hat{B}_i[m]$ for um valor positivo ou zero, ele será dividido em $c = \left\lfloor \frac{\hat{B}_i[m]}{127} \right\rfloor + 1$ *bytes*, formando o conjunto C_k:

    1.1. Os primeiros $c - 1$ *bytes* assumirão o valor 127 e o último *byte* receberá o valor $\hat{B}_i[m] - ((c - 1) \times 127)$;

2. Se $\hat{B}_i[m]$ for um valor negativo, ele será dividido em $c = \left\lfloor \frac{\hat{B}_i[m]}{-128} \right\rfloor + 1$ *bytes*, formando o conjunto C_k:

   2.1. Os primeiros $c-1$ *bytes* assumirão o valor -128 e o último *byte* receberá o valor $\hat{B}_i[m] + ((c-1) \times 128)$;

3. O conjunto C_k com c *bytes* – todos no intervalo $[-128..127]$ – é transformado de acordo com a Equação 5.1 para caber no espaço numérico não negativo [0..255].

no qual $\hat{B}_i[m]$ representa o coeficiente de índice m no bloco transformado e quantizado de índice i.

Antes dos coeficientes quantizados – na seção de cabeçalho do arquivo – o vetor de quantização também é codificado para uso na decodificação. Os codificadores de entropia usados para comprimir os coeficientes no domínio da frequência quantizados foram o PPM-C (Seção 4.3.4), o BPPM (Seção 4.3.5) e o BitGol (Seção 4.3.2).

O decodificador, por sua vez, segue os passos inversos do codificador, com exceção da otimização. Isso faz com que o conjunto Codificador/Decodificador (CODEC) seja assíncrono, ou seja, o tempo de codificação seja diferente do tempo de decodificação. Nesse compressor *lossy*, a decodificação é mais rápida que a codificação. A decodificação consiste em, primeiro, executar o decodificador de entropia, recuperando o vetor de quantização e o sinal quantizado no domínio da frequência. Em seguida, aplica-se uma dequantização, gerando o sinal aproximado ainda no domínio da frequência. Por último, a IDCT desse sinal é calculada, recuperando o sinal aproximado no domínio do tempo.

Um esquema apresentando graficamente os passos seguidos pelo compressor – incluindo os componentes do paradigma TQC – pode ser visto na Figura 5.1. Primeiramente o compressor recebe o sinal de entrada x e o transforma usando a DCT, resultando no sinal no domínio da frequência x_f. Esse sinal é passado, então, para a análise por meio da Minimização Lagrangiana, que calcula iterativamente os vetores de quantização e limiarização ótimos para a distorção passada como entrada para o método. O sinal

x_f é quantizado – resultando no sinal $\hat{x}_f$ – e passado para um codificador de entropia. Nota-se que a única diferença desse compressor para um compactador *lossy* baseado no paradigma TQC clássico é a presença da rotina de otimização para o cálculo dos parâmetros de quantização ótimos.

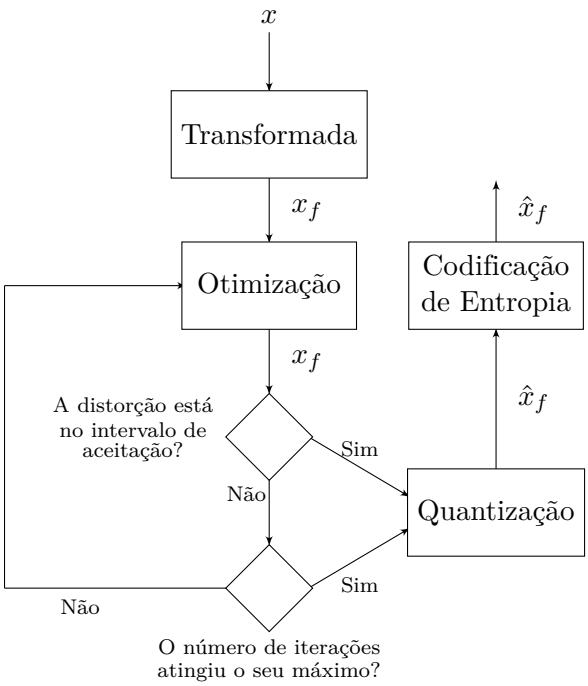


Figura 5.1: Esquema ligando os módulos do codificador DCT+MinLag.

O decodificador, como em muitas estratégias *lossy*, é um simples subconjunto de operações inversas às do codificador, recriando uma aproximação para o sinal original. Primeiramente um decodificador de entropia é executado, recuperando o sinal quantizado no domínio da frequência $\hat{x}_f$. Esse sinal é dequantizado, obtendo uma aproximação para o sinal transformado x_f, referida como $\tilde{x}_f$. Por último, tal aproximação sofre uma operação de transformação inversa, recuperando o sinal aproximado no domínio do tempo $\tilde{x}$. A Figura 5.2 apresenta o esquema da decodificação.

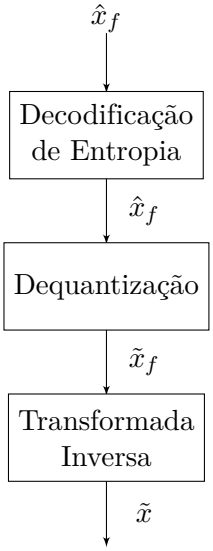


Figura 5.2: Esquema ligando os módulos do decodificador DCT+MinLag.

5.2.2 Minimização Lagrangiana e Quantização Vetorial

Quantizadores vetoriais, apesar de produzirem resultados de compressão melhores que quantizadores escalares, são muitas vezes proibitivos do ponto de vista de tempo de execução. Por exemplo, se nenhum método de otimização for usado para calcular um vetor de quantização ótimo – levando em conta a distorção d – para um bloco transformado com k coeficientes que variam no intervalo $[1 \,..\, q]$, seriam necessárias k^q operações de transformação inversa. Levando em conta um vetor de limiarização para quantização com zona-morta, esse número dobraria de tamanho. Operações de transformação inversa são potencialmente pesadas, logo sua quantidade deve ser minimizada o quanto for possível.

A busca no espaço de soluções realizada por meio da minimização de multiplicadores de Lagrange, além de levar em conta só a BCI – como mostrado na Seção 3.5.3 – também permite que buscas paralelas sejam feitas. Blocos de coeficientes gerados por transformadas como a DCT e a DWT geram coeficientes altamente descorrelacionados. Dessa forma, para aferir os melhores vetores de quantização para compressores *lossy*

baseados nessas transformadas, é possível realizar uma busca paralela no espaço de soluções para os coeficientes de cada índice nos blocos. Em outras palavras, se um sinal de tamanho n foi dividido em $^{n}/_{k}$ blocos de transformação de tamanho k, um total de k otimizações via Minimização Lagrangiana poderão ser executadas em paralelo para formar um vetor de quantização também de tamanho k. Cada otimização dos coeficientes de índice i nos vários blocos do sinal é executada levando em conta apenas o respectivo Plano R-D Operacional (ver Seção 3.4) formado pelos coeficientes de índice i. Isso é possível apenas para transformadas que tenham a propriedade de descorrelacionar os seus coeficientes. Essa busca independente entre os diferentes índices de coeficientes transformados confere ao algoritmo uma complexidade polinomial, substituindo a complexidade exponencial original. Nesse caso, um limite máximo para o número de operações de transformação inversa pode ser definido empiricamente. Nesse trabalho verificou-se que um total de 20 operações de transformação inversa foram suficientes para atingir níveis de distorção próximos aos que foram passados como parâmetro. Tanto o compressor DCT+MinLag quanto o DCT+KMinLag foram implementados com valores máximos de quantização e *thresholding* de 128. Apenas para alguns sinais mais suscetíveis a perda de qualidade – como os RSPs – foi utilizado o valor 256.

O cálculo das distorções reais – de acordo com uma métrica complexa como a NPRD (Seção 3.2) – pode ser substituído por uma medida de distorção de cálculo não dispendioso: a MSE. Devido à proporcionalidade direta entre a MSE e a NPRD, esta age como uma aproximação, o que é suficiente para o cálculo das BCIs sobre as quais as Minimizações Lagrangianas paralelas operam. Isso evita que operações de transformação inversa precisem ser executadas para calcular os pontos no eixo das distorções do Plano R-D Operacional.

Para a construção do eixo da taxa de *bits* de saída do codificador no Plano R-D Operacional, faz-se necessária também uma medida de aproximação, já que, para o cálculo da taxa de *bits* real, operações de transformação inversa seriam necessárias. A

métrica usada nesse caso é a Entropia (Seção 3.3).

Uma vez que os pontos dos diversos Planos R-D Operacional são calculados, a Minimização Lagrangiana começa a agir paralelamente para calcular os pontos da BCI que geram o vetor de quantização e o vetor de limiarização para quantização com zona-morta (ver Seção 4.2.2), de acordo com a distorção predefinida.

5.2.3 DCT+KMinLag

Como introduzido na Seção 5.2, o compressor DCT+KMinLag é bastante parecido com o método DCT+MinLag – apresentado na Seção 5.2.1. A diferença se dá no pré-processamento utilizando o algoritmo K-Médias (Seção 4.6), que tem como o objetivo aumentar a estacionariedade do sinal, melhorando hipoteticamente a compactação da informação nos coeficientes de mais baixo nível da DCT. Para atingir a separação natural dos *clusters* proposta pelo K-Médias, cada bloco DCT é considerado uma amostra e cada coeficiente representa um atributo.

O uso do K-Médias permite que K Minimizações Lagrangianas sejam realizadas em paralelo, resultando em K vetores de quantização com zona-morta. Cada vetor v_k (com $k = 0,1,...,K-1$) de quantização com zona-morta é responsável por quantizar os blocos DCT agrupados no *cluster* do tipo k. Em contrapartida ao potencial ganho de compressão, os tipos dos blocos devem ser passados para o arquivo de saída e codificados, gerando uma carga adicional de informações a serem comprimidas. Além disso, ainda é necessário considerar a sobrecarga de tempo de codificação dos sinais, dado que K otimizações lagrangianas – o procedimento mais custoso do método DCT+MinLag – são necessárias.

5.2.4 DWT+SPIHT

O terceiro método *lossy* testado inclui a utilização de DWTs (Seção 4.1.2), as quais têm seus coeficientes codificados por um SPIHT (Seção 4.3.6). Como mostrado na Figura 5.3,

o sinal no domínio do espaço x primeiramente é transformado usando a DWT, gerando o sinal composto de coeficientes *wavelet* x_w. Posteriormente, o sinal x_w é enviado para o codificador de entropia SPIHT, que age como um quantizador vetorial, otimizando a quantidade de informação enviada para o arquivo de saída de acordo com a distorção passada como parâmetro para o método. Chamar-se-á o sinal quantizado após o SPIHT de $\hat{x}_w$. Por questões de simplificação, o método de otimização por distorção usando SPIHT desse trabalho apenas aceita um limite superior de distorção, codificando o sinal do plano de *bits* mais significativo até o plano de *bits* que resulte numa distorção menor que a passada como parâmetro. Em outras palavras, não se garante que a distorção seja próxima da passada como parâmetro para o método, apenas é assegurado que ela seja menor.

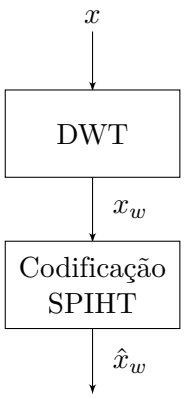


Figura 5.3: Esquema ligando os módulos do codificador DWT+SPIHT.

Na etapa de decodificação, primeiramente o SPIHT recupera os coeficientes *wavelet* quantizados $\hat{x}_w$, dequantizando-os e formando o sinal no domínio *wavelet* aproximado $\tilde{x}_w$. Os coeficientes $\tilde{x}_w$ são, então, passados para uma transformada IDWT, recuperando o registro aproximado no domínio do tempo $\tilde{x}$. Esse processo é caracterizado pela Figura 5.4.

Devido à alta demanda de tempo necessária para os testes desse método, apenas algumas bases *wavelet* puderam ser testadas. O conjunto reduzido de bases *wavelet*

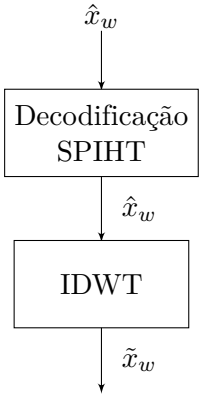


Figura 5.4: Esquema ligando os módulos do decodificador DWT+SPIHT.

usado nesse trabalho – junto com as respectivas execuções dos métodos DCT+MinLag e DCT+KMinLag usadas para comparação – demandou por volta de duas semanas para ser testado. Testes exploratórios de menor escala revelaram que a escolha da base *wavelet* tem pouca influência na eficácia de compressão do método, o que também motivou a escolha de um conjunto reduzido de bases. As bases escolhidas para os testes exaustivos foram as bases biortogonais 9/7 e 5/3 – conhecidas por fazerem parte do formato JPEG 2000 (JPEG 2000 Organization, 2000) – e as bases ortogonais Daubechies 10 (com *kernel* de tamanho 20) e a Symmlet 10 (também com filtros com 20 *taps*). As bases Daubechies 10 e Symmlet 10 foram as escolhidas dentre suas respectivas famílias de bases por sua alta quantidade de *taps*, o que é fundamental para que as *wavelets* sejam capazes de representar sinais cada vez mais complexos com uma maior concentração estatística em seus coeficientes.

5.3 Métodos Descartados

Algumas técnicas pesquisadas nesse trabalho foram consideradas inapropriadas ou proibitivamente complexas para o contexto de compressão de sinais biológicos após uma série de baterias de testes exploratórios. Entre esses métodos estão a utilização de DSTs; a transformada KLT; pré-processamentos para a etapa de transformação usando

predições por blocos, como usado em compressores de vídeo como o H.264 (ITU-T264, 2002); o uso da transformada Hadamard para transformação de níveis DC nos blocos DCT, como descrito por (Richardson, 2010); e pré-processamentos para a codificação de entropia usando Transformadas de Burrows-Wheeler (*Burrows-Wheeler Transform* – BWT) (Burrows e Wheeler, 1994) e a transformada *Move To Front* (MTF).

Todos os testes exploratórios realizados com a transformada DST mostraram-se inefetivos para a compressão de sinais biológicos. Isso provavelmente se dá porque a DST apresenta bons resultados de compactação apenas em casos bem específicos nos quais o sinal tem uma certa distribuição estatística não comumente vista em sinais capturados da natureza. No H.265 (Sullivan et al., 2012), por exemplo, a DST é usada especificamente na codificação de blocos de *pixels* preditos via predições *intra-frame*, ou seja, predições feitas de acordo com blocos de *pixels* do próprio quadro atual do vídeo. Já a transformada KLT (Karhunen, 1947; Loève, 1948), como justificado na Seção 4.1, foi considerada muito computacionalmente complexa para ser usada na prática.

Predições em blocos baseadas nas predições *inter-frames* de codificadores de vídeo também se mostraram inefetivas na compressão de sinais biológicos. Foram executados testes exploratórios usando predições em bloco como pré-processamento para o método DCT+MinLag apresentado na Seção 5.2, resultando em perdas sutis de compressão para sinais eletrocardiográficos. Além de não resultar em ganhos de RC, a procura por blocos de predição prolonga o tempo de execução do codificador.

Testes exploratórios mostraram que, apesar da transformada Hadamard diminuir a quantidade de informação que precisa ser codificada nos níveis DC do método DCT+MinLag, uma codificação diferencial – como um DPCM (Seção 4.5) – obtém resultados consideravelmente melhores a um custo computacional menor.

As BWTs, por sua vez, resultaram em perdas de *performance* de compressão consideráveis, quando usadas como pré-processamento para os métodos de codificação de entropia no caso do compressor DCT+MinLag. Esse resultado pode ser explicado pela

ordem em que os coeficientes são passados para o codificador de entropia, que já foi pensada para explorar ao máximo as similaridades entre coeficientes DCT quantizados de blocos vizinhos, como pode ser visto na Seção 5.2. Devido à ordem de codificação baseada nos índices dos coeficientes e à característica da DCT de concentrar a informação nas baixas frequências, após certo ponto uma vasta quantidade de coeficientes de altas frequências tende a assumir o valor zero. Essa é uma característica da qual muitos codificadores de entropia – como o PPM-C (Cleary e Witten, 1984), o BPPM (Brasileiro e Cavalcanti, 2012) e o BitGol (Golomb, 1966) – se aproveitam para atingir altas RCs. A BWT acaba por misturar coeficientes de alta frequência com coeficientes de baixa frequência, desordenando a sequência com vários coeficientes de valor zero contínuos ao misturá-los com coeficientes de valor não nulo.

5.4 Conclusão do Capítulo

O Capítulo 5 definiu os compressores *lossless* e *lossy* utilizados na etapa de testes desse trabalho. O Capítulo 6 explicará a rotina de testes e os resultados obtidos, os quais serão explicados no Capítulo **??**.

Capítulo 6

Resultados

A rotina de testes desse trabalho está compreendida em três classes: testes dos compressores *lossless*, testes do compressor DCT+MinLag, e testes de comparação da estratégia DWT+SPIHT com as estratégias DCT+MinLag e DCT+KMinLag. Como é de se esperar, os resultados de compressão *lossless* geram RCs menores que os resultados dos codificadores *lossy*.

Como a quantidade de ruído e aleatoriedade em cada registro biomédico podem afetar a compressão, as RCs apresentadas neste capítulo são as médias das RCs individuais de todos os canais de um certo tipo de registro biomédico no MIT/BIH PSGDB, salvo os resultados da Seção 6.3, na qual os resultados são avaliados independentemente para cada registro biomédico. Esse cálculo da média visa suavizar o efeito dos *outliers* – principalmente sinais com erros de sensor que afetam a eficácia da compressão. As Seções 6.1, 6.2 e 6.3 apresentam respectivamente os resultados obtidos com as estratégias *lossless*, DCT+MinLag e uma comparação entre as técnicas DCT+MinLag, DCT+KMinLag e DWT+SPIHT descritas nas Seções 5.1 e 5.2.

6.1 Resultados das Estratégias de Compressão *Lossless*

Os resultados dos codificadores *lossless* estão separados em duas tabelas. A Tabela 6.1 mostra as RCs provenientes da estratégia IWT+SPIHT com diversas bases de transformadas *wavelet*. A Tabela 6.2 contém as RCs do compressor PL com predições de ordens 0, 1, 2 e 3 e os codificadores de entropia PPM-C (Cleary e Witten, 1984), BPPM (Brasileiro e Cavalcanti, 2012) e BitGol (Golomb, 1966). Algumas *wavelets* – Daubechies 4 (D4), e as transformadas (4,2) e (4,4) – foram omitidas dos testes desse trabalho por não terem apresentado boas taxas de compressão para nenhum tipo de sinal.

Tabela 6.1: Resultados de RC para o compressor IWT+SPIHT com 8 bases *wavelet* e 3 tamanhos de bloco nos sinais do MIT/BIH PSGDB (Ichimaru e Moody, 1999).

Canal	**Bloco**	**Bases *wavelet***							
		S	**TS**	**S+P**	**2,2**	**2,4**	**6,2**	**2+2,2**	**9/7**
ECG	**512**	1,91	2,02	2,03	2,06	2,05	2,07	**2,07**	1,98
	1024	1,92	2,03	2,05	2,07	2,06	2,08	**2,09**	1,99
	2048	1,92	2,04	2,05	2,07	2,07	2,09	**2,09**	2,00
BP	**512**	2,19	2,88	2,85	2,92	2,91	**3,04**	3,00	2,49
	1024	2,20	2,94	2,91	2,97	2,96	**3,13**	3,08	2,53
	2048	2,20	2,97	2,95	3,00	2,99	**3,17**	3,11	2,56
EEG	**512**	1,55	1,63	1,63	1,64	1,64	1,65	**1,65**	1,64
	1024	1,56	1,63	1,63	1,65	1,65	1,65	**1,66**	1,65
	2048	1,56	1,63	1,64	1,65	1,65	1,66	**1,66**	1,65
RSP	**512**	2,60	2,89	2,82	**3,02**	3,00	2,96	2,98	2,53
	1024	2,60	3,00	2,92	**3,11**	3,08	3,07	3,09	2,60
	2048	2,61	3,05	2,98	3,15	3,13	3,14	**3,15**	2,64
EOG	**512**	2,13	2,45	2,48	2,47	2,45	**2,58**	2,52	2,30
	1024	2,14	2,48	2,51	2,49	2,48	**2,61**	2,55	2,32
	2048	2,14	2,49	2,52	2,50	2,49	**2,62**	2,56	2,33
EMG	**512**	1,16	1,16	1,15	1,18	1,18	1,17	1,18	**1,19**
	1024	1,16	1,16	1,16	1,18	1,18	1,17	1,18	**1,19**
	2048	1,16	1,16	1,16	1,18	1,18	1,17	1,18	**1,19**
SV	**512**	1,74	2,24	2,31	2,22	2,22	**2,53**	2,38	2,09
	1024	1,75	2,27	2,35	2,25	2,24	**2,58**	2,42	2,12
	2048	1,75	2,29	2,37	2,26	2,25	**2,60**	2,44	2,13
O_2S	**512**	**3,36**	2,99	2,98	3,17	3,17	2,98	3,04	2,83
	1024	**3,39**	3,17	3,16	3,31	3,31	3,17	3,23	3,00
	2048	**3,41**	3,27	3,27	3,39	3,40	3,29	3,34	3,11

Tabela 6.2: Resultados de RC do compressor *lossless* PL usando 4 codificadores de entropia diferentes e preditores lineares com 4 ordens diferentes nos sinais do MIT/BIH PSGDB (Ichimaru e Moody, 1999).

