UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Tiarles da Rocha Moralles Guterres

ANÁLISE DO GRÁFICO DE CONTROLE CUSUM PARA A FILTRAGEM DE SINAIS UTILIZANDO COEFICIENTES WAVELET

Santa Maria, RS
2018

Tiarles da Rocha Moralles Guterres

ANÁLISE DO GRÁFICO DE CONTROLE CUSUM PARA A FILTRAGEM DE SINAIS UTILIZANDO COEFICIENTES WAVELET

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Engenheiro de Computação**.

ORIENTADORA: Prof.ª Alice de Jesus Kozakevicius

COORIENTADOR: Prof. Fábio Mariano Bayer

Santa Maria, RS
2018

RESUMO

ANÁLISE DO GRÁFICO DE CONTROLE CUSUM PARA A FILTRAGEM DE SINAIS UTILIZANDO COEFICIENTES WAVELET

AUTOR: Tiarles da Rocha Moralles Guterres
ORIENTADORA: Alice de Jesus Kozakevicius
COORIENTADOR: Fábio Mariano Bayer

A transformada de Fourier foi por muito tempo uma das poucas opções para processamento de sinais disponível. No entanto, nas últimas décadas diferentes tipos de transformadas vêm se tornando boas alternativas para filtragem, compactação e demais tipos de análises de sinais. Neste trabalho, a transformada wavelet de Daubechies será uma das ferramentas para a filtragem de sinais com diferentes níveis de ruído. Em um sinal corrompido por ruído aditivo, as componentes de alta frequência são as mais comprometidas, e necessitam de um tratamento mais rigoroso para uma reconstrução do sinal filtrado mais próximo ao sinal ideal. O presente trabalho investigou a viabilidade de se utilizar gráficos de controle de soma acumulada (CUSUM) como técnica para determinação dos limiares de corte para o processo de truncamento dos coeficientes wavelets, resultantes da transformada wavelet. A metodologia proposta foi validada por meio de simulações numéricas para diferentes sinais e diferentes níveis de ruído associados, chamados de cenários. Ao fim foi encontrada uma configuração que se adapta a maioria destes cenários e demonstrada neste trabalho a competitividade do gráfico CUSUM adaptado para a filtragem de sinais via wavelet ao compará-lo com algoritmos clássicos de filtragem estatística de sinais.

Palavras-chave: Transformada de Wavelet, Gráfico de Controle, CUSUM, Filtragem, Truncamento

ABSTRACT

WAVELET BASED CUSUM CONTROL CHART ANALYSIS FOR SIGNAL FILTERING

AUTHOR: Tiarles da Rocha Moralles Guterres
ADVISOR: Alice de Jesus Kozakevicius
CO-ADVISOR: Fábio Mariano Bayer

The Fourier transform has long been one of the few available signal processing options. However, in the last decades different types of transform have become efficient alternatives for filtering, compression and other types of signal analysis. In this work, the Daubechies wavelet transform will be one of the tools for filtering signals with different noise levels. In a signal corrupted by additive noise, the high frequency components are the most compromised, and they require a more rigorous treatment for a more faithful reconstruction of the ideal signal. The present work investigates the feasibility of using cumulative sum control chart (CUSUM) as a technique to determine the cut-off thresholds for the truncation process of the wavelet coefficients, resulting from the wavelet transform. The proposed methodology will be validated through numerical simulations for different signals and different associated noise levels, called scenarios. In the end, was finded a configuration that adapts in the major of this scenarios, showing the competivity of CUSUM chart for signal filtering, using wavelet coefficients, to analise and compare your efficiency with classic algorithms for signal statistical filtering.