Canal	Predição	PPM	BPPM	BitGol
ECG	**P0**	2,38	1,96	1,36
	P1	2,50	2,39	1,85
	P2	2,39	2,23	1,83
	P3	2,13	1,97	1,74
BP	**P0**	3,54	2,18	1,19
	P1	3,99	3,22	2,17
	P2	3,91	3,41	2,73
	P3	3,62	2,86	2,49
EEG	**P0**	1,56	1,53	1,31
	P1	1,63	1,74	1,71
	P2	1,57	1,65	1,62
	P3	1,44	1,49	1,45
RSP	**P0**	3,38	2,51	1,20
	P1	3,96	3,53	2,46
	P2	3,97	3,26	2,48
	P3	3,47	2,68	2,16
EOG	**P0**	2,72	2,02	1,36
	P1	2,92	2,69	2,31
	P2	2,86	2,81	2,41
	P3	2,68	2,49	2,28
EMG	**P0**	1,15	1,21	1,19
	P1	1,12	1,22	1,20
	P2	1,06	1,10	1,08
	P3	0,98	0,96	0,91
SV	**P0**	2,34	1,69	1,11
	P1	3,15	2,38	1,70
	P2	3,21	2,94	2,28
	P3	3,10	2,87	2,36
O_2S	**P0**	4,51	3,45	1,17
	P1	4,82	4,09	3,63
	P2	4,02	3,29	2,95
	P3	3,42	2,73	2,58

Uma breve análise das Tabelas 6.1 e 6.2 demonstra que o esquema *lossless* baseado em predições lineares obteve melhores resultados de RC em todos os casos de testes. Os EMGs obtiveram as piores *performances* de compressão para ambos os codificadores,

chegando no máximo a RCs de 1.218 : 1. Nos testes nos quais foram usados preditores de alta ordem ($P3$), esses sinais apresentaram um aumento no tamanho original ao invés de uma compactação, ou seja, RCs menores que 1 : 1. As mais altas *performances* de compressão usando *wavelets* foram obtidas pelos O_2Ss, chegando até RCs de 3,408 : 1 com a transformada S.

Por causa da grande variação de formas, amplitude e quantidade de ruído nos diferentes registros biomédicos, não houve uma única base *wavelet* ótima para a compressão de todos os canais nos testes do codificador IWT+SPIHT. ECGs e EEGs, por exemplo, obtiveram RCs maiores – 2,091 : 1 e 1,660 : 1 respectivamente – com a utilização da transformada $(2 + 2{,}2)$, enquanto BPs, EOGs e SVs foram comprimidos otimamente usando a base (6,2), resultando nas respectivas RCs 3,170 : 1, 2,620 : 1 e 2,600 : 1. Os EMGs, por sua vez, obtiveram RCs de, no máximo, 1,190 : 1 com a utilização da base *wavelet* biortogonal 9/7. Por fim, os RSPs obtiveram melhores performances de compressão com a transformada (2,2) e com sua variação mais complexa $(2 + 2{,}2)$, dependendo do tamanho do bloco, chegando a atingir até 3,151 : 1 de RC.

O método PL mostrou melhores resultados com preditores de primeira ordem (P1), superando P0, P2 e P3, na maioria dos casos, ficando ligeiramente abaixo da *performance* de P2 nos RSPs e SVs. As evoluções das RCs dos sinais de acordo com a variação das ordens de predição são mostradas graficamente na Figura 6.1, na qual fica claro que até nos casos nos quais P2 se comportou melhor a diferença para P1 foi pouca, assegurando a robustez de P1 para a compressão de praticamente todos os sinais biomédicos testados nesse trabalho.

Como mostrado na Figura 6.2, o esquema de compressão PL+PPM-C atingiu resultados de RC melhores que os outros codificadores de entropia e que o compressor IWT+SPIHT para a vasta maioria dos sinais, sendo superado pelo BPPM apenas nos EEGs e EMGs.

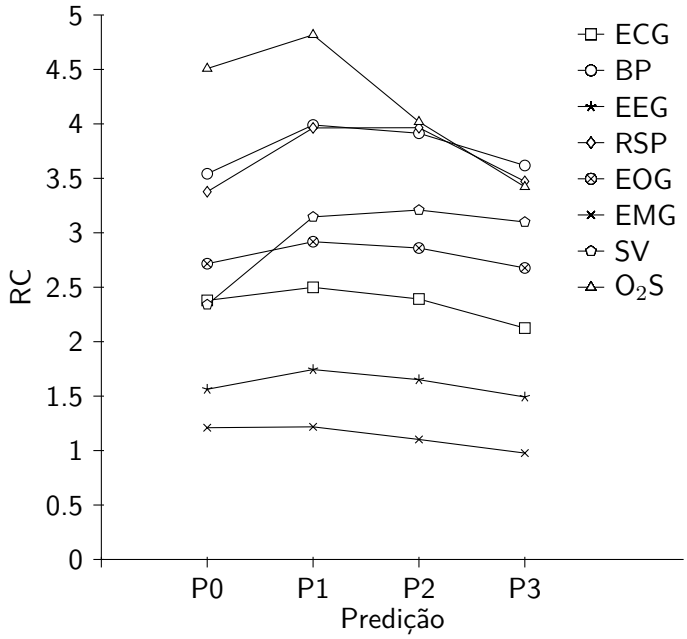


Figura 6.1: Melhores resultados de RC usando os preditores $P0$, $P1$, $P2$ e $P3$.

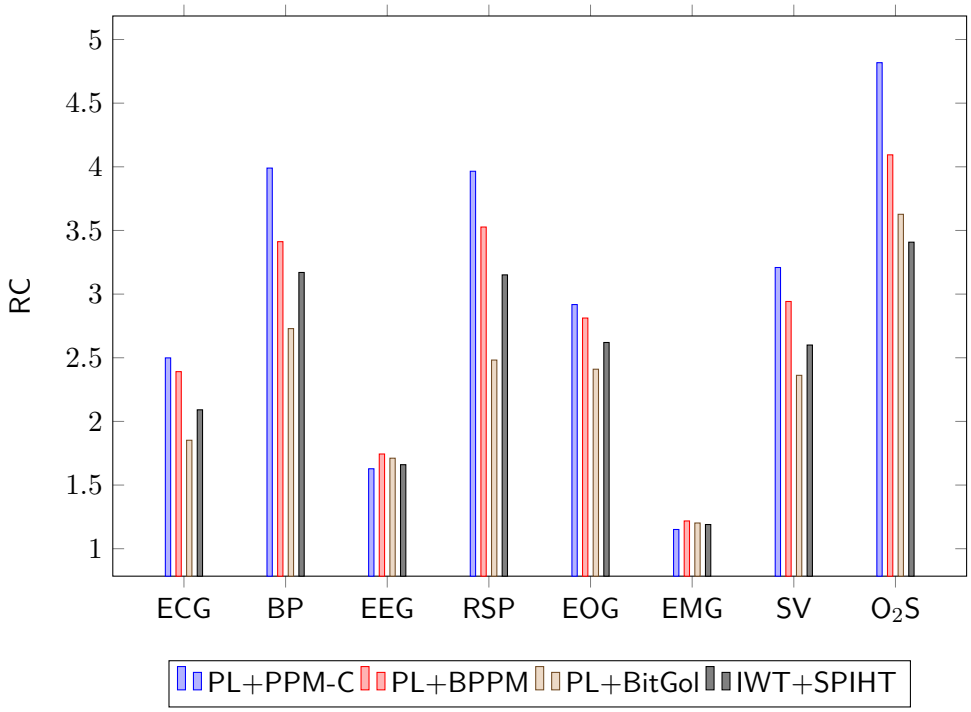


Figura 6.2: Melhores RCs dos compressores *lossless* PL e IWT+SPIHT.

6.2 Resultados da Estratégia de Compressão DCT+MinLag

Lee e Buckley (1999) testaram o uso de DCTs bidimensionais para a compressão de ECGs com tamanhos de blocos de 4×4 até 64×64, limitando os resultados a potências na base 2. Esses resultados demonstraram uma saturação na eficácia de compressão com blocos de 32×32 e 64×64 amostras. A rotina de testes do compressor DCT+MinLag basear-se-á nos experimentos de Lee e Buckley (1999), Batista (2002) e Oliveira et al. (2014) e estará limitada a blocos de 16, 32 e 64 amostras. Segundo Oliveira et al. (2014), canais de dados mais aleatórios e com coeficientes de alta frequência mais acentuados se mostraram mais suscetíveis à compressão usando blocos de transformação pequenos – como ECGs, EEGs e EMGs – logo eles foram comprimidos exclusivamente com blocos de 16 ou 32 amostras. Analogamente, sinais com linhas base mais bem definidas e com variações de alta frequência menores – BPs, RSPs, EOGs, SVs e O_2Ss – foram submetidos a testes com blocos de transformação com 64 amostras.

Novamente o *corpus* de testes usado foi o MIT/BIH PSGDB (Ichimaru e Moody, 1999). Sinais contendo diferentes quantidades de ruído ou amplitude podem ser mais ou menos suscetíveis à compressão, logo as RCs apresentadas da Tabela 6.3 à Tabela 6.9 foram geradas pela média de todas as RCs dos registros dos canais de dados.

O_2Ss mostraram-se muito ruidosos e suscetíveis a erros de captura, o que dificulta a otimização dos vetores de quantização e limiarização pela Minimização Lagrangiana. Dessa forma, esses canais de dados foram tratados de forma diferente: a Tabela 6.10 mostra as RCs para cada registro de O_2S em separado, usando as configurações de vetores de quantização e limiarização mínimas obtidas pela Minimização Lagrangiana. Essa metodologia diferenciada foi adotada porque testes exploratórios mostraram que otimizações para valores de NPRD (Equação 3.6) tão baixos quanto 0,5% e 0,75% já resultam na geração de artefatos artificiais indesejáveis na reconstrução desses sinais, como fortes efeitos de limiarização e o fenômeno de Gibbs (Gibbs, 1898).

Tabela 6.3: Resultados de RC após a codificação usando o método DCT+MinLag dos ECGs do MIT/BIH PSGDB com blocos de transformação com 16 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	2,87:1	2,21:1	2,64:1
1,5%	3,46:1	2,67:1	3,19:1
2,0%	4,07:1	3,12:1	3,71:1
2,5%	4,75:1	3,60:1	4,27:1
3,0%	5,46:1	4,07:1	4,82:1
3,5%	6,22:1	4,54:1	5,34:1
4,0%	6,98:1	4,98:1	5,85:1
4,5%	7,78:1	5,42:1	6,30:1
5,0%	8,59:1	5,83:1	6,61:1

Tabela 6.4: Resultados de RC após a codificação usando o método DCT+MinLag dos BPs do MIT/BIH PSGDB com blocos de transformação com 64 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	9,05:1	8,07:1	9,05:1
1,5%	13,81:1	12,57:1	13,77:1
2,0%	18,00:1	15,88:1	17,20:1
2,5%	21,48:1	18,22:1	19,69:1
3,0%	24,73:1	20,20:1	21,88:1
3,5%	27,79:1	21,81:1	23,97:1
4,0%	30,53:1	23,22:1	25,53:1
4,5%	33,34:1	24,56:1	26,93:1
5,0%	35,72:1	25,57:1	28,00:1

Tabela 6.5: Resultados de RC após a codificação usando o método DCT+MinLag dos EEGs do MIT/BIH PSGDB com blocos de transformação com 32 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	2,54:1	2,06:1	2,53:1
1,5%	2,93:1	2,34:1	2,94:1
2,0%	3,32:1	2,62:1	3,32:1
2,5%	3,69:1	2,90:1	3,69:1
3,0%	4,06:1	3,18:1	4,05:1
3,5%	4,43:1	3,46:1	4,41:1
4,0%	4,83:1	3,76:1	4,79:1
4,5%	5,24:1	4,07:1	5,18:1
5,0%	5,65:1	4,39:1	5,58:1

Tabela 6.6: Resultados de RC após a codificação usando o método DCT+MinLag dos RSPs do MIT/BIH PSGDB com blocos de transformação com 64 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	22,58:1	18,49:1	19,26:1
1,5%	33,98:1	27,17:1	27,57:1
2,0%	42,28:1	32,93:1	32,48:1
2,5%	58,50:1	44,19:1	43,27:1
3,0%	67,21:1	50,43:1	46,22:1
3,5%	71,41:1	53,24:1	49,56:1

Tabela 6.7: Resultados de RC após a codificação usando o método DCT+MinLag dos EOGs do MIT/BIH PSGDB com blocos de transformação com 64 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	5,79:1	4,88:1	6,06:1
1,5%	7,82:1	6,65:1	8,31:1
2,0%	9,68:1	8,25:1	10,30:1
2,5%	11,83:1	10,12:1	12,59:1
3,0%	13,79:1	11,83:1	14,62:1
3,5%	15,82:1	13,61:1	16,79:1
4,0%	17,99:1	15,52:1	19,01:1
4,5%	20,35:1	17,68:1	21,41:1
5,0%	22,74:1	19,88:1	23,81:1

Tabela 6.8: Resultados de RC após a codificação usando o método DCT+MinLag dos EMGs do MIT/BIH PSGDB com blocos de transformação com 32 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	1,96:1	1,67:1	1,92:1
1,5%	2,19:1	1,80:1	2,15:1
2,0%	2,38:1	1,91:1	2,34:1
2,5%	2,55:1	2,01:1	2,50:1
3,0%	2,71:1	2,11:1	2,64:1
3,5%	2,86:1	2,20:1	2,78:1
4,0%	2,99:1	2,29:1	2,91:1
4,5%	3,13:1	2,38:1	3,04:1
5,0%	3,27:1	2,47:1	3,17:1

Tabela 6.9: Resultados de RC após a codificação usando o método DCT+MinLag dos SVs do MIT/BIH PSGDB com blocos de transformação com 64 amostras.

NPRD	**Codificador de Entropia**		
	PPM	**Golomb**	**BitGol**
1,0%	9,12:1	8,06:1	9,63:1
1,5%	11,22:1	9,89:1	11,84:1
2,0%	13,37:1	11,68:1	13,01:1
2,5%	15,38:1	11,64:1	13,59:1
3,0%	18,04:1	14,27:1	17,05:1
3,5%	18,81:1	15,04:1	17,55:1
4,0%	20,60:1	17,62:1	19,42:1
4,5%	22,43:1	19,07:1	21,03:1
5,0%	23,97:1	20,33:1	21,40:1

Tabela 6.10: Resultados de RC após a codificação usando o método DCT+MinLag dos O_2Ss do MIT/BIH PSGDB com blocos de transformação com 64 amostras.

Registro	**NPRD**	**Codificador de Entropia**		
		PPM	**Golomb**	**BitGol**
slp59	0,134%	6,60:1	4,37:1	6,11:1
slp60	0,262%	7,10:1	4,61:1	6,75:1
slp61	0,356%	6,70:1	4,56:1	6,51:1
slp66	1,072%	7,03:1	4,52:1	6,66:1
slp67x	0,305%	5,98:1	3,95:1	5,68:1

Os resultados de RC dos canais de dados de RSP, mostrados na Tabela 6.6, apenas variam entre NPRDs de 1,0% até 3,5% porque para atingir níveis de distorção maiores que isso é necessário usar valores de quantização e limiarização muito altos, tornando o processo de codificação proibitivamente lento. Além disso, distorções de 3,5% em RSPs já geram efeitos indesejáveis de limiarização dos blocos de transformação, como mostrado na Figura B.4g.

As RCs obtidas pela execução do PPM-C mostraram-se visivelmente maiores que as RCs do Golomb para todos os canais. O codificador BitGol, por sua vez, superou as *performances* de compressão do PPM-C em EEGs, EOGs e SVs para alguns níveis de distorção e chegou próximo da eficácia de compressão do PPM-C para a maioria dos sinais. Como mostra a Tabela 6.11, em todos os casos o BitGol rodou aproximadamente

de 2 a 4 vezes mais rápido que o PPM-C, além de ser um algoritmo que, por não necessitar de estruturas de dados complexas, ocupa menos espaço em memória. É importante notar que essa comparação apenas levou em conta o tempo usado pelos algoritmos de codificação de entropia, não incluindo o tempo usado para o cálculo dos parâmetros de quantização ótimos. As RCs mostradas da Tabela 6.3 à Tabela 6.10 e os tempos apresentados na Tabela 6.11 relativos ao PPM-C levam em conta apenas a execução de maior RC dentre os contextos máximos 1, 2, 3, 4 e 5.

Tabela 6.11: Comparação dos tempos de execução dos compressores PPM-C e BitGol para todos os tipos de sinais do MIT/BIH PSGDB.

Canal	**Execução do PPM-C (em segundos)**	**Execução do BitGol (em segundos)**	$\frac{\text{Execução do PPM-C}}{\text{Execução do BitGol}}$
ECG	17,8 s	4,5 s	4,0
BP	9,2 s	3,7 s	2,5
EEG	16,2 s	6,0 s	2,7
RSP	9,7 s	3,8 s	2,5
EOG	14,1 s	5,0 s	2,8
EMG	30,2 s	8,8 s	3,4
SV	7,4 s	3,5 s	2,1
O_2S	8,0 s	3,6 s	2,2

As Figuras B.1, B.2, B.3, B.4, B.5, B.6, B.7 e B.8, no Apêndice B, mostram os resultados de reconstrução com seções de alguns sinais do MIT/BIH PSGDB após compressão ótima com várias NPRDs diferentes. As Figuras 6.3, 6.4, 6.5, e 6.6 mostram as curvas de variação de RC de acordo com o aumento da NPRD usando o PPM-C e o BitGol como codificadores de entropia para os registros de ECG, BP, EEG e RSP. Esses resultados de compressão são comparados com os apresentados por Oliveira et al. (2014). Como Oliveira et al. (2014) só realizaram testes com ECGs, BPs, EEGs e RSPs, as Figuras 6.7, 6.8 e 6.9 apenas comparam as RCs dos codificadores PPM-C e BitGol apresentados nesse trabalho para EOGs, EMGs e SVs, respectivamente. O eixo y representa os valores já digitalizados das amostras em todas as figuras de reconstrução nesse trabalho. O *website* da PhysioNet (`http://physionet.org/cgi-bin/atm/ATM`) descreve com mais detalhes as unidades analógicas prévias à digitalização.

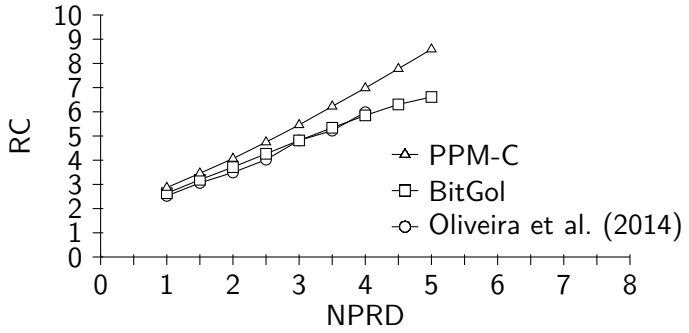


Figura 6.3: Comparação de RC entre o PPM-C, o BitGol e a estratégia apresentada por Oliveira et al. (2014) para registros de ECG.

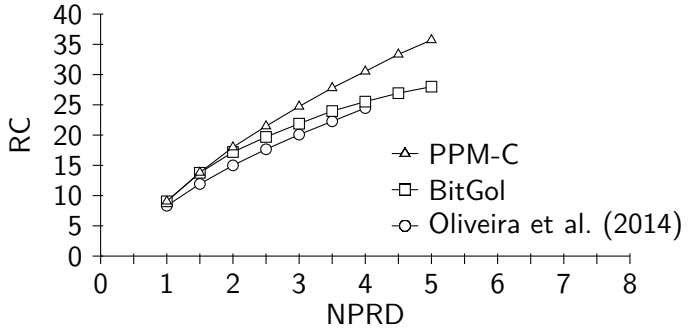


Figura 6.4: Comparação de RC entre o PPM-C, o BitGol e a estratégia apresentada por Oliveira et al. (2014) para registros de BP.

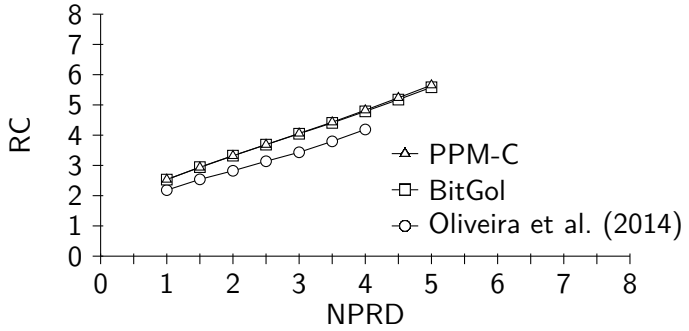


Figura 6.5: Comparação de RC entre o PPM-C, o BitGol e a estratégia apresentada por Oliveira et al. (2014) para registros de EEG.

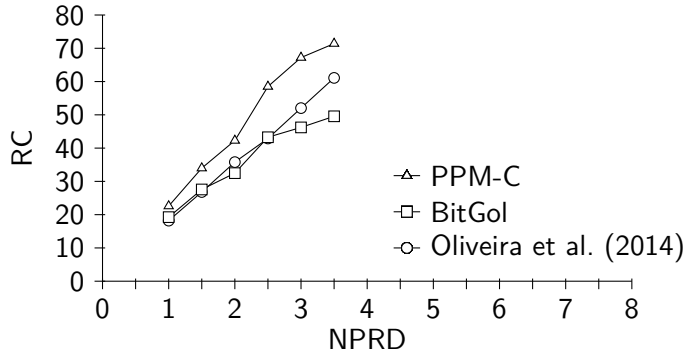


Figura 6.6: Comparação de RC entre o PPM-C, o BitGol e a estratégia apresentada por Oliveira et al. (2014) para registros de RSP.

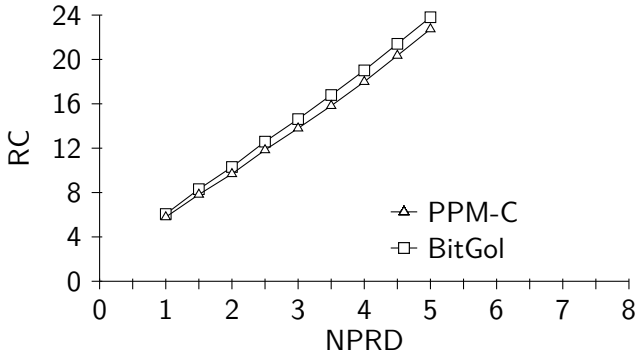


Figura 6.7: Comparação de RC entre o PPM-C e o BitGol para registros de EOG.

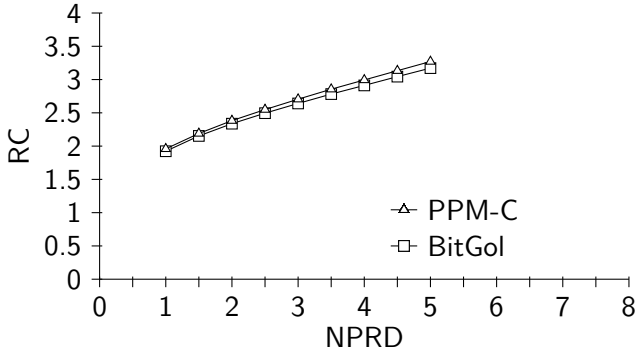


Figura 6.8: Comparação de RC entre o PPM-C e o BitGol para registros de EMG.

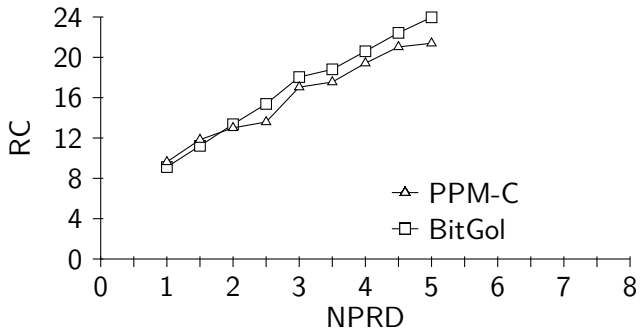


Figura 6.9: Comparação de RC entre o PPM-C e o BitGol para registros de SV.

6.3 Comparação Entre as Estratégias de Compressão *Lossy*

Como explicado na Seção 5.2.4, o codificador DWT+SPIHT foi implementado para otimizar a distorção para qualquer valor abaixo do recebido como entrada. Dessa forma, para uma avaliação justa, primeiramente o codificador DWT+SPIHT foi executado, para, em seguida, se comprimir os sinais usando os codificadores DCT+MinLag e DCT+KMinLag. Essa metodologia permite que o valor de distorção retornado pelo método DWT+SPIHT seja usado como parâmetro para os métodos baseados na Minimização Lagrangiana. As Tabelas 6.12 a 6.22 mostram os valores de comparação de RC e NPRD com a *wavelet* Symmlet 10.

Os coeficientes quantizados gerados pelos métodos DCT+MinLag e DCT+KMinLag foram comprimidos usando apenas um PPM-C com contexto de tamanho 2 como codificador de entropia. Observa-se que os métodos baseados em Minimização Lagrangiana principalmente para RSPs e O_2Ss não conseguiram atingir a NPRD retornada pelo método DWT+SPIHT devido à limitação nos coeficientes de quantização e zona-morta aplicados no método. A utilização de valores de quantização e zona-morta demasiadamente grandes – no caso, 512 ou 1024 – representa para a Minimização Lagrangiana um tempo de execução adicional bastante considerável, tornando a complexidade computacional do método proibitivamente custosa. Os resultados com outras bases *wavelet* podem ser vistos no Apêndice C.

Tabela 6.12: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para ECGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp01a	4,43	3,05	**4,75**	3,02	4,68	3,02
slp01b	6,51	4,24	**7,48**	4,21	7,28	4,26
slp02a	3,60	3,35	**3,90**	3,34	3,82	3,32
slp02b	3,49	3,05	**3,66**	3,04	3,64	3,06
slp03	6,29	3,93	**7,78**	3,94	7,41	3,97
slp04	7,12	4,05	**7,59**	4,05	7,24	4,03
slp14	7,32	3,62	**8,19**	3,61	7,81	3,59
slp16	4,46	3,78	**5,21**	3,82	5,04	3,79
slp32	4,41	3,77	**5,28**	3,76	5,21	3,81
slp37	4,17	4,09	**4,85**	4,13	4,74	4,10
slp41	6,26	3,89	**7,13**	3,89	6,76	3,89
slp45	6,11	3,86	**6,82**	3,86	6,59	3,89
slp48	6,01	4,01	**6,50**	3,98	6,40	4,03
slp59	2,43	3,42	**4,42**	3,47	4,29	3,41
slp60	2,69	4,33	**7,11**	4,32	6,85	4,31
slp61	2,69	4,15	**6,57**	4,11	6,46	4,18
slp66	10,08	4,55	**12,51**	4,50	11,68	4,58
slp67x	9,35	4,51	**11,17**	4,50	10,41	4,51

Tabela 6.13: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para BPs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp01a	22,97	3,28	**21,94**	3,24	21,19	3,33
slp01b	24,45	3,49	**30,56**	3,51	28,08	3,46
slp02a	21,61	3,41	**25,31**	3,36	25,09	3,39
slp02b	20,43	3,67	**23,28**	3,69	22,57	3,69
slp03	21,63	3,53	**24,85**	3,51	24,55	3,49
slp04	25,94	3,47	**28,95**	3,47	27,50	3,52
slp14	25,94	4,12	**29,05**	4,09	27,12	4,09
slp16	24,49	3,78	**25,37**	3,74	24,36	3,80
slp32	24,22	3,04	**27,04**	3,01	25,63	3,02
slp37	21,85	4,05	**23,96**	4,01	23,32	4,03
slp41	28,25	3,73	**30,62**	3,76	28,64	3,71
slp45	20,63	3,21	**22,37**	3,24	21,37	3,21
slp48	23,09	3,07	**25,99**	3,04	24,95	3,08
slp59	27,15	3,92	**28,88**	3,95	27,09	3,89
slp60	23,82	3,84	**28,10**	3,82	26,52	3,80
slp61	22,78	3,53	**26,69**	3,49	25,56	3,56
slp66	22,09	3,81	**25,20**	3,76	24,46	3,85
slp67x	21,44	3,27	**23,21**	3,23	22,43	3,24

Tabela 6.14: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para os EEGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp01a	6,91	4,08	**7,55**	4,04	7,47	4,07
slp01b	2,65	3,50	**3,16**	3,49	3,16	3,51
slp02a	3,42	4,20	**4,68**	4,19	4,63	4,17
slp02b	2,93	3,54	**3,59**	3,55	3,58	3,58
slp03	4,44	3,54	5,89	3,53	**5,93**	3,58
slp04	3,69	3,69	**4,44**	3,68	4,43	3,70
slp14	3,51	3,81	**4,90**	3,83	4,88	3,83
slp16	3,16	3,63	**4,17**	3,62	4,13	3,59
slp32	3,05	3,61	**3,88**	3,58	3,87	3,58
slp37	2,13	3,59	**2,77**	3,59	2,75	3,55
slp41	3,01	3,63	**4,30**	3,62	4,28	3,61
slp45	4,29	4,01	**5,72**	4,05	5,58	3,97
slp48	3,02	3,72	**3,88**	3,71	3,84	3,68
slp59	3,95	4,55	**5,82**	4,55	5,78	4,54
slp60	2,62	4,07	**3,85**	4,07	3,84	4,07
slp61	4,77	4,00	**6,37**	3,99	6,27	3,98
slp66	3,17	4,16	**4,18**	4,18	4,17	4,21
slp67x	4,78	3,92	5,66	3,88	**5,67**	3,95

Tabela 6.15: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para sinais de RSP (abdominal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp37	**108,92**	3,86	**80,05**	3,79	73,18	3,78
slp41	**111,93**	4,14	**85,44**	4,16	78,97	4,16
slp45	**160,17**	4,13	**112,98**	3,99	100,30	3,99
slp59	24,82	3,59	**100,53**	3,55	94,22	3,57
slp60	26,00	3,59	**98,98**	3,58	88,02	3,56
slp61	**204,74**	4,16	**133,54**	3,70	111,67	3,70
slp66	**128,11**	4,40	**89,64**	4,44	81,01	4,44

Tabela 6.16: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para sinais de RSP (peitoral) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp32	**212,97**	3,85	**109,25**	2,85	100,26	2,85
slp48	4,76	3,01	**12,61**	2,98	12,60	3,02
slp67x	**227,31**	4,23	**137,30**	3,71	121,24	3,69

Tabela 6.17: Comparação das RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para sinais de RSP (nasal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp02a	**31,55**	4,07	**87,31**	4,11	84,64	4,05
slp02b	32,76	4,21	**85,25**	4,24	83,02	4,20
slp03	**87,03**	4,10	**67,91**	4,10	60,89	4,11
slp04	**133,55**	4,07	109,10	4,09	90,90	4,17
slp14	14,87	3,82	**39,27**	3,79	37,15	3,79
slp16	**120,19**	4,11	90,81	4,14	80,60	4,11
slp32	24,94	4,14	**54,89**	4,12	52,31	4,12
slp37	10,57	3,85	**14,77**	3,88	14,48	3,84
slp41	**82,11**	4,20	**60,74**	3,20	59,36	3,20
slp45	14,44	3,60	**28,81**	3,63	28,14	3,60
slp48	**173,62**	3,98	**113,98**	3,36	98,29	3,36
slp59	**200,35**	4,34	**105,31**	3,04	91,23	3,04
slp60	11,16	3,41	**55,91**	3,43	53,95	3,39
slp66	**95,88**	4,08	72,73	4,07	66,14	4,07
slp67x	28,86	4,11	**70,34**	4,10	65,71	4,08

Tabela 6.18: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para sinais de RSP (soma) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp01a	**162,87**	4,05	**114,19**	3,98	92,09	3,98
slp01b	**117,57**	4,23	**116,50**	3,68	100,19	3,68

Tabela 6.19: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para EOGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp32	5,66	3,74	**7,82**	3,79	7,72	3,77
slp37	7,06	4,10	**10,47**	4,09	10,34	4,09
slp41	29,32	3,99	34,68	3,95	**34,81**	4,00
slp45	12,77	3,79	**14,99**	3,82	14,87	3,81
slp48	12,31	4,26	**22,98**	4,28	22,63	4,24

Tabela 6.20: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para EMGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp32	1,86	3,79	**2,43**	3,76	2,42	3,75
slp37	1,95	3,55	**2,44**	3,54	2,43	3,55
slp41	2,37	3,40	**3,18**	3,42	3,15	3,36
slp45	2,29	3,79	**3,33**	3,79	**3,33**	3,81
slp48	2,04	3,75	**2,88**	3,74	**2,88**	3,77

Tabela 6.21: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para SVs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp59	20,83	3,72	**21,62**	3,72	21,33	3,72
slp60	17,24	3,73	**17,42**	3,17	17,19	3,17
slp61	16,93	2,99	**19,24**	2,98	18,77	2,96
slp66	14,52	3,65	**15,75**	3,61	15,62	3,61
slp67x	**13,83**	4,13	**13,63**	3,62	13,44	3,62

Tabela 6.22: Comparação de RCs e NPRDs dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag usando a *wavelet* Symmlet 10 para O_2Ss do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**		**DCT+KMinLag**	
	RC	**NPRD**	**RC**	**NPRD**	**RC**	**NPRD**
slp59	**818,05**	3,19	**267,79**	0,66	260,27	0,66
slp60	15,26	4,04	**596,96**	1,33	602,34	1,31
slp61	8,80	3,69	**483,58**	2,18	456,66	2,18
slp66	14,41	3,94	**269,99**	3,92	255,72	3,93
slp67x	14,10	3,91	**401,24**	1,76	374,40	1,76

A Tabela 6.23 mostra as médias dos tempos de codificação e decodificação dos sinais usando os métodos DWT+SPIHT, DCT+MinLag e DCT+KMinLag.

Tabela 6.23: Comparação de tempo de codificação e decodificação dos compressores DWT+SPIHT, DCT+MinLag e DCT+KMinLag com a *wavelet* Symmlet 10.

Canal de Dados	**Tempo de Codificação (segundos)**			**Tempo de Decodificação (segundos)**		
	DWT+ SPIHT	**DCT+ MinLag**	**DCT+ KMinLag**	**DWT+ SPIHT**	**DCT+ MinLag**	**DCT+ KMinLag**
ECG	125,5	**67,0**	117,3	80,4	**13,8**	15,3
BP	**48,5**	94,7	175,8	14,7	**14,1**	15,5
EEG	146,9	**140,9**	285,9	106,7	14,2	**14,2**
RSP (abdominal)	**62,9**	197,6	384,7	19,9	**14,4**	14,9
RSP (chest)	**68,3**	308,5	654,5	33,4	**12,7**	13,4
RSP (nasal)	**59,9**	322,9	735,9	25,0	**13,5**	14,2
RSP (sum)	**22,6**	226,3	590,7	**6,3**	6,8	7,4
EOG	**100,9**	131,4	225,5	48,6	20,4	**19,6**
EMG	**193,6**	345,1	760,6	138,7	**19,1**	19,2
SV	**46,3**	92,3	212,4	15,6	13,0	**12,4**
O_2S	75,5	**75,4**	168,0	41,6	**11,7**	12,8

As Figuras 6.10, 6.11, 6.12, 6.13, 6.14, 6.15, 6.16, e 6.17 apresentam seções dos sinais biomédicos reconstruídos com os métodos DCT+MinLag e DWT+SPIHT.

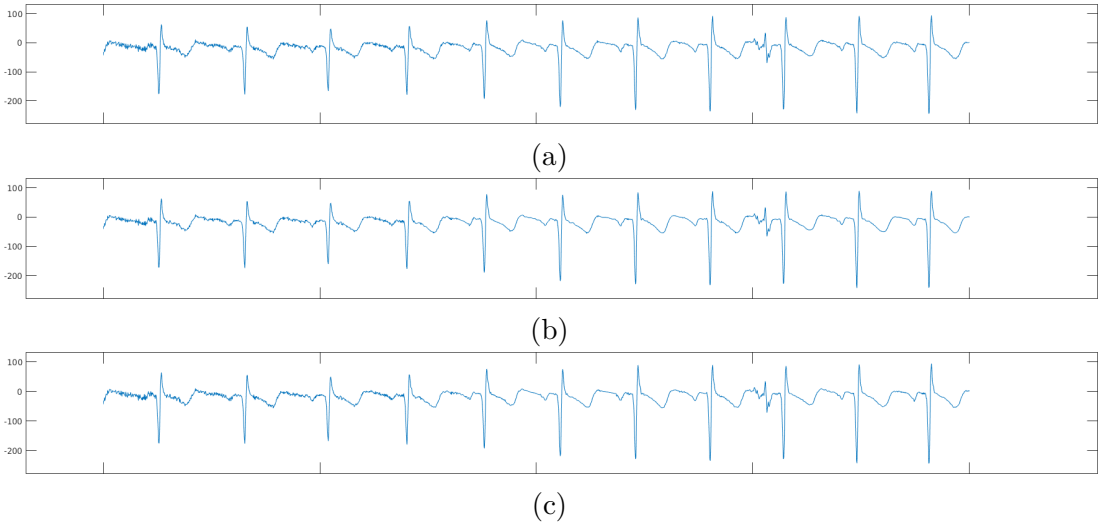
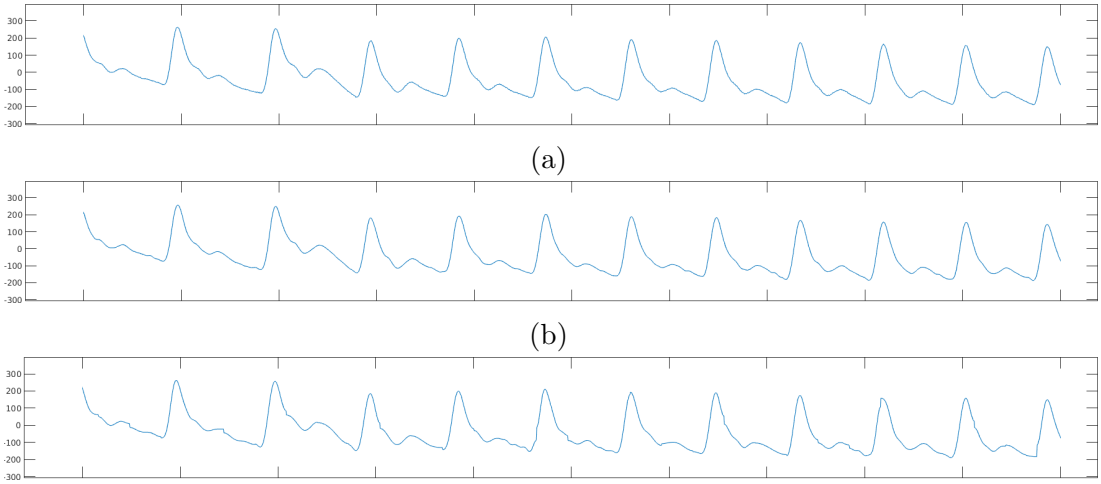


Figura 6.10: Seção entre 1h00m00s e 1h00m08s do canal de ECG do registro slp01b do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 4,24%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 4,21%.

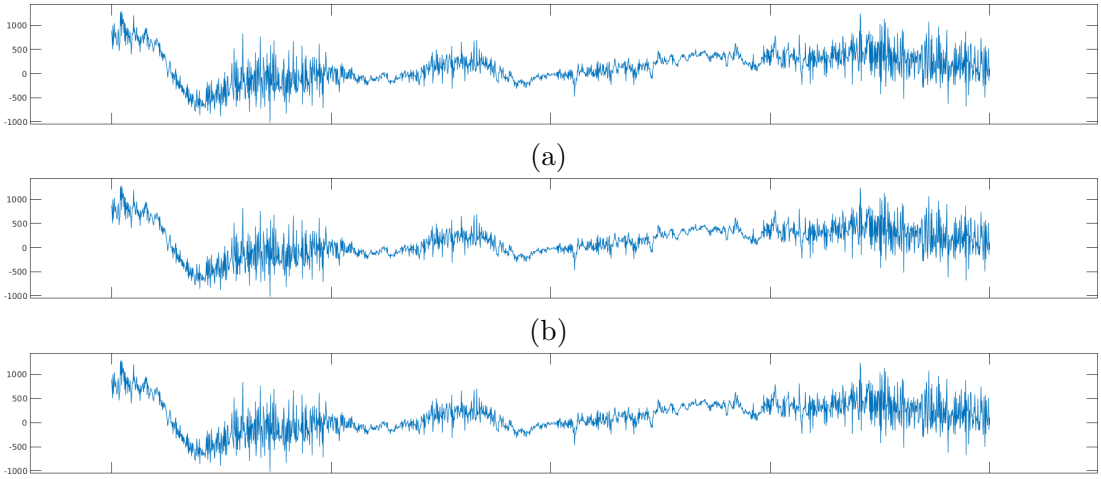


(a)

(b)

(c)

Figura 6.11: Seção entre 1h00m00s e 1h00m08s do canal de BP do registro slp01b do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 3,49%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 3,51%.



(a)

(b)

(c)

Figura 6.12: Seção entre 1h00m00s e 1h00m08s do canal de EEG do registro slp01b do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 3,50%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 3,49%.

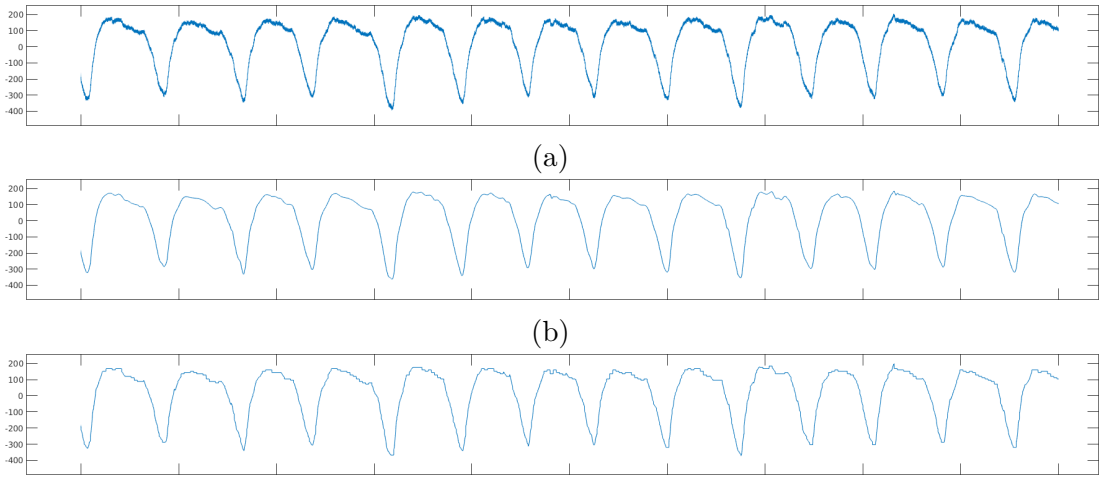


Figura 6.13: Seção entre 1h00m00s e 1h00m08s do canal de RSP do registro slp01b do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 4,23%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 2,85%.

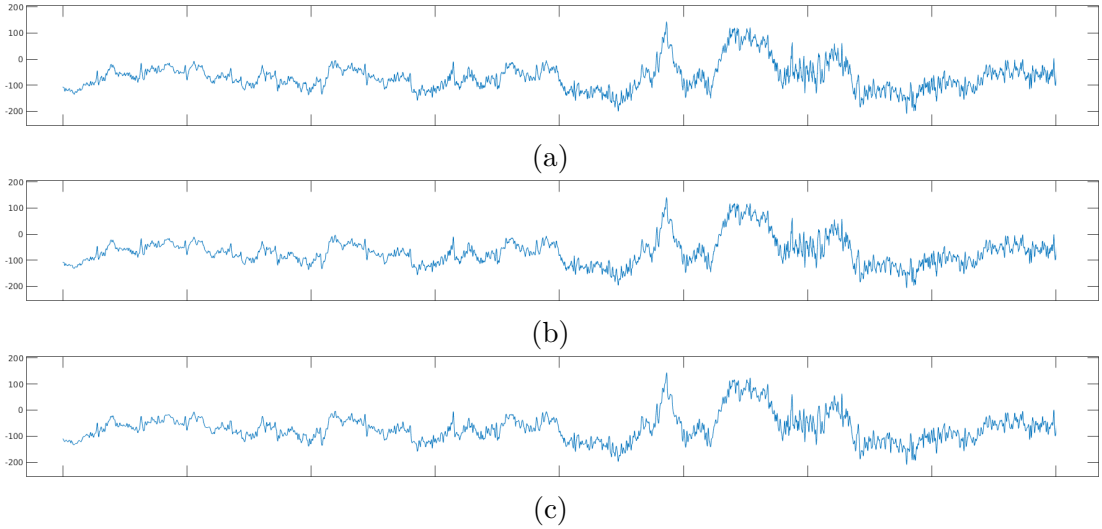


Figura 6.14: Seção entre 1h00m00s e 1h00m16s do canal de EOG do registro slp37 do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 4,10%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 4,09%.