Keywords: Wavelet Transform, Control Chart, CUSUM, Filtration, Truncation

LISTA DE FIGURAS

Figura 2.1 – Wavelets de Daubechies com oito momentos nulos (db8) 12
Figura 2.2 – Sinal *Doppler* ideal e sua decomposição em cinco níveis 14
Figura 2.3 – Sinal *Doppler* ruidoso e sua decomposição em cinco níveis 14
Figura 2.4 – Sinais de teste ideais e com ruído ($\sigma^2_{ruido} = 0.01$) 16
Figura 2.5 – Dois exemplos de análise pelo CUSUM 19
Figura 2.6 – Imagens utilizadas para extração dos coeficientes wavelet e para o teste do decaimento do desvio-padrão 21
Figura 2.7 – Gráfico com a média dos desvios-padrão das imagens de teste, utilizando transformada wavelet Daubechies com oito momentos nulos 22
Figura 2.8 – Gráfico com a média dos desvios-padrão dos sinais de teste em cada nível j da transformada wavelet. $\sigma^2_{ruido} = 0.01$ 22
Figura 3.1 – Fluxograma básico da metodologia aplicada em cada cenário 26
Figura 3.2 – Gráficos com todos os mapas de calor dos cenários acumulados, para a na vertical e para b na horizontal, valores médios e normalizados 27
Figura 3.3 – Gráfico que mostra os valores de h_j para os cinco níveis da transformada wavelet, utilizando a Equação (2.27) e os parâmetros encontrados na Figura 3.2 ($a = -7.0$ e $b = 0.201$) 27
Figura 3.4 – Gráfico com a filtragem do sinal de teste *Doppler* com uma $\sigma^2_{ruido} = 0.005$ 29
Figura A.1 – Sinal: *Block* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0} 34
Figura A.2 – Sinal: *Bump* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0} 35
Figura A.3 – Sinal: *Doppler* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0} 36
Figura A.4 – Sinal: *Heavsine* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0} 37
Figura C.1 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Block** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados 40
Figura C.2 – Mapa de calor médio de todos os mapas de calor da Figura C.1 40
Figura C.3 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Bump** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados 41
Figura C.4 – Mapa de calor médio de todos os mapas de calor da Figura C.3 41
Figura C.5 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Doppler** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados 42
Figura C.6 – Mapa de calor médio de todos os mapas de calor da Figura C.5 42
Figura C.7 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Heavsine** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados 43
Figura C.8 – Mapa de calor médio de todos os mapas de calor da Figura C.7 43
Figura A.1 – Gráficos que relacionam o nível wavelet dos coeficientes e o desvio-padrão médio no nível 47

LISTA DE TABELAS

Tabela 3.1 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Block** 30
Tabela 3.2 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Bump** 30
Tabela 3.3 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Doppler** 30
Tabela 3.4 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Heavsine** 30
Tabela A.1 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Block** ($\sigma^2_{ideal} = 0.0748$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0836$) 34
Tabela A.2 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Bump** ($\sigma^2_{ideal} = 0.0191$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0278$) ... 35
Tabela A.3 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Doppler** ($\sigma^2_{ideal} = 0.0834$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0970$) . 36
Tabela A.4 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Heavsine** ($\sigma^2_{ideal} = 0.0882$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0989$) 37
Tabela A.5 – Variância dos sinais de teste ideais 37

LISTA DE SÍMBOLOS

μ	Média
σ	Desvio-padrão
σ^2	Variância
σ^2_{ideal}	Variância de um sinal ideal
σ^2_{ruido}	Variância do ruído aplicado ao sinal ideal
$\sigma^2_{ideal+ruido}$	Variância do sinal ruidoso
$\sigma^2_{filtrado}$	Variância do sinal ruidoso filtrado
$\sigma^2_{residuo}$	Variância do que resta de ruído no sinal filtrado
$d_{j,i}$	Coeficiente wavelet
d_j	Vetor de coeficientes wavelet
$\hat{\sigma}$	Estimador robusto
$\#$	Cardinalidade
λ	Limiar para o threshold dos coeficientes wavelet na filtragem
S^+	Limite de controle superior
S^-	Limite de controle inferior
H^+	Intervalo de Decisão Superior
H^-	Intervalo de Decisão Inferior

SUMÁRIO

1 INTRODUÇÃO 9
2 METODOLOGIA 11
2.1 TRANSFORMADA WAVELET 12
2.1.1 Representação Multirresolução 13
2.1.2 Filtragem via wavelet 15
2.1.3 Determinação do limiar de corte 16
2.2 GRÁFICO DE CONTROLE DE SOMA ACUMULADA (CUSUM) 18
2.2.1 CUSUM e os coeficientes wavelet 20
2.2.2 Truncamento via CUSUM 20
2.2.2.1 Adaptação do gráfico de controle CUSUM para a filtragem via wavelet 21
2.3 FIGURA DE MÉRITO: RAZÃO SINAL-RUÍDO 23
2.4 MÉTODO DE MONTE CARLO PARA VARIAÇÃO DOS PARÂMETROS 24
3 RESULTADOS OBTIDOS 26
3.1 PARÂMETROS ESCOLHIDOS PARA O GRÁFICO CUSUM 26
3.2 COMPARAÇÃO DO CUSUM ADAPTADO COM OUTROS MÉTODOS EM TESTE UNITÁRIO 28
3.3 COMPARAÇÃO DO CUSUM ADAPTADO COM OUTROS MÉTODOS EM TESTES ITERATIVOS 29
4 CONSIDERAÇÕES FINAIS 32
APÊNDICE A – COEFICIENTES WAVELET E ESCALA DOS SINAIS DE TESTE 34
APÊNDICE B – ALGORITMOS QUE MODELAM OS SINAIS DE TESTE 38
APÊNDICE C – MAPAS DE CALOR PARA CADA CENÁRIO TESTADO 40
APÊNDICE D – ALGORITMOS DOS MÉTODOS DE TRUNCAMENTO CONSAGRADOS 44
ANEXO A – GRÁFICOS ORIGINAIS DE CHEN PARA A EVOLUÇÃO DOS DESVIOS-PADRÃO DOS VETORES DE COEFICIENTES WAVELET EM CADA NÍVEL 47
REFERÊNCIAS BIBLIOGRÁFICAS 48