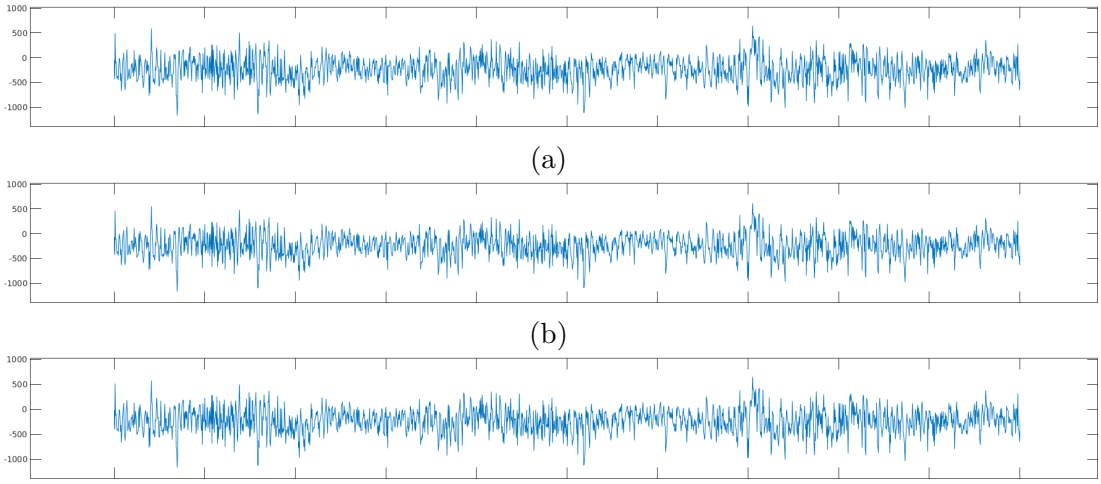


Figura 6.15: Seção entre 1h00m00s e 1h00m08s do canal de EMG do registro slp37 do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 3,55%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 3,54%.

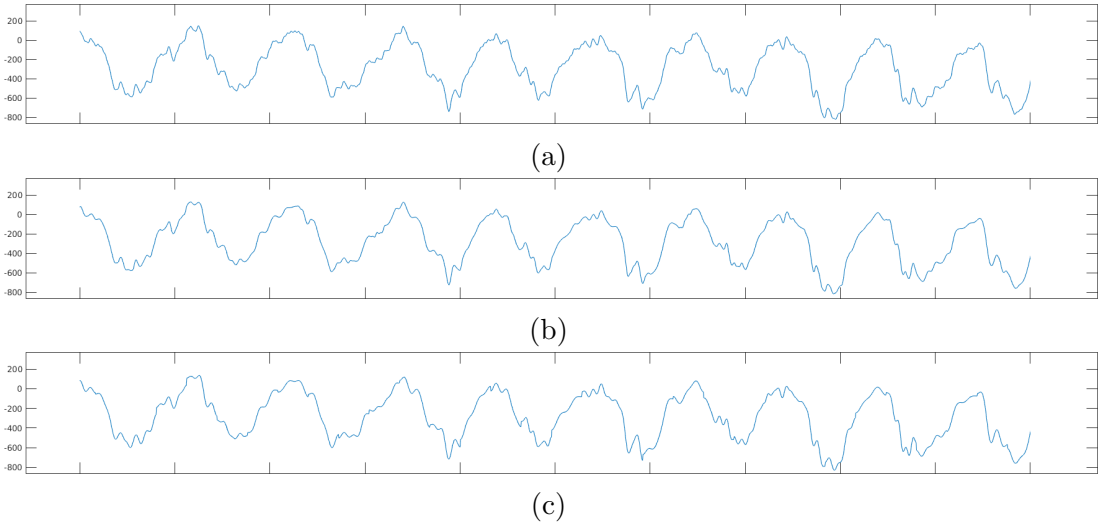


Figura 6.16: Seção entre 1h00m00s e 1h00m08s do canal de SV do registro slp60 do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 3,73%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 3,17%.

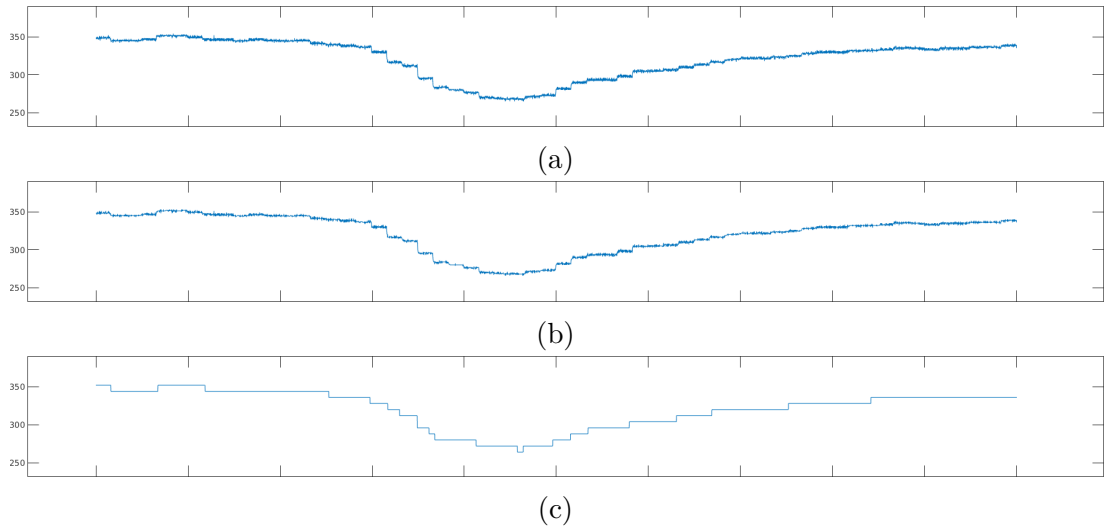


Figura 6.17: Seção entre 1h00m00s e 1h00m40s do canal de O_2S do registro slp60 do MIT/BIH PSGDB (a) original. (b) reconstruído após compressão com o método DWT+SPIHT e NPRD de 4,04%. (c) reconstruído após compressão com o método DCT+MinLag e NPRD de 1,33%.

6.4 Conclusão do Capítulo

O presente capítulo apresentou os resultados obtidos após a execução dos compressores *lossless* e *lossy* descritos no Capítulo 5. Por fim, o Capítulo 7 discutirá esses resultados e apresentará as conclusões obtidas.

Capítulo 7

Discussão e Conclusões

Esse trabalho introduziu várias estratégias de compressão *lossless* e *lossy* para sinais biomédicos unidimensionais. Métodos de transformação de domínio, predição linear e codificação de entropia foram testados, obtendo vários resultados que serão discutidos neste capítulo.

7.1 Discussão dos Métodos *Lossless*

De acordo com os resultados obtidos das estratégias *lossless*, fica claro pelas Tabelas 6.1 e 6.2 que o método PL obteve melhores *performances* de compressão que a IWT+SPIHT. Esse resultado indica que estratégias baseadas em Predição Linear são mais indicadas para a compressão de sinais biomédicos. Os resultados dos dois compressores *lossless* serão discutidos adiante.

As altas *performances* de compressão do PPM-C e do BPPM, mostradas na Figura 6.2, são explicadas pela alta eficiência dos modelos estatísticos contextuais sobre os símbolos passados para prever os símbolos futuros. O BPPM, diferentemente do PPM-C, utiliza as informações contextuais de cada plano de *bits* do sinal separadamente. Quando a correlação entre os *bits* nos planos é maior que a correlação entre

amostras vizinhas do sinal contendo todos os planos de *bits* – o que pode acontecer em sinais muito ruidosos ou muito imprevisíveis – o BPPM supera a eficiência do PPM-C. A estratégia binária obteve resultados notavelmente altos em comparação com os outros compressores, principalmente, em EEGs e EMGs.

Nota-se na Tabela 6.1 que, para o compressor IWT+SPIHT, as RCs melhoram modestamente à medida que o tamanho do bloco de transformação das IWTs é aumentado. Isso indica que, após um certo tamanho, o custo computacional envolvido no cálculo de blocos de transformação maiores não compensa o ganho pequeno de compressão. Outro fator que limitou o tamanho dos blocos de transformação a, no máximo, 2048 amostras, nos testes realizados, foi o aumento de complexidade computacional do SPIHT. Não houve uma única base *wavelet* ótima para a codificação de todos os tipos de sinais biomédicos testados, o que é natural devido às diferenças de amplitude, quantidade de ruído e formato dos diversos registros. A maioria das RCs obtidas por meio da aplicação do IWT+SPIHT ficaram entre 1,5 : 1 e 2,5 : 1, o que é uma compressão considerável, dada a simplicidade e a eficiência temporal do método, quando comparado com um codificador complexo e custoso como o PPM-C.

O fato da transformada S ter obtido a maior RC na compressão de sinais tão afetados por ruído como os O_2Ss indica que a tentativa de realizar predições mais complexas – tal qual a da transformada $(2 + 2{,}2)$ – em dados altamente ruidosos acaba por prejudicar a compressão. Isso se dá porque, à medida que a ordem da predição é aumentada, a magnitude dos erros de predição para sinais não contínuos tende a crescer rapidamente. Um resultado parecido pode ser visto na Tabela 6.2 não só para os O_2Ss, mas em todos os outros canais de dados, já que as predições ótimas foram, na grande maioria dos casos, de ordem 1 ($P1$). Os preditores de primeira ordem obtiveram RCs entre 1,117 : 1 e 4,818 : 1 usando o PPM-C como codificador de entropia. A maioria das RCs esteve no intervalo entre 2 : 1 e 3 : 1, provando a eficácia desses codificadores na codificação *lossless* dos PSGs. O BitGol obteve resultados insatisfatórios, o que

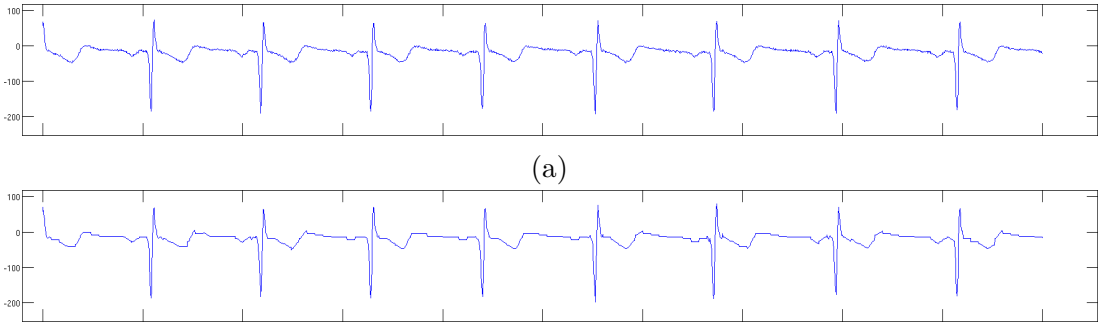
pode ser explicado pela natureza gaussiana da distribuição estatística dos dados, já que compressores baseados em codificadores de Golomb são ótimos para compactar sinais com distribuições estatísticas geométricas.

7.2 Discussão do Método DCT+MinLag

Para o compressor DCT+MinLag, a maioria dos sinais – ECGs, EEGs, EOGs, EMGs e SVs – ainda apresentaram boas qualidades de reconstrução visual com NPRDs de até 5,0%, indicando que testes com NPRDs mais altas devem ser executados para descobrir seus pontos de saturação. Esses resultados de qualidade visual corroboram o compromisso dessa estratégia de compressão com a qualidade de reconstrução de sinais biomédicos, já que, mesmo para as maiores NPRD usadas, 5 tipos de canais de um total de 8 ainda foram decodificados sem a geração de artefatos artificiais visivelmente marcantes.

Alguns sinais que contêm altas amplitudes, principalmente RSPs e BPs, sofrem fortes efeitos de limiarização nas fronteiras entre os blocos DCT após reconstruções com NPRDs mais altas, como mostrado nas Figuras B.2 e B.4. Esses artefatos são causados pelo alto desvio padrão σ que é obtido a partir dos picos e vales de alta magnitude nos traçados desses sinais. Como a fórmula para o cálculo da NPRD usa o desvio padrão para determinar a distorção objetiva, se o valor de σ para esses sinais for muito alto, maiores quantidades de erros serão aceitáveis para um mesmo nível de NPRD. Os altos valores de RC apresentados para RSPs e BPs, mesmo quando o nível de distorção é pequeno, confirmam que a estratégia de compressão baseada na qualidade do sinal também funciona de forma aceitável para esses canais de dados. Levando em consideração esses resultados, valores menores de NPRD devem ser passados para comprimir BPs e RSPs, quando comparados com as NPRDs de sinais como ECGs e EEGs. A Figura 7.1 mostra que efeitos de limiarização similares são observados em

outros sinais – no caso, ECGs – quando se usam valores de distorção mais altos no processo de compressão *lossy* ótima. Em outras palavras, a diferença em termos de qualidade de reconstrução entre sinais de alta e baixa amplitude é a velocidade em que eles se degradam à medida que os valores de distorção objetiva crescem. Esses resultados confirmam o que foi indicado por Zigel et al. (1997): medidas de distorção objetiva como a PRD e a NPRD não são perfeitamente relacionadas com a qualidade percebida através de verificações visuais subjetivas, mas são frequentemente usadas como aproximações.



(a)

(b)

Figura 7.1: Seção entre 1h00m00s e 1h00m08s do canal de ECG do registro slp01a do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 10,0%.

Os O_2Ss foram os únicos que não atingiram boas *performances* com o uso da quantização vetorial otimizada por meio da Minimização Lagrangiana. Esse mal funcionamento provavelmente está relacionado com a grande quantidade de ruído de alta frequência presente nesses canais de dados, como mostrado na Figura 7.2. As Figuras 7.2c e 7.2d apresentam respectivamente as reconstruções de um O_2S codificado otimamente com NPRDs de 0,5% e 0,75%. É possível ver nessas imagens de reconstrução que grande parte do ruído de alta frequência é eliminado dos registros, o que pode inclusive auxiliar um algoritmo que vá processar esses sinais e/ou um processo de verificação visual por um médico. O problema de reconstrução nesses canais de dados se dá por causa da forte presença do Fenômeno de Gibbs (Gibbs, 1898) nas regiões de alta frequência do

sinal. Esse fenômeno pode ser observado em qualquer transformada de domínio baseada em Séries de Fourier – como a DCT – e é acentuado quando são aplicadas quantizações aos coeficientes transformados. A NPRD de 0,134% – Figura 7.2a – foi a menor que a Minimização Lagrangiana conseguiu atingir e a reconstrução do sinal com essa distorção mantém a maior parte das características importantes do registro. Apesar dos O_2Ss terem sido comprimidos usando NPRDs entre 0,134% e 1,072%, as quais são quantidade de distorção pequenas, todos os registros desse tipo de canal obtiveram RCs entre 5,979 : 1 e 7,103 : 1. Os O_2Ss sofrem de um fenômeno parecido com a degradação sofrida por RSPs e BPs, mas apresentam uma perda de qualidade muito mais rápida com o aumento da NPRD.

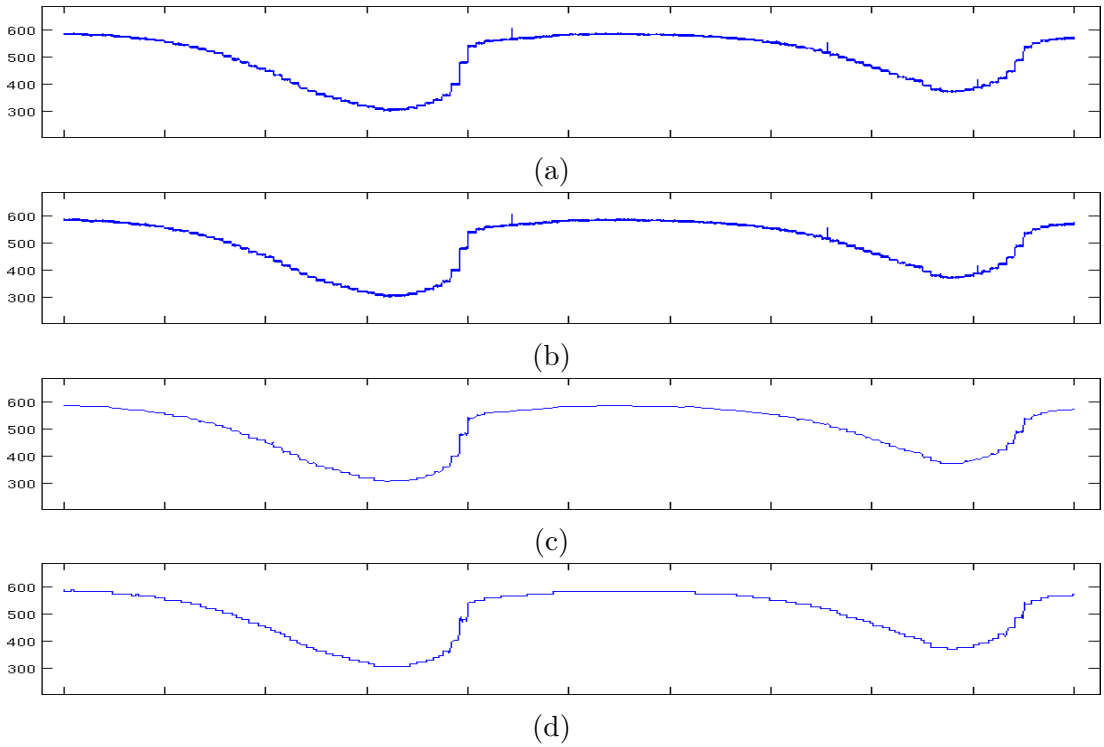


Figura 7.2: Seção entre 1h30m00s e 1h31m20s do canal de O_2S do registro slp60 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 0,134%. (c) reconstruído com uma NPRD de 0,5%. (d) reconstruído com uma NPRD de 0,75%.

As Figuras 6.3, 6.4, 6.5 e 6.6 mostram as evoluções de RC obtidas pelo PPM-C

e pelo BitGol à medida que se varia a NPRD. Como esperado, as curvas dos valores de RC tendem a aumentar com o aumento do valor de NPRD usado como parâmetro de compressão. A grande maioria dos resultados obtidos por esse trabalho se mostrou visivelmente superior aos resultados apresentados por Oliveira et al. (2014), com o codificador BitGol sendo superado apenas em algumas NPRDs nos RSPs – Figura 6.6.

O PPM-C obteve os melhores resultados na maioria dos casos, mas o BitGol o superou nos EOGs e parcialmente nos SVs – dependendo da NPRD – enquanto que os traçados dos dois codificadores ficaram muito próximos para EEGs e EMGs. Nos canais de dados em que o BitGol foi superado, porém, ele esteve muito próximo para valores de NPRD mais baixos. Dessa forma, esse codificador mostrou-se uma alternativa bem menos custosa e quase tão eficiente quanto o PPM-C para a codificação de entropia de sinais biomédicos unidimensionais, especialmente, quando recursos computacionais – como espaço em memória primária e poder de processamento – são escassos. Essas características reforçadas pelos resultados de tempo de execução apresentados na Tabela 6.11 tornam o BitGol um codificador ideal para a implementação de sistemas embarcados e/ou que precisem realizar compressão em tempo real.

Como era de se esperar, os resultados de compressão *lossy* obtiveram valores de RC claramente melhores que os resultados de compressão *lossless*. Esse compromisso entre confiabilidade de reconstrução e *performance* deve ser estudado de acordo com a finalidade que se deseja dar para os sinais biomédicos. A criação de um banco de dados nacional unificado de sinais biomédicos, por exemplo, idealmente deveria contar com estratégias de compressão *lossless*; porém o espaço de armazenamento necessário poderia vir a ser proibitivo. Para esse caso, compressões *lossy* com baixas distorções poderiam ser usadas. Uma NPRD praticamente imperceptível de 1,0% num BP, por exemplo, promove um aumento de RC por um fator próximo a 2, comparado com um compressor *lossless*. Para os mesmos sinais, o aumento da NPRD para 1,5% amplifica esse fator de ganho de RC para mais que 3, ainda apresentando diferenças mínimas

para o sinal original. Essas diferenças de *performance* de compressão ficam ainda mais marcantes em sinais de difícil compressão, como os ECGs, os EEGs, os EOGs e os EMGs. Para as estratégias *lossless*, esses sinais obtiveram, respectivamente, RCs máximas de 2,499 : 1, 1,744 : 1, 2,918 : 1 e 1,218 : 1, enquanto que para NPRDs de 5,0% na estratégia DCT+MinLag apresentaram reconstruções muito parecidas com os sinais originais e RCs de 8,585 : 1, 5,654 : 1, 23,806 : 1 e 3,271 : 1.

RSPs, por sua vez, apesar de sofrerem uma degradação de qualidade muito rápida, ainda podem ser comprimidos usando estratégias *lossy* com NPRDs de 2,0%, resultando em aumentos de RC por um fator 11, comparados com as estratégias *lossless*.

Já os SVs apresentaram relativamente altas RCs nos resultados *lossless* – chegando a 3,209 : 1 com o uso do PPM-C. Apesar de NPRDs de 5,0% já produzirem uma grande quantidade de distorções visuais nos SVs, NPRDs de 2,0% e até 3,0% resultam em boas qualidades de reconstrução visual e com RCs de 13,365 : 1 e 15,379 : 1, respectivamente.

Canais de dados contendo O_2Ss foram os que tiveram os menores ganhos de RC, entre as técnicas *lossless* e *lossy*, por causa da metodologia que necessitou ser aplicada devido à rápida degeneração desses registros. Ainda assim, a diferença dos sinais comprimidos de forma *lossy* são praticamente imperceptíveis. Caso a única característica desses sinais que necessite ser armazenada seja a forma das ondas para uma determinada aplicação, NPRDs maiores podem ser passadas para o método, aumentando drasticamente a taxa de compressão, como afirma Oliveira et al. (2014).

7.3 Comparação dos Métodos DWT+SPIHT, DCT+MinLag e DCT+KMinLag

Pode ser observado nas Tabelas 6.12, 6.13, 6.14, 6.15, 6.16, 6.17, 6.18, 6.19, 6.20, 6.21 e 6.22 que a DCT se comportou melhor na grande maioria dos resultados, obtendo RCs maiores para NPRDs semelhantes.

Como pode ser visto na Tabela 6.23, o método DCT+KMinLag se mostrou desnecessariamente dispendioso, já que obteve RCs menores que as do método DCT+MinLag e um custo computacional aproximadamente duas vezes maior devido ao custo das duas operações de Minimização Lagrangiana que são necessárias para a execução do método com $K = 2$. Por esse motivo, esse método foi considerado ineficaz e não foi incluído nos testes mais extensivos apresentados no Apêndice C.

De acordo com os resultados da Tabela 6.23, o método DCT+MinLag obteve tempos de codificação maiores que o DWT+SPIHT na maioria dos casos. Esses resultados podem ser explicados pelo dispendioso processo de otimização de histogramas quantizados da Minimização Lagrangiana, enquanto o método DWT+SPIHT é composto apenas de uma transformada *wavelet* – de execução quase instantânea – e um SAQ (Seção 4.3.6). Porém, por se tratar de um CODEC assíncrono, praticamente todos os tempos de decodificação foram menores para os métodos baseados em DCT e Minimização Lagrangiana, dado que esse processo não necessita de otimização – etapa mais custosa do processo de codificação.

As Figuras 6.11, 6.13, 6.16 e 6.17 demonstram partes de reconstruções de BPs, RSPs, SVs e O_2Ss. Observa-se que o método DWT+SPIHT preserva de forma mais suave a linha de base dos sinais. É importante ressaltar que, mesmo para NPRDs semelhantes, o BP apresentou artefatos de reconstrução altamente prejudiciais nas fronteiras dos blocos após o compressor DCT+MinLag, enquanto o método DWT+SPIHT conseguiu reconstruir suavemente o sinal. O mesmo efeito, embora um pouco mais sutil, é observado nas reconstruções de SVs, mesmo com o método DWT+SPIHT codificando o sinal com uma NPRD maior. Em adição, um forte efeito de limiarização foi observado nas Figuras 6.13c e 6.17c e não nas Figuras 6.13b e 6.17b, apesar de a DWT+SPIHT ter codificado o sinal com NPRD notavelmente maior que a DCT+MinLag. Dessa forma, apesar de ter obtido RCs consideravelmente menores na maioria dos casos, o método DWT+SPIHT é mais indicado para a compressão de sinais mais periódicos e com

variações mais suaves por causa de sua característica de preservação de coeficientes de baixa frequência e, consequentemente, de preservação da linha de base. Esses resultados podem ser explicados pela tendência do codificador SPIHT de dar preferência aos coeficientes de alta hierarquia da DWT no seu processo de codificação, quantizando de forma mais intensa os detalhes – como, por exemplo, o ruído de alta frequência de alguns sinais.

Outro fator que pode explicar a incapacidade da DCT de representar fielmente sinais de BP, RSP e O_2S usando NPRDs mais altas pode ser visto no Apêndice A. Principalmente para os níveis DC desse sinais, as CDFs Empíricas diferem consideravelmente das CDFs Normais, o que desvia das definições de Rao e Yip (2014) e Jayant e Noll (1990), os quais consideram distribuições Gaussianas ou Rayleigh para os níveis DC e distribuições Laplacianas para os níveis AC.

Em sinais menos dependentes da linha base, os artefatos de quantização dos compressores DWT+SPIHT e DCT+MinLag foram visualmente menos perceptíveis e equivalentes, como pode ser visto nas Figuras 6.10, 6.12, 6.14 e 6.15. Os resultados visuais apresentados na seção presente reforçam a afirmativa de Zigel et al. (1997) que ressaltam que a NPRD não indica com precisão a qualidade da reconstrução de um sinal.

7.4 Trabalhos Futuros

Como mencionado no Capítulo 5, algumas outras técnicas de Compressão de Dados e Processamento de Sinais foram consideradas promissoras para a codificação de sinais biomédicos unidimensionais. Algumas delas serão detalhadas nesta seção como trabalhos futuros previstos para continuidade a esse trabalho.

É possível introduzir um filtro passa-baixa nas fronteiras entre blocos DCT quantizados para suavizar os efeitos de limiarização encontrados em RSPs e BPs. Essa suavização funcionaria como os filtros de *deblocking* encontrados em muitos codifica-

dores de vídeo modernos, como o H.264 (ITU-T264, 2002) e o H.265 (Pourazad et al., 2012). Um esquema mais avançado poderia contar com um conjunto de filtros passa-baixa e decidir automaticamente qual deles seria ideal para cada fronteira de bloco, ou se seria sequer necessário aplicar uma operação de *deblocking*.

Outras transformadas também podem ser testadas, como a Transformada Inteira do Cosseno (*Integer Cosine Transform* – ICT), voltada para a compressão *lossless*; e algumas formas de cálculo acelerado para KLTs, como a descrita por Wang et al. (2009). Uma solução para tentar remover correlações entre os blocos transformados é executar as versões bidimensionais de cada transformação, o que pode vir a melhorar as RCs após a codificação de entropia.

Além de modificar as transformadas e os métodos de pré e pós-processamento, é possível realizar testes com outros codificadores de entropia, como os baseados em dicionários (Lempel e Ziv, 1976).

7.5 Considerações Finais

Esse trabalho apresentou esquemas de compressão *lossless* e *lossy* para sinais biomédicos. As estratégias *lossless* foram baseadas em preditores lineares e transformadas *wavelet*, enquanto os métodos *lossy* se basearam em transformadas trigonométricas, transformadas *wavelet* e quantização vetorial ótima. Os métodos foram testados usando os sinais presentes no MIT/BIH PSGDB da Physionet, obtendo os melhores resultados de compressão *lossless* com RCs de até 4,818 : 1. No contexto de compactadores *lossy*, as razões de compressão chegaram até 818,055 : 1, dependendo do sinal, da distorção e do método escolhidos. Os resultados apresentados comprovaram o compromisso do métodos *lossy* com a fidelidade de reconstrução dos sinais biomédicos, fazendo comparações entre os métodos baseados em *wavelets* e transformadas trigonométricas.

Verificou-se a eficácia de compressão de dois compressores baseados no paradigma

TQC – Seção 4.4 – no contexto de sinais biomédicos. Os resultados *lossy* foram comparados entre si e discutidos em conjunto com os resultados *lossless*, justificando a inserção de pequenos níveis de distorção nos sinais para alguns tipos de aplicação.

No contexto dos métodos de compressão *lossy*, chegou-se à conclusão que os sinais mais dependentes da linha de base – como os BPs, RSPs, SVs e O_2Ss – são comprimidos com menos artefatos visuais artificiais quando se utiliza o método DWT+SPIHT por causa da tendência do SPIHT de priorizar os níveis de maior hierarquia na codificação. Já sinais menos dependentes da linha base e com maiores variações – por exemplo, ECGs, EEGs, EOGs e EMGs – obtêm melhores RCs com o método DCT+MinLag, enquanto a distorção visual após os dois métodos permanece semelhante.

Embora testes mais abrangentes devam ser realizados para generalizar os resultados obtidos, as análises dos compressores *lossy* feitas nesse trabalho com relação à eficácia de compressão e qualidade de reconstrução podem ser extrapoladas para outros tipos de sinais biomédicos. O mesmo pode ser feito não apenas para sinais temporais, mas também para sinais com amostragens espaciais e com mais dimensões, como imagens bidimensionais, imagens volumétricas e sinais quadridimensionais, como, por exemplo, imagens de Ressonância Magnética Funcional (*Functional magnetic resonance imaging* – fMRI). Isso se dá porque as transformadas usadas – a DCT e a DWT – são separáveis, ou seja, seu cálculo para sinais com mais dimensões pode ser composto de iterações de transformadas unidimensionais.

Os resultados apresentados nesse trabalho proveem uma boa base de conhecimento inicial para a compressão tanto *lossless* quanto *lossy* de diversos tipos de sinais biomédicos, servindo de base para futuras pesquisas na área, a qual carece de fontes.

Referências Bibliográficas

Abenstein, J. P. e Tompkins, W. J. (1982). A new data-reduction algorithm for real-time ecg analysis. *Biomedical Engineering, IEEE Transactions on*, BME-29(1):43–48.

Antoniol, G. e Tonella, P. (1997). Eeg data compression techniques. *Biomedical Engineering, IEEE Transactions on*, 44(2):105–114.

Batista, L. V. (2002). *Compressão de Sinais Eletrocardiográficos Baseada na Transformada Cosseno Discreta*. PhD thesis, D. Sc. Thesis in Electrical Engineering, PPGEE/DEE/UFPB, Campina Grande, Brasil.

Bell, T. C., Cleary, J. G., e Witten, I. H. (1990). *Text Compression*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Brasileiro, J. J. e Cavalcanti, A. C. (2012). On-chip integration of a binary ppm algorithm for lossless compression of ecg: Design space exploration for low hardware complexity. *II Workshop on Circuits and System Design*.

Brechet, L., Lucas, M.-F., Doncarli, C., e Farina, D. (2007). Compression of biomedical signals with mother wavelet optimization and best-basis wavelet packet selection. *Biomedical Engineering, IEEE Transactions on*, 54(12):2186–2192.

Britanak, V., Yip, P. C., e Rao, K. R. (2006). *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations*.

Burrows, M. e Wheeler, D. J. (1994). A block-sorting lossless data compression algorithm. Report 124, Digital Systems Research Center, Palo Alto, CA, USA.

Calderbank, A., Daubechies, I., Sweldens, W., e Yeo, B.-L. (1998). Wavelet transforms that map integers to integers. *Applied and Computational Harmonic Analysis*, 5(3):332 – 369.

Cleary, J. G. e Witten, I. H. (1984). Data compression using adaptive coding and partial string matching. COM-32(4):396–402.

Cohen, A., Daubechies, I., e Feauveau, J.-C. (1992). Biorthogonal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 45(5):485–560.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2009). *Introduction to Algorithms, 3rd Edition*. The MIT Press, 3rd edition.

Cover, T. M. e Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

da Silva, E. A., DA Fonini, J., e Craizer, M. (2002). Successive approximation quantization for image compression. *Circuits and Systems Magazine, IEEE*, 2(3):20–45.

Daubechies, I. (1988). Orthonormal bases of compactly supported wavelets. 41:909–996.

de A.Berger, P., de O.Nascimento, F., da Rocha, A., e Carvalho, J. (2007). A new wavelet-based algorithm for compression of emg signals. Em *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pgs. 1554–1557.

Dempster, A. P., Laird, N. M., e Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pgs. 1–38.

Fano, R. M. (1949). The transmission of information. Technical Report 65, Research Laboratory for Electronics, MIT, Cambridge, MA, USA.

Fukunaga, K. e Hostetler, L. D. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40.

Gallager, R. G. e van Voorhis, D. C. (1975). Optimal source codes for geometrically distributed integer alphabets. IT-21(3):228–230.

Gersho, A. e Gray, R. M. (1992). *Vector quantization and signal compression*. Springer Science & Business Media.

Gibbs, J. W. (1898). Fourier's series. *Nature*, 59:200.

Golomb, S. W. (1966). Run-length encodings. IT-12(3):399–401.

Han, J., Saxena, A., e Rose, K. (2010). Towards jointly optimal spatial prediction and adaptive transform in video/image coding. Em *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pgs. 726–729.

Higgins, G., McGinley, B., Jones, E., e Glavin, M. (2010). Efficient eeg compression using jpeg2000 with coefficient thresholding. Em *Signals and Systems Conference (ISSC 2010), IET Irish*, pgs. 59–64.

Hoang, D. T. (1997). *Fast and efficient algorithms for text and video compression*. PhD thesis, Brown University.

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101.

Ichimaru, Y. e Moody, G. B. (1999). Development of the polysomnographic database on cd-rom. *Psychiatry and clinical neurosciences*, 53(2):175–177.

Istepanian, R. e Petrosian, A. (2000). Optimal zonal wavelet-based ecg data compression for a mobile telecardiology system. *Information Technology in Biomedicine, IEEE Transactions on*, 4(3):200–211.

ITU-T (1990). Recommendation G.726 (12/90): 40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM).

ITU-T264 (2002). ITU-T Recommendation H.264, ISO/IEC 11496-10: Advanced Video Coding. Final Committee Draft, Document JVT-E022.

Jargas, A. M. (2008). *Shell Script Profissional*. Novatec Editora.

Jayant, N. S. e Noll, P. (1990). *Digital Coding of Waveforms: Principles and Applications to Speech and Video*. Prentice Hall Professional Technical Reference.

JPEG 2000 Organization (2000). JPEG 2000 our new standard!

Karhunen, K. (1947). *Über lineare Methoden in der Wahrscheinlichkeitsrechnung*, volume 37. Universitat Helsinki.

Kedir-Talha, M.-D. e Amer, M. (2013). The lifted wavelet transform for encephalic signal compression. Em *Telecommunications and Signal Processing (TSP), 2013 36th International Conference on*, pgs. 541–544.

Kim, B., Yoo, S., e Lee, M. (2006). Wavelet-based low-delay ecg compression algorithm for continuous ecg transmission. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):77–83.

Lai, S.-C., Lan, C.-S., e Lei, S.-F. (2013). An efficient method of ecg signal compression by using a dct-iv spectrum. Em *Communications, Circuits and Systems (ICCCAS), 2013 International Conference on*, volume 1, pgs. 46–49. IEEE.

Lee, H. e Buckley, K. (1999). Ecg data compression using cut and align beats approach and 2-d transforms. *Biomedical Engineering, IEEE Transactions on*, 46(5):556–564.

Lempel, A. e Ziv, J. (1976). On the complexity of finite sequences. 22(1):75–81.

Loève, M. (1948). Fonctions aléatoires de second order. *Supplement to P. Levy, Processus Stochastiques et Mouvement Brownien. Gauthier-Villars: Paris.*

Lu, Z., Kim, D. Y., e Pearlman, W. A. (2000). Wavelet compression of ecg signals by the set partitioning in hierarchical trees algorithm. *Biomedical Engineering, IEEE Transactions on*, 47(7):849–856.

Martucci, S. A. (1992). Convolution-multiplication properties for the entire family of discrete sine and cosine transforms. Em *Proc. Twenty-Sixth Annual Conf. on Information Sciences and Systems.*

Meares, D. (1974). Differential pulse-code modulation. US Patent 3,800,225.

Memon, N., Kong, X., e Cinkler, J. (1999). Context-based lossless and near-lossless compression of eeg signals. *Information Technology in Biomedicine, IEEE Transactions on*, 3(3):231–238.

Miaou, S.-G. e Chao, S.-N. (2005). Wavelet-based lossy-to-lossless ecg compression in a unified vector quantization framework. *Biomedical Engineering, IEEE Transactions on*, 52(3):539–543.

Moody, G. e Mark, R. (2001). The impact of the mit-bih arrhythmia database. *Engineering in Medicine and Biology Magazine, IEEE*, 20(3):45–50.

Moody, G., Mark, R., e Goldberger, A. (2001). Physionet: a web-based resource for the study of physiologic signals. *Engineering in Medicine and Biology Magazine, IEEE*, 20(3):70–75.

Mukhopadhyay, S., Mitra, M., e Mitra, S. (2011). An ecg data compression method via r-peak detection and ascii character encoding. Em *Computer, Communication and*

Electrical Technology (ICCCET), 2011 International Conference on, pgs. 136–141. IEEE.

Mulcahy, C. (1996). Plotting and scheming with wavelets. 69(5):323–343.

Mulcahy, C. (1997). Image compression using the Haar wavelet transform. 1(1):22–31. It has been claimed that any smart 15-year-old could follow this introduction to wavelets.

Oeff, M., Koch, H., Bousseljot, R., e Kreiseler, D. (2012). The ptb diagnostic ecg database. *National Metrology Institute of Germany, http://www. physionet. org/physiobank/database/ptbdb*.

Oliveira, H., Silva, A., Diniz, I., Sampaio, G., e Batista, L. (2014). Compression of polysomnographic signals using the discrete cosine transform and dead-zone optimal quantization. Em *AICT 2014, The Tenth Advanced International Conference on Telecommunications*, pgs. 26–34.

Pourazad, M. T., Doutre, C., Azimi, M., e Nasiopoulos, P. (2012). Hevc: the new gold standard for video compression: how does hevc compare with h. 264/avc? *Consumer Electronics Magazine, IEEE*, 1(3):36–46.

Pradhan, N. e Dutt, D. (1994). Data compression by linear prediction for storage and transmission of {EEG} signals. *International Journal of Bio-Medical Computing*, 35(3):207 – 217.

Rabbani, M. e Jones, P. W. (1991). *Digital image compression techniques*, volume 7. SPIE Press.

Ranjeet, K., Kumar, A., e Pandey, R. K. (2013). Electronics & comm. engineering, pdpm indian institute of information technology, design & manufacturing, jabalpur,(mp)-

482005 india. Em *Control, Automation, Robotics and Embedded Systems (CARE), 2013 International Conference on*, pgs. 1–6. IEEE.

Rao, K. R. e Yip, P. (2014). *Discrete cosine transform: algorithms, advantages, applications*. Academic press.

Rao, K. R. e Yip, P. C. (2010). *The transform and data compression handbook*. CRC press.

Ratnakar, V. (1997). *Quality-controlled lossy image compression*. PhD thesis, Citeseer.

Ratnakar, V. e Livny, M. (1995). RD-OPT: an efficient algorithm for optimizing DCT quantization tables. pgs. 332–341.

Ratnakar, V. e Livny, M. (1996). Extending RD-OPT with global thresholding for JPEG optimization. pgs. 379–386.

Rice, R. F. (1979). Some practical universal noiseless coding techniques.

Richardson, I. E. (2010). *The H.264 Advanced Video Compression Standard*. John Wiley & Sons, Ltd.

Robinson, T. (1994). SHORTEN: Simple lossless and near-lossless waveform compression. Technical report CUED/F-INFENG/TR.15, Cambridge University Engineering Department, Trumpington Street, Cambridge, CB2 1PZ, UK.

Said, A. e Pearlman, W. A. (1996). A new fast and efficient image codec based on set partitioning in hierarchical trees. 6(6):243–250.

Salomon, D. (2006). *Data Compression: The Complete Reference*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Sayood, K. (1996). *Introduction to Data Compression*.

Schröder, P. e Sweldens, W. (1995). Spherical wavelets: Efficiently representing functions on the sphere. Em *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pgs. 161–172, New York, NY, USA. ACM.

Shannon, C. E. (1948). A mathematical theory of communication. 27(3):379–423.

Shannon, C. E. e Weaver, W. (1998). *The Mathematical Theory of Communication*. Preface by Richard Blahut and Bruce Hajek.

Shapiro, J. M. (1993). Embedded image coding using zerotrees of wavelet coefficients. *Signal Processing, IEEE Transactions on*, 41(12):3445–3462.

Srinivasan, K., Dauwels, J., e Reddy, M. R. (2013). Multichannel eeg compression: Wavelet-based image and volumetric coding approach. *Biomedical and Health Informatics, IEEE Journal of*, 17(1):113–120.

Stollnitz, E. J., DeRose, T. D., e Salesin, D. H. (1996). *Wavelets for Computer Graphics*. Morgan Kaufmann, San Francisco, CA.

Strassen, V. (1969). Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356.

Stroustrup, B. (1986). *The C++ programming language*. Pearson Education India.

Sullivan, G. J., Ohm, J., Han, W.-J., e Wiegand, T. (2012). Overview of the high efficiency video coding (hevc) standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(12):1649–1668.

Šušmáková, K. (2004). Human sleep and sleep eeg. *Measurement science review*, 4(2):59–74.

Sweldens, W. (1996). The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and computational harmonic analysis*, 3(2):186–200.

Sweldens, W. (1998). The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis*, 29(2):511–546.

Sweldens, W. e Schröder, P. (1996). Building your own wavelets at home. SIGGRAPH 96 Course Notes. Available on the WWW.

Wallace, G. K. (1991). The JPEG still image compression standard. 34(4):30–44.

Wang, L., Wu, J., Jiao, L., e Shi, G. (2009). 3d medical image compression based on multiplierless low-complexity rklt and shape-adaptive wavelet transform. Em *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pgs. 2521–2524.

Wang, Z. (1984). Fast algorithms for the discrete w transform and for the discrete fourier transform. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 32(4):803–816.

Welch, T. A. (1984). A technique for high-performance data compression. 17(6):8–19.

Wellig, P., Cheng, Z., Semling, M., e Moschytz, G. S. (1998). Electromyogram data compression using single-tree and modified zero-tree wavelet encoding. Em *Engineering in Medicine and Biology Society, 1998. Proceedings of the 20th Annual International Conference of the IEEE*, volume 3, pgs. 1303–1306 vol.3.

Witten, I. H., Frank, E., e Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Zigel, Y. e Cohen, A. (1999). On the optimal distortion measure for ecg compression. Em *EMBEC*, volume 99, pgs. 1618–19.

Zigel, Y., Cohen, A., Abu-Ful, A., Wagshal, A., e Katz, A. (1997). Analysis by synthesis ecg signal compression. Em *Computers in Cardiology 1997*, pgs. 279–282.