1 INTRODUÇÃO

Para se extrair informações de um sinal qualquer, muitas vezes comprometido por interferências, é necessário construir um sistema para pré-processamento do sinal, este sistema pode ser chamado de filtragem, estimação paramétrica, etc. A filtragem de sinais é uma das formas de processar sinais e preocupa-se ainda com a representação, transformação, manipulação dos sinais e da informação contida neles. (OPPENHEIM; SCHAFER, 2009)

A engenharia elétrica emprega métodos de filtragem utilizando filtros ativos com circuitos amplificadores (RAZAVI, 2008) e transformada de Fourier (OPPENHEIM; SCHAFER, 2009) há muito tempo. As técnicas mais modernas de filtragem de sinais digitais também empregam o uso da transformada de Fourier para resposta em espectro de frequência (BENDAT; PIERSOL, 1993), sinais elétricos simulados (ALEXANDER; SADIKU, 2009), imagens (ESCOFET; MILLAN; RALLO, 2001) e sons (DESAINTE-CATHERINE; MARCHAND, 2000).

A transformada wavelet (NIELSEN, 1998) tem sido usada nas últimas duas décadas e é comercialmente aplicada para compressão de arquivos de imagem JPEG-2000 (MOCCAGATTA; COBAN; CHEN, 1999). Ela tem presença acadêmica em textos relacionados à filtragem de sinais unidimensionais (DONOHO; JOHNSTONE, 1994; DONOHO; JOHNSTONE, 1995) e imagens (CHANG; YU; VETTERLI, 2000), reconhecimento de borda em imagens (CHAGANTI, 2005), reconhecimento de face via codificação de imagem por fractal (TANG; QU, 2010), análise de eletrocardiogramas (KOZAKEVICIUS et al., 2005) e etc.

Donoho e Johnstone foram pioneiros ao apresentarem a transformada de wavelet como forma de filtragem de sinais no ponto de vista estatístico. Em seus dois trabalhos (DONOHO; JOHNSTONE, 1994; DONOHO; JOHNSTONE, 1995) eles introduziram métodos de truncamento (Seções 2.1.2 e 2.1.3) a partir dos coeficientes wavelet de um sinal baseados em minimizar o erro quadrático médio destes coeficientes, obtidos por meio da decomposição em vários níveis. A análise multirresolução (MALLAT, 1989) descreve essa estratégia de decomposição em vários níveis que é a chave principal para investigação de técnicas de truncamento (Seção 2.1.1).

Com a abordagem estatística do truncamento dos coeficientes wavelet proposto por Donoho e Johnstone outras técnicas semelhantes foram desenvolvidas como, por exemplo, o trabalho de Chang, Yu e Vetterli (2000), em que um método inspirado em aspectos Bayesianos foi proposto. Uma década depois do trabalho de Chang, a abordagem de truncamento dos coeficientes wavelet a partir de gráficos de controle foi proposta em Bayer e Kozakevicius (2010) e amplamente comparada em Kozakevicius e Bayer (2014). Nesta estratégia de filtragem, o gráfico de controle de Shewhart (1931) foi adaptado para o truncamento dos coeficientes wavelet, chamado *SPC-Threshold*.

Em tese, gráficos de controle (MONTGOMERY, 2009) são utilizados para monitorar o que na estatística se chama de processo. Um processo é qualquer dado obtido a partir de um sistema sensorial de algum equipamento, como preenchimento de garrafas de refrigerante ou depósito de tinta em uma peça, por exemplo. Estes gráficos servem para avaliar se: um processo está operando de forma correta, com alguns desvios aleatórios aceitáveis; ou se este processo está fora de controle, causado por outro evento não aceitável.