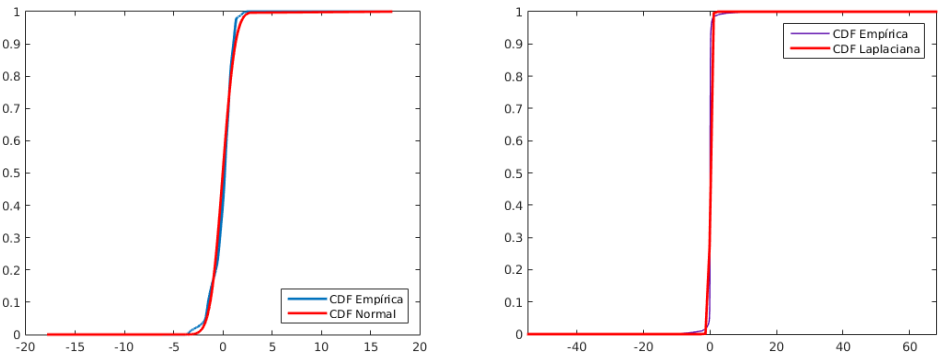
Zigel, Y., Cohen, A., e Katz, A. (2000). The weighted diagnostic distortion (wdd) measure for ecg signal compression. *Biomedical Engineering, IEEE Transactions on*, 47(11):1422–1430.

Zou, F. e Gallagher, R. (1993). Ecg data compression with wavelet and discrete cosine transforms. *Biomedical sciences instrumentation*, 30:57–62.

Apêndice A

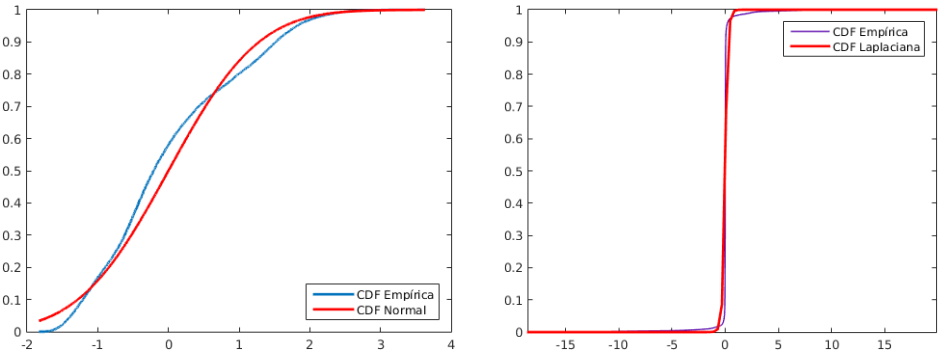
Funções de Distribuição Cumulativas de Sinais Biomédicos

As Figuras A.1, A.2, A.3, A.4, A.5, A.6, A.7, e A.8 demonstram as CDFs dos sinais biomédicos usados nesse trabalho após transformados por DCTs.



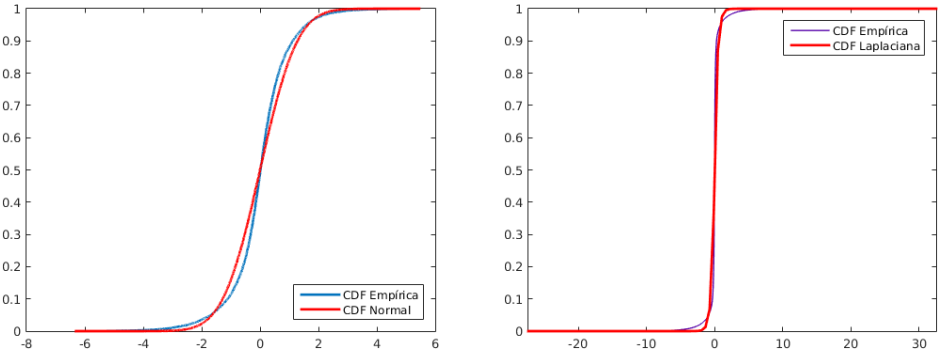
(a) (b)

Figura A.1: CDFs de coeficientes DCT do sinal de ECG do registro slp01a do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



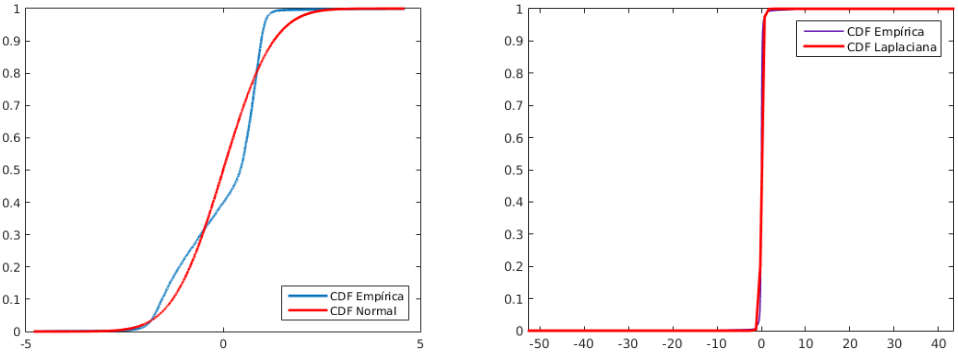
(a) (b)

Figura A.2: CDFs de coeficientes DCT do sinal de BP do registro slp01a do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



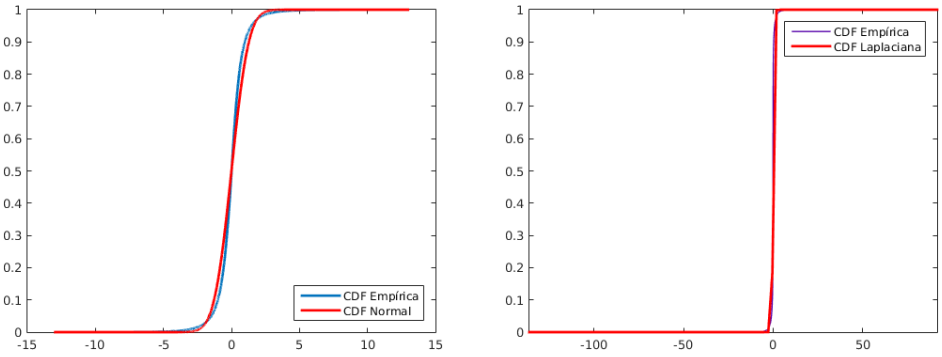
(a) (b)

Figura A.3: CDFs de coeficientes DCT do sinal de EEG do registro slp01a do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



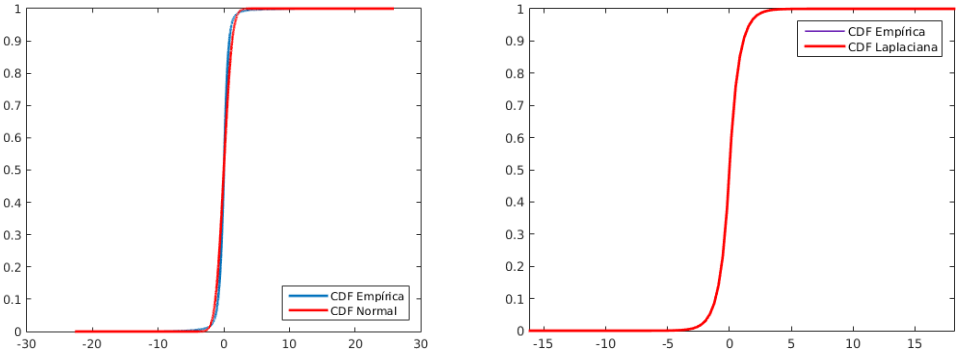
(a) (b)

Figura A.4: CDFs de coeficientes DCT do sinal de RSP do registro slp01a do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



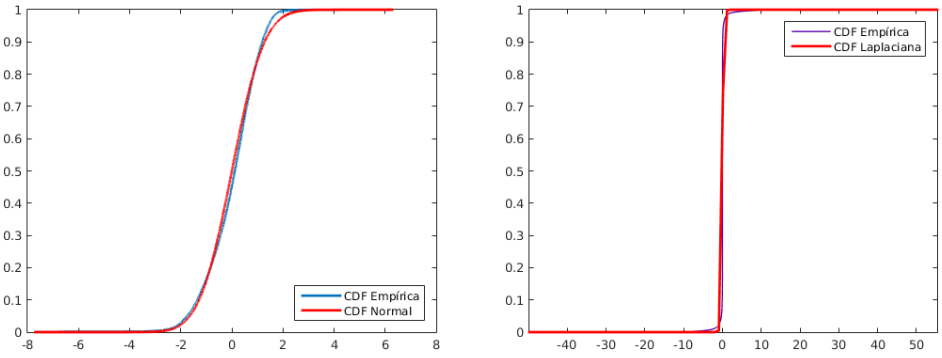
(a) (b)

Figura A.5: CDFs de coeficientes DCT do sinal de EOG do registro slp32 do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



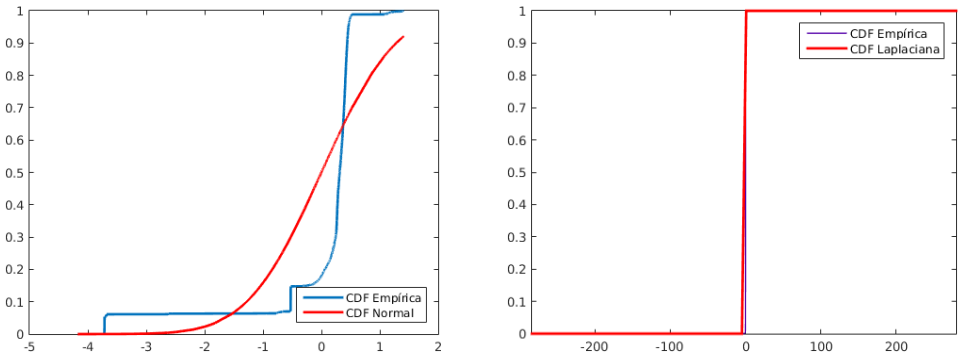
(a) (b)

Figura A.6: CDFs de coeficientes DCT do sinal de EMG do registro slp32 do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



(a) (b)

Figura A.7: CDFs de coeficientes DCT do sinal de SV do registro slp59 do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.



(a) (b)

Figura A.8: CDFs de coeficientes DCT do sinal de O_2S do registro slp59 do MIT/BIH PSGDB. (a) Coeficientes DC. (b) Coeficientes AC.

Apêndice B

Reconstruções de Seções dos Sinais Biomédicos

Este apêndice demonstra as imagens de reconstrução de algumas seções de sinais de todos os canais de dados do MIT/BIH PSGDB (Ichimaru e Moody, 1999) usando o compressor DCT+MinLag. Cada conjunto de figuras está disposto em uma página visando facilitar a inspeção visual dos artefatos gerados pela reconstrução *lossy*.

As Figuras B.1, B.2, B.3, B.4, B.5, B.6, B.7 e B.8 referem-se respectivamente a ECGs, BPs, EEGs, RSPs, EOGs, EMGs, SVs e O_2Ss.

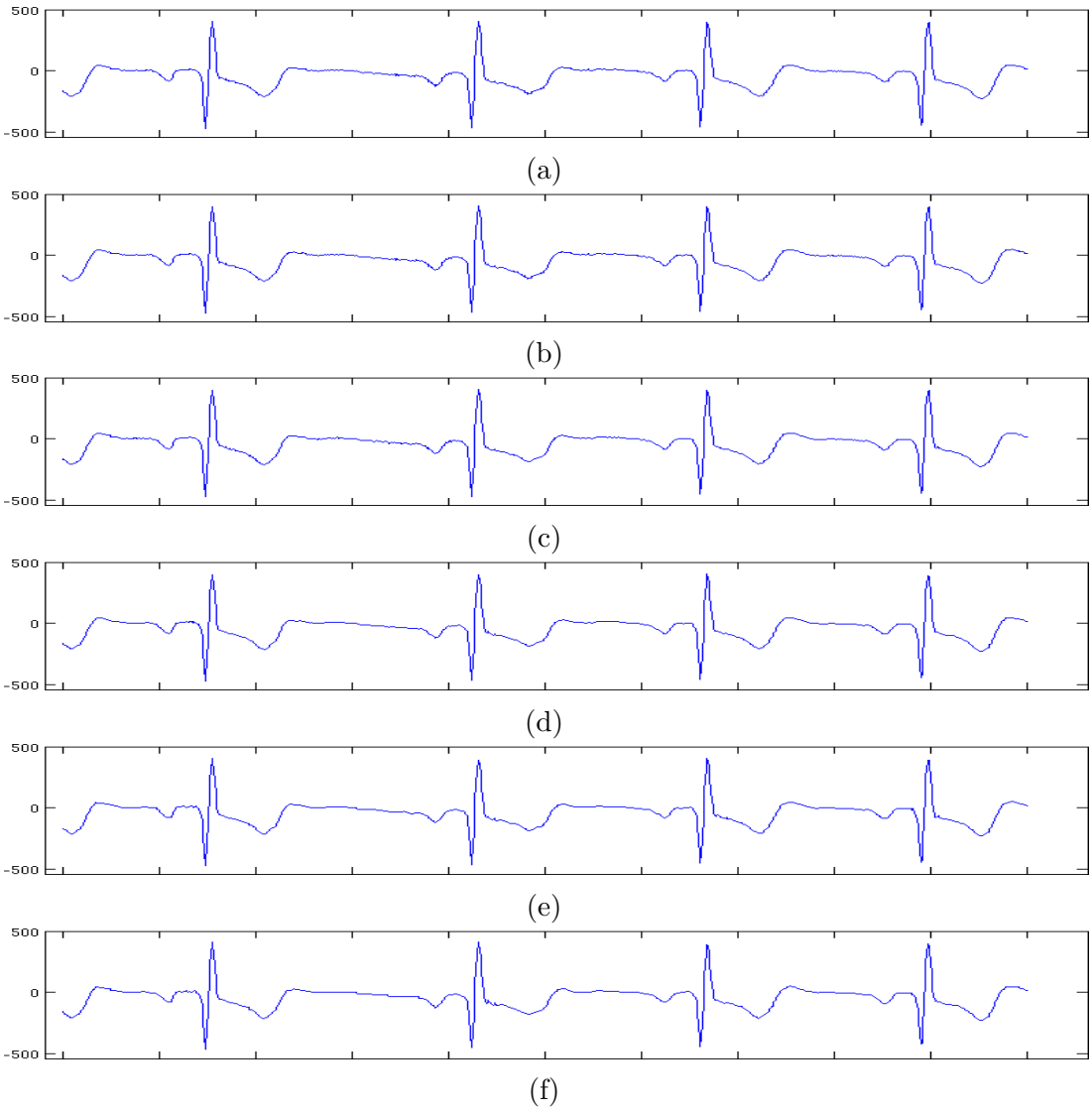


Figura B.1: Seção entre 1h00m00s e 1h00m04s do canal de ECG do registro slp16 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.

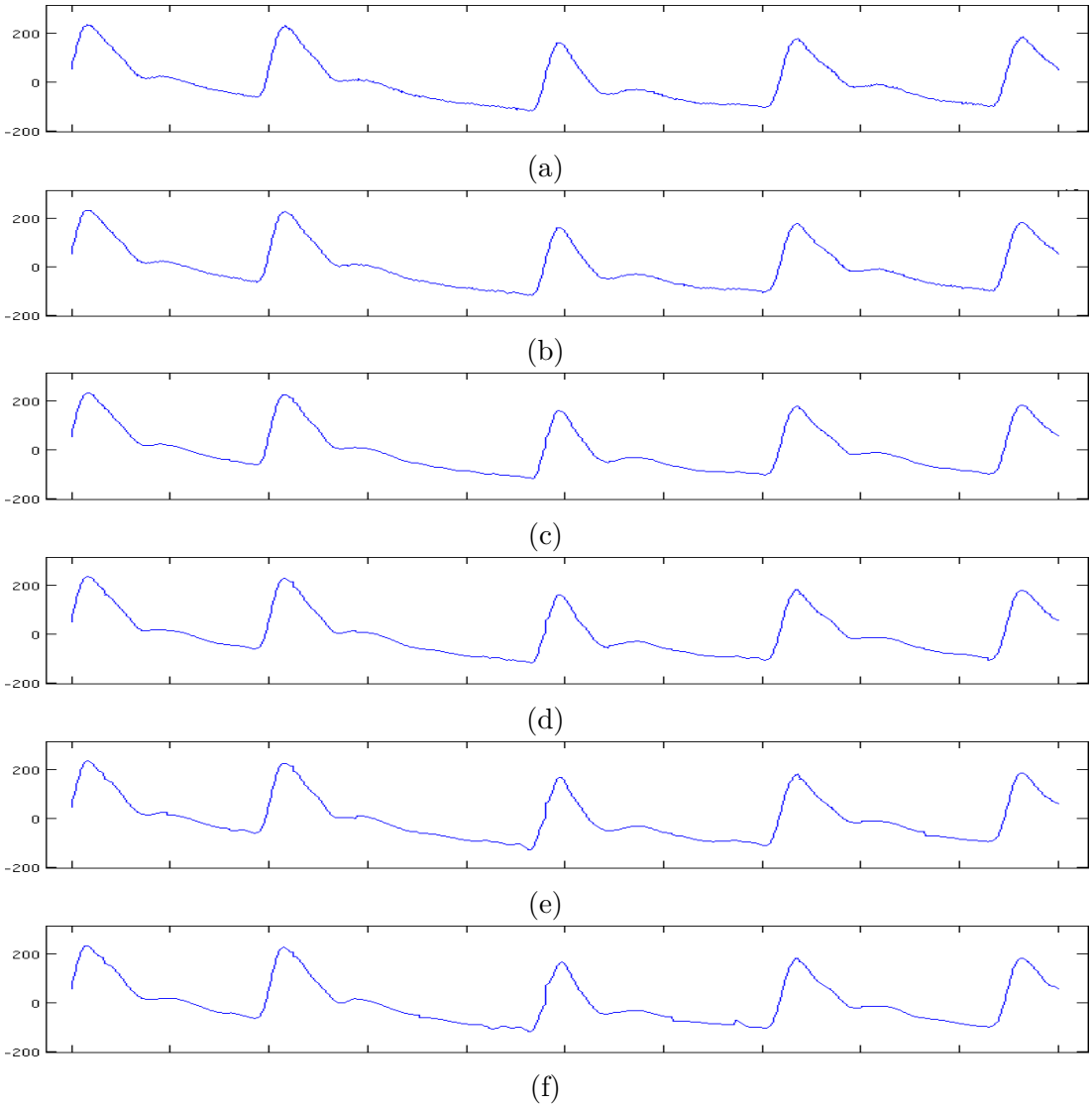


Figura B.2: Seção entre 1h00m00s e 1h00m04s do canal de BP do registro slp04 BP do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.

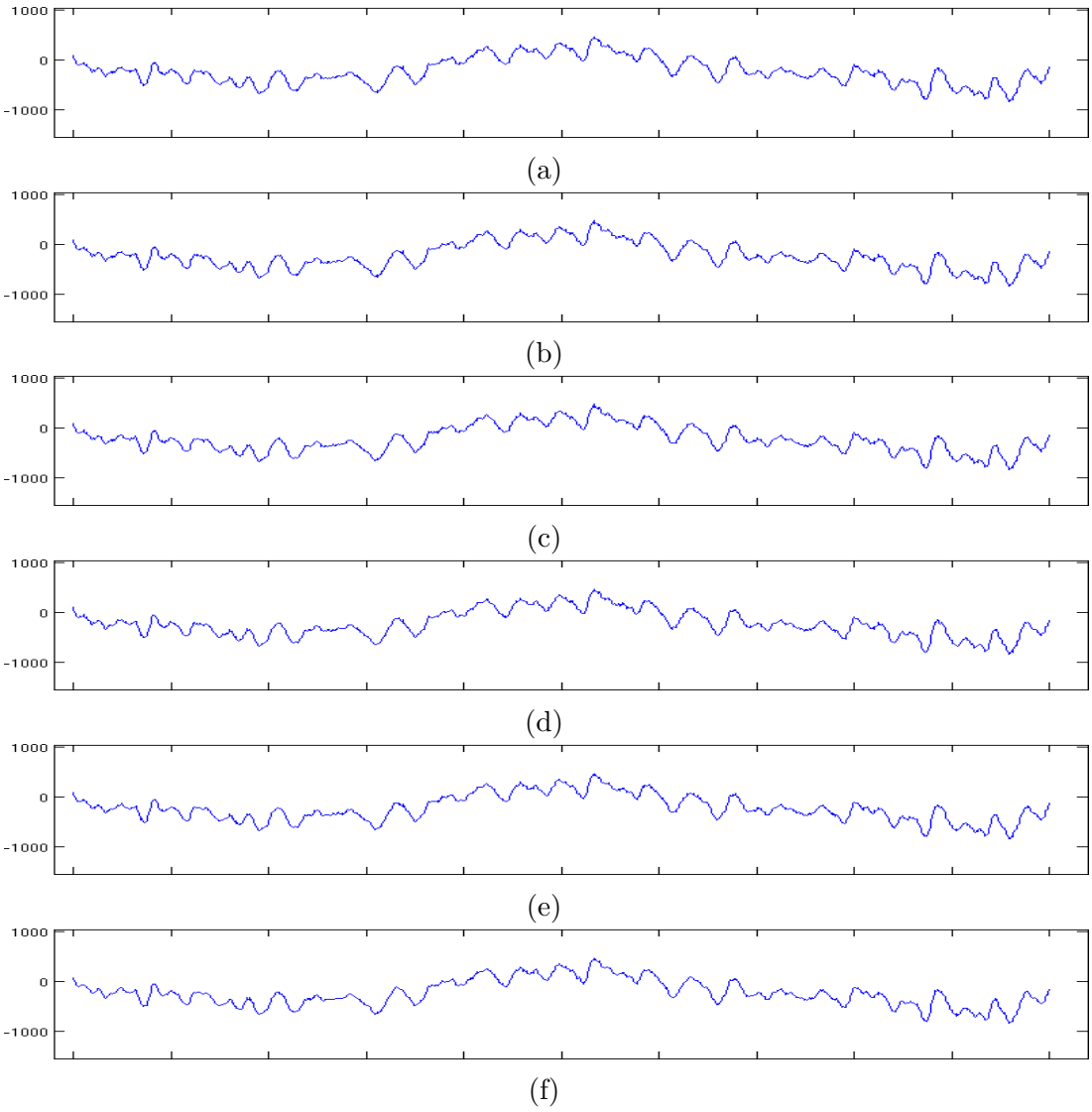


Figura B.3: Seção entre 1h00m00s e 1h00m04s do canal de EEG do registro slp16 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.