A proposta deste trabalho é realizar a filtragem de sinais utilizando os coeficientes wavelet e o gráfico de controle de soma acumulada (CUSUM) (MONTGOMERY, 2009). O CUSUM (Seção 2.2) acumula e propaga informação de um elemento do processo analisado ao seu vizinho, sendo um sistema com memória. O CUSUM baseia-se na média μ e no desvio-padrão σ do processo para a análise.

Um exemplo simples de aplicação do gráfico CUSUM é na fotolitografia em que um fotorresistor deve ser aplicado a um waffer de silício, para a produção de semicondutores, e após inúmeros processos este deve passar por cozimento em fornos de alta temperatura. O gráfico CUSUM, a partir de um sistema de controle, por exemplo, pode verificar quanto a espessura deste material varia no seu cozimento, para um controle de qualidade. Se o forno estiver com problemas no controle da temperatura o cozimento ocorrerá de forma errada variando a espessura deste material, fazendo com que o sistema de controle possa identificar esta disparidade nas espessuras dos waffers e avisando ao operador de que existe uma disparidade no processo, ou ainda, que o processo está fora de controle (MONTGOMERY, 2009).

Assim, como no exemplo descrito, o gráfico de controle CUSUM pode ser eficiente para filtragem de coeficientes wavelet por desconsiderar atividades que são puramente aleatórias no processo. Considerando um sinal tomado por ruído a ideia de análise e atuação do gráfico CUSUM é bastante plausível. Quando o gráfico identificar uma disparidade nos coeficientes analisados esta disparidade não é mais aleatória e sim algo importante e que deve ser mantido no sinal.

Além disso, é necessária a adaptação do gráfico de controle para seu uso na filtragem de coeficientes wavelet. Esta adaptação será feita nos parâmetros que desenham o CUSUM para o seu funcionamento levando em conta o decaimento logarítmico da dispersão a cada nível da transformada wavelet, conforme constatada por Chen e Han (2005) (Seção 2.2.2.1).

Utilizando as estruturas da linguagem de programação *Python* foram feitos testes iterativos com o método de Monte Carlo (Seção 2.4) para descobrir quais parâmetros de design do CUSUM oferecem a melhor razão sinal-ruído (SNR), figura de mérito da tese, à filtragem de coeficientes wavelet (Seção 2.3).

2 METODOLOGIA

Inicialmente, nesta Seção, serão mostradas algumas notações, conceitos, definições e convenções adotadas que serão importantes para a compreensão do problema de filtragem de um sinal ruidoso e de como a metodologia foi adotada para a abordagem deste problema. Essa contextualização segue as referências (BAYER; KOZAKEVICIUS, 2010; DONOHO; JOHNSTONE, 1994).

Através de experimentos ou medições obtemos um sinal ruidoso $s(t),\ t \in [0, T]$, captado em um certo período de tempo T. Assumimos $s(t)$ como sendo formado por duas componentes:

$$s(t) = x(t) + z(t), \tag{2.1}$$

em que $x(t)$ representa o sinal na forma ideal, que é a componente a ser estimada e $z(t)$ é uma variável independente e identicamente distribuída com média $\mu = 0$ e variância σ^2, ou seja, é a componente que representa o ruído. Deste sinal, podemos extrair um valor que relaciona o sinal ideal ao ruído presente, chamado razão sinal-ruído (SNR), que pode ser dado pela razão das variâncias de $x(t)$ (σ^2_{ideal}) e $z(t)$ (σ^2_{ruido}) dados por:

$$\sigma^2_{ideal} = \frac{1}{T}\sum_{t=0}^{T-1}(x(t) - \overline{x})^2, \tag{2.2}$$

$$\sigma^2_{ruido} = \frac{1}{T}\sum_{t=0}^{T-1}(z(t) - \overline{z})^2, \tag{2.3}$$

em que $\overline{x} = \sum_{t=0}^{T-1} \frac{x(t)}{T}$ e $\overline{z} = \sum_{t=0}^{T-1} \frac{z(t)}{T}$. O valor SNR do sinal pode ser dado por:

$$SNR(\sigma^2_{ideal}, \sigma^2_{ruido}) = \frac{\sigma^2_{ideal}}{\sigma^2_{ruido}}. \tag{2.4}$$