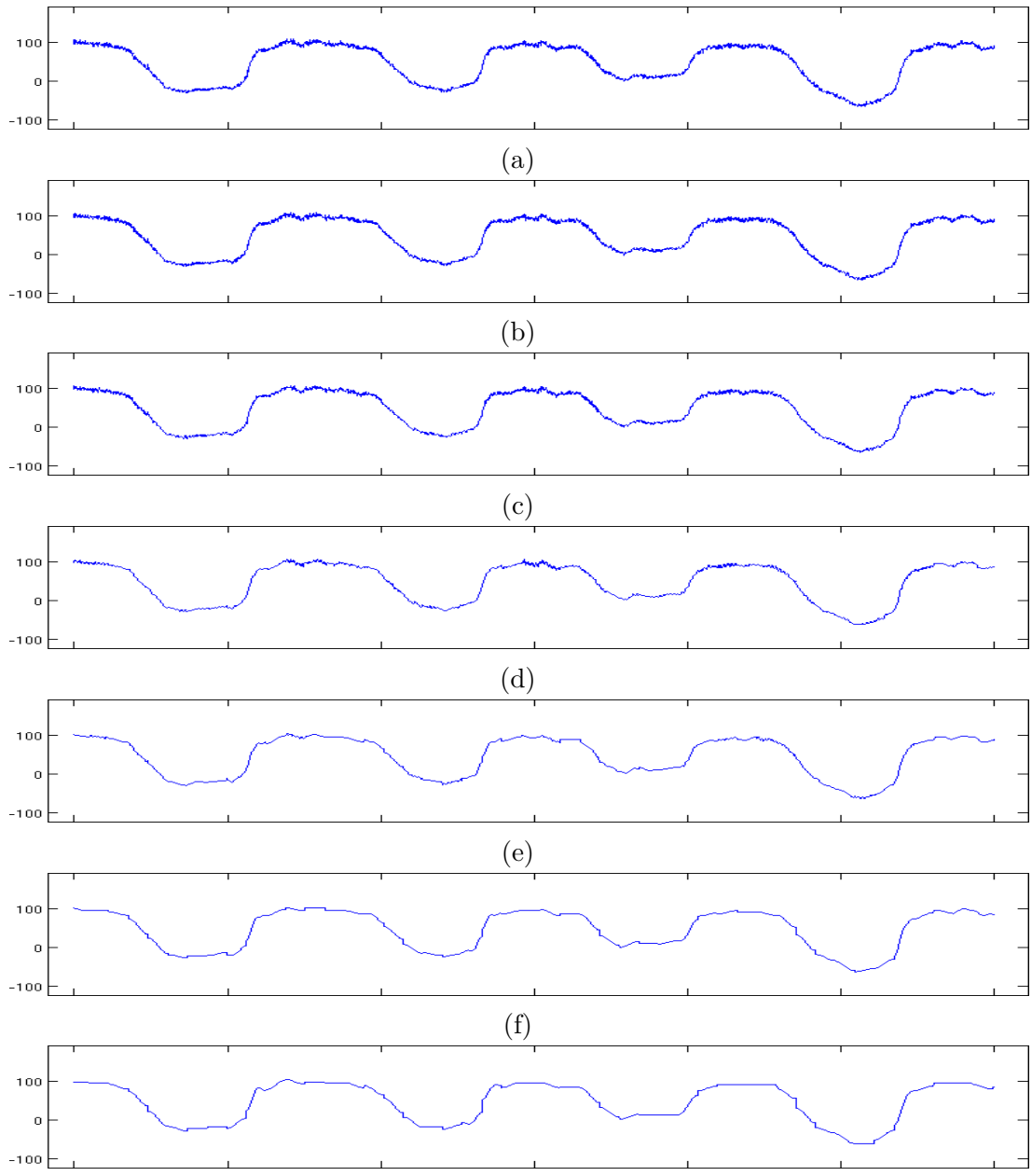


Figura B.4: Seção entre 1h15m00s e 1h15m12s do canal de respiração do registro slp02a do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 1,5%. (d) reconstruído com uma NPRD de 2,0%. (e) reconstruído com uma NPRD de 2,5%. (f) reconstruído com uma NPRD de 3,0%. (g) reconstruído com uma NPRD de 3,5%.

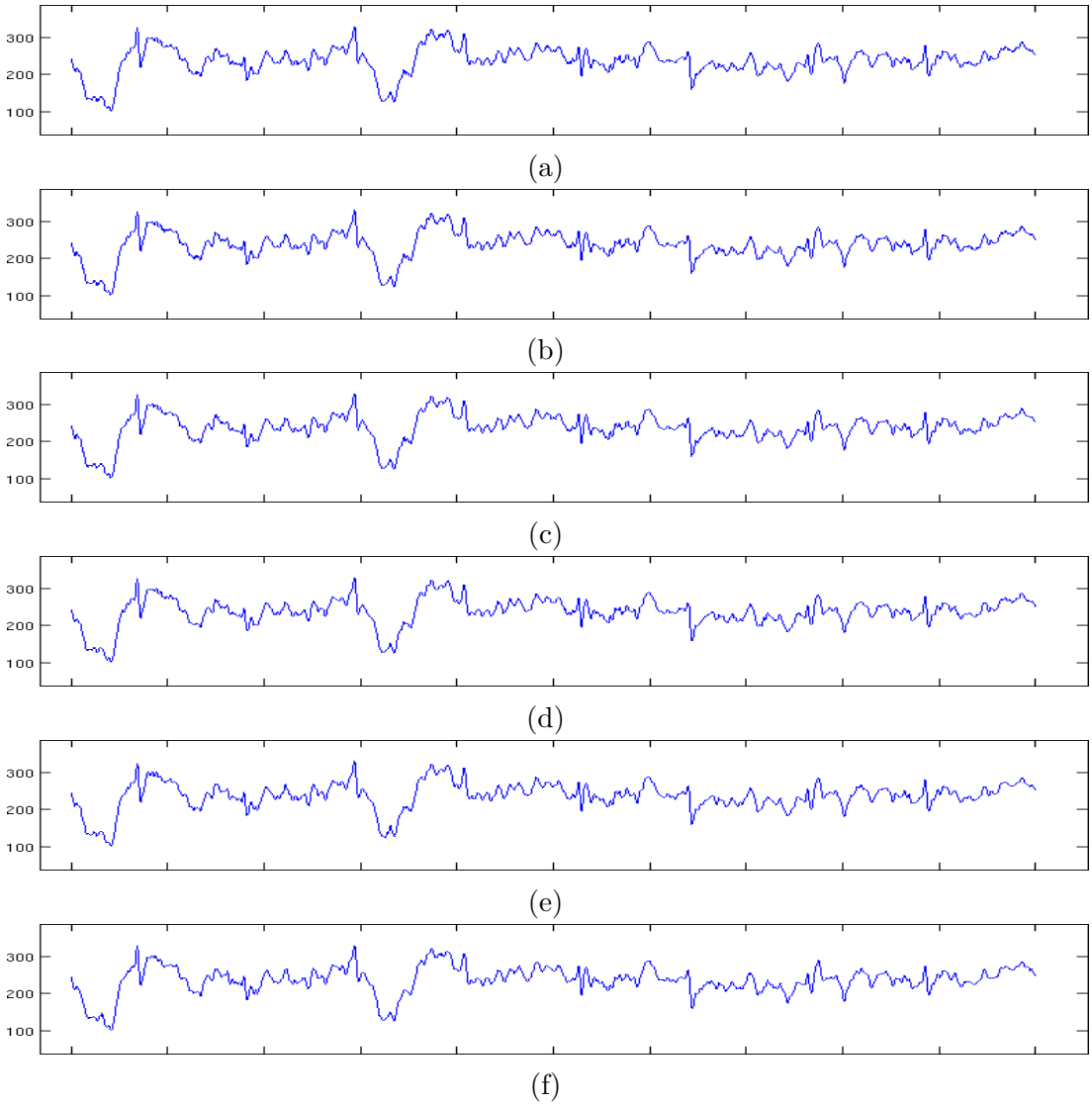


Figura B.5: Seção entre 2h00m00s e 2h00m08s do canal de EOG do registro slp32 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.

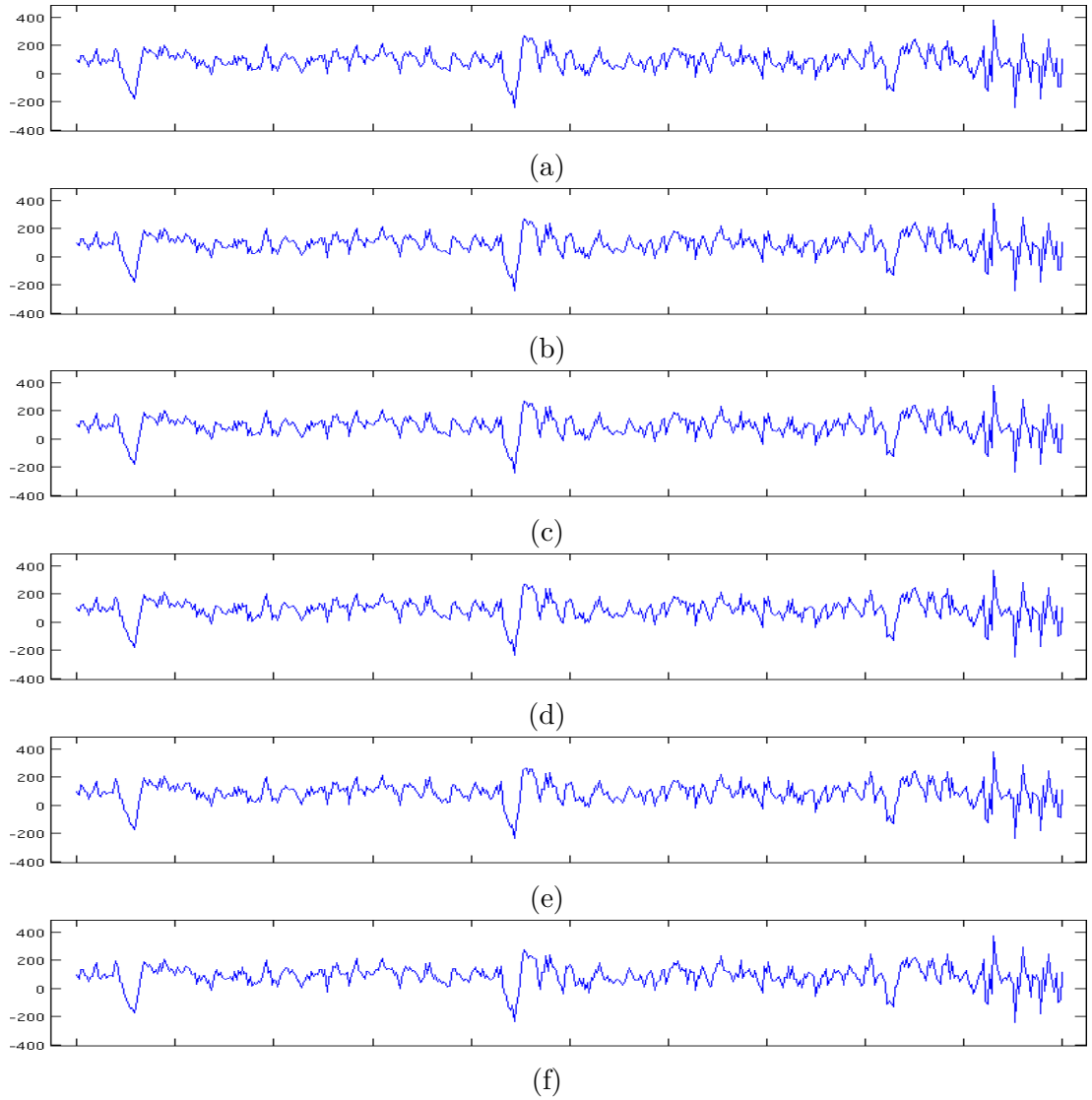


Figura B.6: Seção entre 1h00m00s e 1h00m02s do canal de EMG do registro slp45 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.

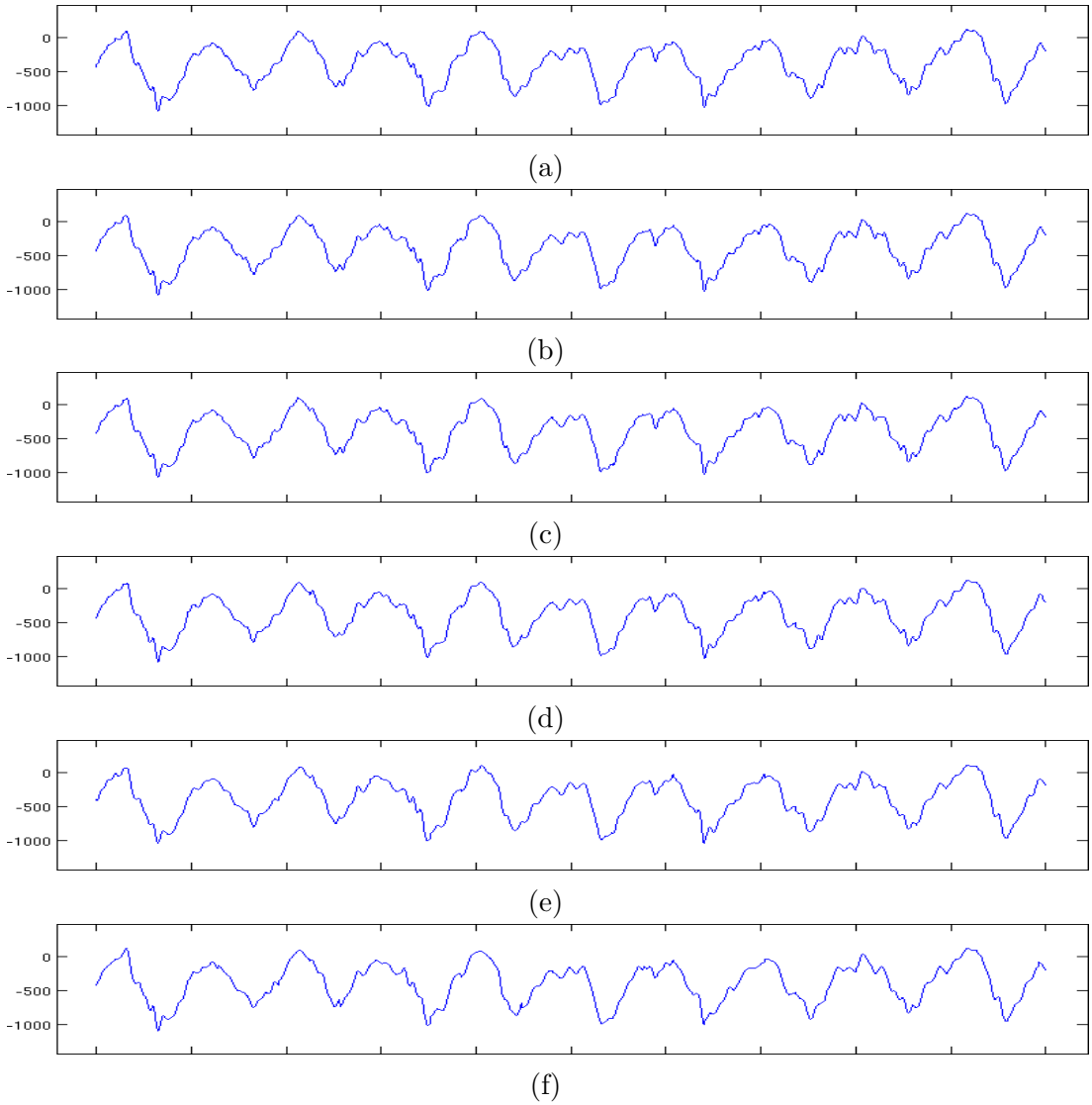
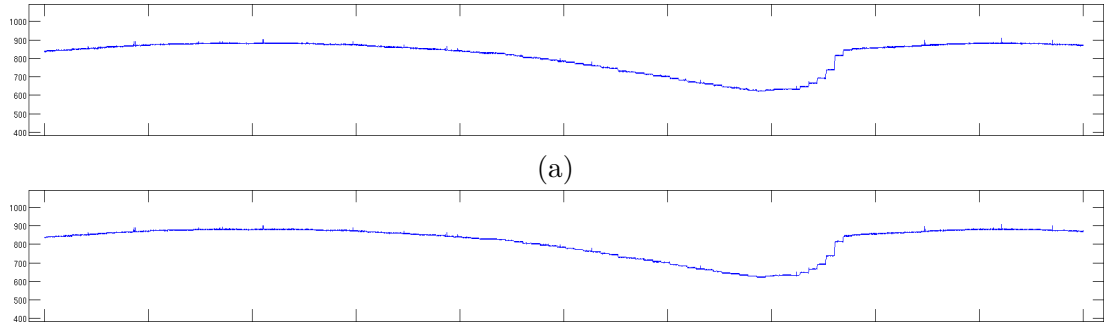


Figura B.7: Seção entre 2h00m00s e 2h00m08s do canal de SV do registro slp60 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 1,0%. (c) reconstruído com uma NPRD de 2,0%. (d) reconstruído com uma NPRD de 3,0%. (e) reconstruído com uma NPRD de 4,0%. (f) reconstruído com uma NPRD de 5,0%.



(a)

(b)

Figura B.8: Seção entre 1h00m00s e 1h01m20s do canal de O_2S do registro slp61 do MIT/BIH PSGDB (a) original. (b) reconstruído com uma NPRD de 0,356%.

Apêndice C

Resultados de Comparação Entre DWT+SPIHT e DCT+MinLag

Este apêndice complementa os resultados da Seção 6.3 com outras bases *wavelet*. Mais especificamente, serão apresentados os resultados para a base ortogonal Daubechies 10 e as bases biortogonais CDF 9/7 e CDF 5/3 nas Tabelas C.1 a C.33.

Tabela C.1: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para ECGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	4,27	3,05	**4,75**	3,02
slp01b	6,17	4,26	**7,52**	4,23
slp02a	3,51	3,37	**3,90**	3,34
slp02b	3,41	3,05	**3,66**	3,04
slp03	5,83	3,88	**7,70**	3,88
slp04	6,64	3,98	**7,49**	3,98
slp14	6,90	3,63	**8,19**	3,61
slp16	4,39	3,79	**5,21**	3,82
slp32	4,27	3,78	**5,28**	3,76
slp37	4,07	4,10	**4,85**	4,13
slp41	5,80	3,78	**6,93**	3,76
slp45	5,83	3,89	**6,82**	3,86
slp48	5,80	4,05	**6,61**	4,07
slp59	2,40	3,43	**4,42**	3,47
slp60	2,66	4,33	**7,11**	4,32
slp61	2,66	4,17	**6,59**	4,13
slp66	9,12	4,61	**12,74**	4,64
slp67x	8,70	4,55	**11,39**	4,59

Tabela C.2: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para BPs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	21,57	3,26	**21,94**	3,24
slp01b	23,39	3,55	**30,56**	3,51
slp02a	21,39	3,43	**25,68**	3,45
slp02b	20,26	3,69	**23,46**	3,74
slp03	20,88	3,44	**24,59**	3,44
slp04	25,22	3,48	**28,95**	3,47
slp14	24,78	4,01	**28,76**	4,03
slp16	24,16	3,80	**25,59**	3,80
slp32	23,88	3,04	**27,04**	3,01
slp37	20,18	3,48	**22,33**	3,46
slp41	28,01	3,76	**30,62**	3,76
slp45	20,27	3,24	**22,37**	3,24
slp48	22,13	2,87	**25,03**	2,86
slp59	26,07	3,92	**28,88**	3,95
slp60	23,45	3,92	**28,35**	3,89
slp61	22,46	3,57	**27,10**	3,59
slp66	21,70	3,79	**25,20**	3,76
slp67x	20,74	3,28	**23,64**	3,31

Tabela C.3: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para EEGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	6,87	4,10	**7,65**	4,10
slp01b	2,64	3,50	**3,16**	3,49
slp02a	3,41	4,20	**4,68**	4,19
slp02b	2,93	3,55	**3,59**	3,55
slp03	4,43	3,54	**5,89**	3,53
slp04	3,68	3,70	**4,44**	3,68
slp14	3,50	3,81	**4,90**	3,83
slp16	3,09	3,45	**4,06**	3,45
slp32	3,04	3,61	**3,88**	3,58
slp37	2,131	3,59	**2,77**	3,59
slp41	3,01	3,63	**4,30**	3,62
slp45	4,28	4,03	**5,72**	4,05
slp48	3,02	3,72	**3,88**	3,71
slp59	3,76	3,79	**5,15**	3,84
slp60	2,62	4,08	**3,85**	4,07
slp61	4,77	4,01	**6,37**	3,99
slp66	3,17	4,17	**4,18**	4,18
slp67x	4,76	3,93	**5,72**	3,94

Tabela C.4: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para sinais de RSP (abdominal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp37	**107,43**	3,97	**80,05**	3,79
slp41	**104,45**	3,95	84,39	3,94
slp45	**148,02**	4,01	112,98	3,99
slp59	24,72	3,64	**103,52**	3,67
slp60	25,90	3,60	**101,69**	3,65
slp61	**202,75**	4,21	**133,54**	3,70
slp66	**126,71**	4,44	89,64	4,44

Tabela C.5: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para sinais de RSP (chest) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	**195,76**	3,47	**109,25**	2,85
slp48	4,75	3,03	**12,61**	2,98
slp67x	**220,45**	4,30	**137,30**	3,71

Tabela C.6: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para sinais de RSP (nasal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp02a	31,45	4,11	**87,31**	4,11
slp02b	32,60	4,24	**86,01**	4,27
slp03	**83,52**	3,96	66,74	4,00
slp04	**129,96**	4,19	110,33	4,16
slp14	14,83	3,87	**40,50**	3,88
slp16	**112,57**	3,99	89,52	4,04
slp32	24,74	4,20	**56,79**	4,23
slp37	10,49	3,89	**14,77**	3,88
slp41	**80,24**	4,32	**60,74**	3,20
slp45	14,26	3,64	**28,81**	3,63
slp48	**171,65**	4,04	**113,98**	3,36
slp59	**189,29**	4,12	**105,31**	3,04
slp60	11,06	3,39	**55,91**	3,43
slp66	**94,58**	4,15	74,50	4,17
slp67x	28,49	4,15	**72,25**	4,19

Tabela C.7: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para sinais de RSP (sum) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	**160,30**	4,05	**114,19**	3,98
slp01b	115,96	4,27	**116,50**	3,68

Tabela C.8: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para EOGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	5,64	3,75	**7,82**	3,79
slp37	7,04	4,11	**10,47**	4,09
slp41	29,14	4,03	**35,60**	4,05
slp45	12,60	3,82	**14,99**	3,82
slp48	12,28	4,28	**22,98**	4,28

Tabela C.9: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para EMGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	1,86	3,79	**2,43**	3,76
slp37	1,95	3,55	**2,44**	3,54
slp41	2,36	3,40	**3,18**	3,42
slp45	2,29	3,79	**3,33**	3,79
slp48	2,04	3,75	**2,88**	3,74

Tabela C.10: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para SVs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	20,77	3,74	**21,62**	3,72
slp60	16,82	3,65	**17,42**	3,17
slp61	16,95	3,01	**19,55**	3,05
slp66	13,78	3,46	**15,66**	3,45
slp67x	12,55	3,77	**13,63**	3,62

Tabela C.11: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* Daubechies 10 para O_2Ss do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	**776,43**	3,46	**267,79**	0,66
slp60	15,27	2,59	**596,96**	1,33
slp61	8,80	3,77	**483,58**	2,18
slp66	13,83	3,91	**264,54**	3,89
slp67x	14,09	4,14	**401,24**	1,76

Tabela C.12: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para ECGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	4,50	3,07	**4,87**	3,11
slp01b	6,74	4,31	**7,74**	4,34
slp02a	3,67	3,43	**3,98**	3,45
slp02b	3,55	3,10	**3,70**	3,10
slp03	6,40	3,92	**7,70**	3,88
slp04	6,94	3,80	**7,16**	3,78
slp14	7,72	3,66	**8,38**	3,71
slp16	4,54	3,85	**5,21**	3,82
slp32	4,49	3,82	**5,36**	3,84
slp37	4,16	4,09	**4,85**	4,13
slp41	6,10	3,72	**6,93**	3,76
slp45	6,36	3,95	**6,99**	3,98
slp48	6,11	4,10	**6,72**	4,15
slp59	2,44	3,47	**4,44**	3,50
slp60	2,66	4,37	**7,11**	4,32
slp61	2,64	4,18	**6,59**	4,13
slp66	10,81	4,58	**12,64**	4,57
slp67x	9,93	4,58	**11,39**	4,59

Tabela C.13: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para BPs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	**23,52**	3,49	23,43	3,50
slp01b	25,06	3,72	**32,16**	3,75
slp02a	21,31	3,49	**25,68**	3,45
slp02b	21,04	4,06	**24,91**	4,09
slp03	21,67	3,48	**24,85**	3,51
slp04	26,99	3,70	**30,56**	3,73
slp14	23,27	3,44	**25,28**	3,42
slp16	24,85	4,00	**26,26**	3,95
slp32	24,76	3,38	**29,34**	3,39
slp37	20,31	3,17	**21,28**	3,21
slp41	28,20	3,85	**31,02**	3,85
slp45	21,68	3,55	**23,72**	3,53
slp48	24,18	3,13	**26,39**	3,12
slp59	27,98	4,28	**30,47**	4,27
slp60	25,15	4,26	**29,85**	4,28
slp61	23,28	3,63	**27,10**	3,59
slp66	22,81	4,17	**27,04**	4,22
slp67x	22,83	3,55	**24,55**	3,54

Tabela C.14: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para EEGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	6,85	4,23	**7,81**	4,18
slp01b	2,66	3,56	**3,20**	3,59
slp02a	3,43	4,28	**4,75**	4,30
slp02b	2,94	3,61	**3,62**	3,63
slp03	4,32	3,55	**5,89**	3,53
slp04	3,69	3,81	**4,58**	3,86
slp14	3,53	3,92	**4,97**	3,91
slp16	3,11	3,52	**4,10**	3,51
slp32	3,05	3,71	**3,96**	3,72
slp37	2,14	3,62	**2,77**	3,59
slp41	2,96	3,61	**4,30**	3,62
slp45	4,31	4,13	**5,80**	4,12
slp48	2,99	3,66	**3,84**	3,63
slp59	3,68	3,41	**4,76**	3,40
slp60	2,58	4,06	**3,85**	4,07
slp61	4,80	4,13	**6,53**	4,11
slp66	3,06	4,09	**4,12**	4,07
slp67x	4,80	4,02	**5,79**	4,02

Tabela C.15: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para sinais de RSP (abdominal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	**205,59**	3,69	**109,25**	2,85
slp48	4,76	3,09	**12,88**	3,06
slp67x	**239,71**	4,47	**137,30**	3,71
slp59	25,06	3,58	**100,53**	3,55
slp60	25,23	3,61	**101,69**	3,65
slp61	**194,21**	4,00	**133,54**	3,70
slp66	**117,34**	4,28	88,38	4,26

Tabela C.16: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para sinais de RSP (chest) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	**205,59**	3,69	**109,25**	2,85
slp48	4,76	3,09	**12,88**	3,06
slp67x	**239,71**	4,47	**137,30**	3,71

Tabela C.17: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para sinais de RSP (nasal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp02a	32,09	4,32	**90,49**	4,29
slp02b	33,19	4,44	**88,86**	4,42
slp03	**88,15**	3,92	65,79	3,90
slp04	**137,35**	4,22	111,87	4,25
slp14	14,87	3,96	**41,78**	3,96
slp16	**120,43**	4,14	91,42	4,18
slp32	24,91	4,05	**53,85**	4,05
slp37	10,58	3,93	**14,77**	3,88
slp41	**85,95**	4,40	**60,74**	3,20
slp45	14,38	3,51	**28,08**	3,54
slp48	**181,51**	4,26	**113,98**	3,36
slp59	**182,62**	3,79	**105,31**	3,04
slp60	11,83	3,52	**56,42**	3,53
slp66	**100,09**	4,34	76,03	4,30
slp67x	28,83	4,24	**72,25**	4,19

Tabela C.18: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para sinais de RSP (sum) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	**166,26**	4,28	**114,19**	3,98
slp01b	80,65	4,38	**116,50**	3,68

Tabela C.19: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para EOGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	5,43	3,64	**7,59**	3,66
slp37	6,81	3,82	**9,83**	3,79
slp41	30,95	4,19	**36,79**	4,15
slp45	12,82	3,87	**15,18**	3,88
slp48	11,32	3,47	**19,16**	3,49

Tabela C.20: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para EMGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	1,85	3,80	**2,43**	3,76
slp37	1,95	3,58	**2,44**	3,54
slp41	2,37	3,43	**3,18**	3,42
slp45	2,30	3,82	**3,33**	3,79
slp48	2,04	3,76	**2,88**	3,74

Tabela C.21: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para SVs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	21,68	4,06	**22,77**	4,013
slp60	16,47	3,74	**17,42**	3,177
slp61	16,16	3,03	**19,55**	3,056
slp66	12,92	3,13	**15,36**	3,178
slp67x	12,95	4,01	**13,63**	3,626

Tabela C.22: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 9/7 para O_2Ss do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	**868,05**	3,44	**267,79**	0,66
slp60	15,39	4,23	**596,96**	1,33
slp61	8,91	3,66	**483,58**	2,18
slp66	14,09	3,89	**264,54**	3,89
slp67x	14,04	3,70	**401,24**	1,76

Tabela C.23: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para ECGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	4,58	3,25	**5,16**	3,29
slp01b	6,49	4,25	**7,48**	4,21
slp02a	3,64	3,47	**3,98**	3,45
slp02b	3,53	3,14	**3,70**	3,10
slp03	6,33	3,98	**7,89**	3,99
slp04	6,75	4,02	**7,49**	3,98
slp14	7,29	3,65	**8,19**	3,61
slp16	4,45	3,89	**5,31**	3,92
slp32	4,46	3,87	**5,36**	3,84
slp37	4,10	4,14	**4,85**	4,13
slp41	5,83	3,71	**6,87**	3,73
slp45	6,09	3,94	**6,99**	3,98
slp48	5,85	4,16	**6,72**	4,15
slp59	2,39	3,68	**4,52**	3,66
slp60	2,68	4,26	**7,09**	4,30
slp61	2,86	4,53	**6,99**	4,50
slp66	10,2	4,53	**12,51**	4,50
slp67x	9,23	4,46	**11,17**	4,50

Tabela C.24: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para BPs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	20,64	3,32	**22,40**	3,33
slp01b	22,26	3,45	**29,87**	3,41
slp02a	19,44	3,49	**25,68**	3,45
slp02b	18,80	3,88	**24,19**	3,93
slp03	20,40	3,90	**26,52**	3,90
slp04	24,33	3,45	**28,95**	3,47
slp14	20,88	3,47	**25,67**	3,49
slp16	23,36	3,90	**25,84**	3,86
slp32	22,41	3,15	**27,77**	3,14
slp37	19,35	3,70	**23,23**	3,68
slp41	26,35	3,76	**30,62**	3,76
slp45	19,24	3,43	**23,25**	3,42
slp48	23,90	3,64	**28,26**	3,61
slp59	26,26	3,98	**28,88**	3,95
slp60	23,20	4,17	**29,44**	4,15
slp61	20,92	3,52	**26,69**	3,49
slp66	20,78	4,04	**26,49**	4,07
slp67x	20,58	3,28	**23,64**	3,31

Tabela C.25: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para EEGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	6,34	4,19	**7,81**	4,18
slp01b	2,67	3,70	**3,24**	3,69
slp02a	3,43	4,33	**4,75**	4,30
slp02b	2,96	3,72	**3,65**	3,70
slp03	4,25	3,58	**6,00**	3,61
slp04	3,62	3,93	**4,65**	3,96
slp14	3,30	3,76	**4,83**	3,75
slp16	3,10	3,60	**4,14**	3,58
slp32	3,02	3,80	**4,00**	3,79
slp37	2,07	3,18	**2,65**	3,18
slp41	2,97	3,73	**4,39**	3,75
slp45	4,14	3,94	**5,57**	3,91
slp48	2,93	3,51	**3,78**	3,53
slp59	3,60	3,50	**4,83**	3,48
slp60	2,52	3,85	**3,75**	3,86
slp61	4,73	4,14	**6,53**	4,11
slp66	2,94	4,01	**4,08**	4,00
slp67x	4,72	4,10	**5,86**	4,10

Tabela C.26: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para sinais de RSP (abdominal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp37	**104,45**	3,91	**80,05**	3,79
slp41	**96,59**	4,06	84,97	4,08
slp45	**150,01**	4,15	**112,98**	3,99
slp59	24,86	3,54	**99,91**	3,52
slp60	25,00	3,55	**97,48**	3,53
slp61	**182,84**	4,11	**133,54**	3,70
slp66	**120,15**	4,52	**89,64**	4,44

Tabela C.27: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para sinais de RSP (chest) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	**193,16**	4,15	**109,25**	2,85
slp48	4,63	2,93	**12,26**	2,89
slp67x	**207,14**	4,11	**137,30**	3,71

Tabela C.28: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para sinais de RSP (nasal) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp02a	30,05	4,09	**87,31**	4,11
slp02b	31,17	4,23	**86,01**	4,27
slp03	**85,12**	4,24	69,96	4,28
slp04	**127,28**	3,93	105,96	3,94
slp14	14,75	3,82	**39,27**	3,79
slp16	112,21	4,11	**90,81**	4,14
slp32	24,47	4,12	**54,89**	4,12
slp37	10,50	3,89	**14,77**	3,88
slp41	**76,72**	4,02	**60,74**	3,20
slp45	14,28	3,64	**28,81**	3,63
slp48	**159,01**	3,96	**113,98**	3,36
slp59	**168,00**	4,13	**105,31**	3,04
slp60	10,53	3,49	**56,42**	3,53
slp66	**86,55**	4,02	72,73	4,07
slp67x	28,62	4,05	**69,58**	4,07

Tabela C.29: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para sinais de RSP (sum) do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp01a	**141,77**	4,08	**114,19**	3,98
slp01b	116,17	4,20	**116,50**	3,68

Tabela C.30: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para EOGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	5,13	3,72	**7,64**	3,69
slp37	6,52	4,17	**10,63**	4,16
slp41	26,64	3,89	**33,97**	3,88
slp45	11,77	3,92	**15,32**	3,93
slp48	11,04	4,23	**22,67**	4,22

Tabela C.31: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para EMGs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp32	1,87	3,91	**2,47**	3,95
slp37	1,97	3,70	**2,47**	3,68
slp41	2,35	3,46	**3,18**	3,42
slp45	2,31	3,92	**3,36**	3,88
slp48	2,06	3,85	**2,90**	3,84

Tabela C.32: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para SVs do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	19,14	3,83	**22,13**	3,84
slp60	15,38	3,76	**17,42**	3,17
slp61	15,56	3,10	**19,55**	3,05
slp66	13,19	3,70	**15,75**	3,61
slp67x	11,47	3,85	**13,63**	3,62

Tabela C.33: Comparação de valores de RC e NPRD dos compressores DWT+SPIHT e DCT+MinLag usando a base *wavelet* CDF 5/3 para O_2Ss do MIT/BIH PSGDB.

Sinal	**DWT+SPIHT**		**DCT+MinLag**	
	RC	**NPRD**	**RC**	**NPRD**
slp59	**972,96**	2,78	**267,79**	0,66
slp60	16,19	3,42	**596,96**	1,33
slp61	9,07	3,25	**483,58**	2,18
slp66	14,32	3,75	**239,54**	3,72
slp67x	23,76	3,08	**401,24**	1,76

Anexo D

Coeficientes de Filtros *Wavelet*

As Tabelas D.1, D.2, D.3, D.4 e D.5 apresentam os coeficientes de algumas das bases ortogonais mais utilizadas para o cálculo de *wavelets* na literatura.

Tabela D.1: Coeficientes do filtro Beylkin. Fonte: Salomon (2006).

0,099305765374	0,424215360813	0,699825214057	0,449718251149
-0,110927598348	-0,264497231446	0,026900308804	0,155538731877
-0,017520746267	-0,088543630623	0,019679866044	0,042916387274
-0,017460408696	-0,014365807969	0,010040411845	0,001484234782
-0,002736031626	0,000640485329		
Beylkin (18-*taps*)			

Tabela D.2: Coeficientes do filtro Vaidyanathan. Fonte: Salomon (2006).

-0,000062906118	0,000343631905	-0,000453956620	-0,000944897136
0,002843834547	0,000708137504	-0,008839103409	0,003153847056
0,019687215010	-0,014853448005	-0,035470398607	0,038742619293
0,055892523691	-0,077709750902	-0,083928884366	0,131971661417
0,135084227129	-0,194450471766	-0,263494802488	0,201612161775
-0,083928884366	0,572797793211	0,250184129505	0,045799334111
Vaidyanathan (24-*taps*)			

Tabela D.3: Coeficientes dos filtros da família Coifman com 6, 12, 18, 24 e 30 *taps*. Fonte: Salomon (2006).

0,038580777748	-0,126969125396	-0,077161555496	0,607491641386
0,745687558934	0,226584265197		
Coifman 1 (6-*taps*)			
0,016387336463	-0,041464936782	-0,067372554722	0,386110066823
0,812723635450	0,417005184424	-0,076488599078	-0,059434418646
0,023680171947	0,005611434819	-0,001823208871	-0,000720549445
Coifman 2 (12-*taps*)			
-0,003793512864	0,007782596426	0,023452696142	-0,065771911281
-0,061123390003	0,405176902410	0,793777222626	0,428483476378
-0,071799821619	-0,082301927106	0,034555027573	0,015880544864
-0,009007976137	-0,002574517688	0,001117518771	0,000466216960
-0,000070983303	-0,000034599773		
Coifman 3 (18-*taps*)			
0,000892313668	-0,001629492013	-0,007346166328	0,016068943964
0,026682300156	-0,081266699680	-0,056077313316	0,415308407030
0,782238930920	0,434386056491	-0,066627474263	-0,096220442034
0,039334427123	0,025082261845	-0,015211731527	-0,005658286686
0,003751436157	0,001266561929	-0,000589020757	-0,000259974552
0,000062339034	0,000031229876	-0,000003259680	-0,000001784985
Coifman 4 (24-*taps*)			
-0,000212080863	0,000358589677	0,002178236305	-0,004159358782
-0,010131117538	0,023408156762	0,028168029062	-0,091920010549
-0,052043163216	0,421566206729	0,774289603740	0,437991626228
-0,062035963906	-0,105574208706	0,041289208741	0,032683574283
-0,019761779012	-0,009164231153	0,006764185419	0,002433373209
-0,001662863769	-0,000638131296	0,000302259520	0,000140541149
-0,000041340484	-0,000021315014	0,000003734597	0,000002063806
-0,000000167408	-0,000000095158		
Coifman 5 (30-*taps*)			

Tabela D.4: Coeficientes dos filtros da família Daubechies com 4, 6, 8, 10, 12, 14, 16, 18 e 20 *taps*. Fonte: Daubechies (1988) e Salomon (2006).

0,482962913145	0,836516303738	0,224143868042	-0,129409522551
Daubechies 2 (4-*taps*)			
0,332670552950	0,806891509311	0,459877502118	-0,135011020010
-0,085441273882	0,035226291882		
Daubechies 3 (6-*taps*)			
0,230377813309	0,714846570553	0,630880767930	-0,027983769417
-0,187034811719	0,030841381836	0,032883011667	-0,010597401785
Daubechies 4 (8-*taps*)			
0,160102397974	0,603829269797	0,724308528438	0,138428145901
-0,242294887066	-0,032244869585	0,077571493840	-0,006241490213
-0,012580751999	0,003335725285		
Daubechies 5 (10-*taps*)			
0,111540743350	0,494623890398	0,751133908021	0,315250351709
-0,226264693965	-0,129766867567	0,097501605587	0,027522865530
-0,031582039317	0,000553842201	0,004777257511	-0,001077301085
Daubechies 6 (12-*taps*)			
0,077852054085	0,396539319482	0,729132090846	0,469782287405
-0,143906003929	-0,224036184994	0,071309219267	0,080612609151
-0,038029936935	-0,016574541631	0,012550998556	0,000429577973
-0,001801640704	0,000353713800		
Daubechies 7 (14-*taps*)			
0,054415842243	0,312871590914	0,675630736297	0,585354683654
-0,015829105256	-0,284015542962	0,000472484574	0,128747426620
-0,017369301002	-0,044088253931	0,013981027917	0,008746094047
-0,004870352993	-0,000391740373	0,000675449406	-0,000117476784
Daubechies 8 (16-*taps*			
0,038077947364	0,243834674613	0,604823123690	0,657288078051
0,133197385825	-0,293273783279	-0,096840783223	0,148540749338
0,030725681479	-0,067632829061	0,000250947115	0,022361662124
-0,004723204758	-0,004281503682	0,001847646883	0,000230385764
-0,000251963189	0,000039347320		
Daubechies 9 (18-*taps*)			
0,026670057901	0,188176800078	0,527201188932	0,688459039454
0,281172343661	-0,249846424327	-0,195946274377	0,127369340336
0,093057364604	-0,071394147166	-0,029457536822	0,033212674059
0,003606553567	-0,010733175483	0,001395351747	0,001992405295
-0,000685856695	-0,000116466855	0,000093588670	-0,000013264203
Daubechies 10 (20-*taps*)			

Tabela D.5: Coeficientes dos filtros da família *Symmlet* com 8, 10, 12, 14, 16, 18 e 20 *taps*. Fonte: Salomon (2006).

-0,107148901418	-0,041910965125	0,703739068656	1,136658243408
0,421234534204	-0,140317624179	-0,017824701442	0,045570345896
Symmlet 4 (8-*taps*)			
0,038654795955	0,041746864422	-0,055344186117	0,281990696854
1,023052966894	0,896581648380	0,023478923136	-0,247951362613
-0,029842499869	0,027632152958		
Symmlet 5 (10-*taps*)			
0,038654795955	0,004936612372	-0,166863215412	-0,068323121587
0,694457972958	1,113892783926	0,477904371333	-0,102724969862
-0,029783751299	0,063250562660	0,002499922093	-0,011031867509
Symmlet 6 (12-*taps*)			
0,003792658534	-0,001481225915	-0,017870431651	0,043155452582
0,096014767936	-0,070078291222	0,024665659489	0,758162601964
1,085782709814	0,408183939725	-0,198056706807	-0,152463871896
0,005671342686	0,014521394762		
Symmlet 7 (14-*taps*)			
0,002672793393	-0,000428394300	-0,021145686528	0,005386388754
0,069490465911	-0,038493521263	-0,073462508761	0,515398670374
1,099106630537	0,680745347190	-0,086653615406	-0,202648655286
0,010758611751	0,044823623042	-0,000766690896	-0,004783458512
Symmlet 8 (16-*taps*)			
0,001512487309	-0,000669141509	-0,014515578553	0,012528896242
0,087791251554	-0,025786445930	-0,270893783503	0,049882830959
0,873048407349	1,015259790832	0,337658923602	-0,077172161097
0,000825140929	0,042744433602	-0,016303351226	-0,018769396836
0,000876502539	0,001981193736		
Symmlet 9 (18-*taps*)			
0,001089170447	0,000135245020	-0,012220642630	-0,002072363923
0,064950924579	0,016418869426	-0,225558972234	-0,100240215031
0,667071338154	1,088251530500	0,542813011213	-0,050256540092
-0,045240772218	0,070703567550	0,008152816799	-0,028786231926
-0,001137535314	0,006495728375	0,000080661204	-0,000649589896
Symmlet 10 (20-*taps*)			