O SNR pode ser expresso em escala logarítmica (decibéis ou dB):

$$\begin{aligned} SNR_{dB}(\sigma^2_{ideal}, \sigma^2_{ruido}) &= 10\log_{10}\left(\frac{\sigma^2_{ideal}}{\sigma^2_{ruido}}\right) \\ &= 10(\log_{10}\sigma^2_{ideal} - \log_{10}\sigma^2_{ruido}). \end{aligned} \tag{2.5}$$

Nosso objetivo principal é estimar a componente $x(t)$ por meio de técnicas de filtragem que envolvam a transformada wavelet do sinal $s(t)$. Assim, definimos nosso vetor de entrada como sendo o vetor que guarda as amostras do sinal captado (Equação (2.1)):

$$s = (s(i)),\ s(i) = s(t_i),\ t_i = \frac{i \times T}{2^J},\ J \in \mathbb{N},\ i = 0, 1, .., 2^J - 1. \tag{2.6}$$

Aqui, 2^J é o número de amostras do sinal, que é dado em potência de dois para que se

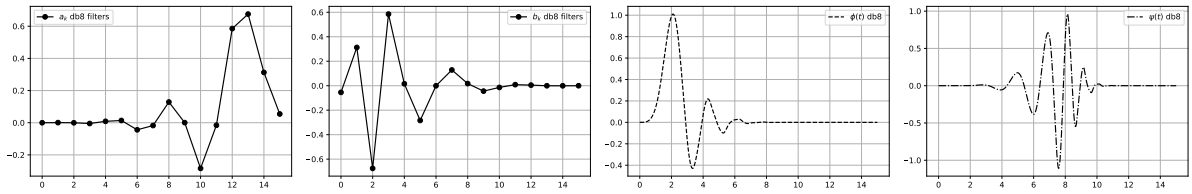
possa aplicar diretamente a transformada wavelet em s. Como a transformada wavelet decompõe um sinal em diferentes níveis de resolução, denotamos $J-1$ como sendo o nível mais fino e assim assumimos o sinal $s_J = s$.

2.1 TRANSFORMADA WAVELET

Neste trabalho vamos considerar a família de funções ortogonais de Daubechies: $\phi(t)$ função escala e $\psi(t)$ função wavelet. A propriedade fundamental desta família de funções é a associação de $\phi(t)$ e $\psi(t)$ com um conjunto de filtros, passa-baixa denotado por a_k e passa-alta b_k. Estes filtros são vetores de tamanho constante D para cada escolha de $\phi(t)$ e $\psi(t)$ que independem da escala na qual as funções estão discretizadas. No entanto, a_k e b_k carregam informações de suavidade e diferenciabilidade das próprias funções da base wavelet (NIELSEN, 1998). Estes conceitos são associados com a propriedade dos $N = \frac{D}{2}$ momentos nulos, na qual os polinômios de grau $0, \ldots, N-1$ são gerados apenas por funções escala, $\phi_{j,k}(t) = 2^{j/2}\phi(2^j t - k)$, independente do nível j e da posição k. Consequentemente, estes polinômios são ortogonais às funções wavelet. Com isso as funções da família de Daubechies são denotadas dbN, sendo N o número de momentos nulos. Na Figura 2.1 são apresentados os filtros a_k e b_k associados às funções $\phi(t)$ e $\psi(t)$ de Daubechies com $N = 8$ momentos nulos, que serão consideradas nas Seções 2.1.2, 2.1.3 e 3 para a realização da transformada wavelet. Nas Figuras 2.1 (c) e 2.1 (d) as funções foram geradas com 10 níveis de resolução (Seção 2.1.1).

Figura 2.1 – Wavelets de Daubechies com oito momentos nulos (db8)

(a) Filtros a_k (b) Filtros b_k (c) $\phi_{10,k}$ (d) $\psi_{10,k}$



Fonte: Do Autor, via Python utilizando as bibliotecas *PyWavelets* e *Matplotlib*.

Outra propriedade fundamental da família de Daubechies é a obtenção de algoritmos rápidos para a transformada wavelet de sinais, cuja formulação depende apenas do sinal discreto de entrada s_J e os filtros a_k e b_k (MALLAT, 1989).

As expressões da Equação (2.7) representam um nível da transformada wavelet direta (TWD), assumindo o sinal de entrada s_j, tendo como saída os vetores s_{j-1} e d_{j-1}. O vetor s_{j-1} contém os coeficientes de escala que são as informações médias do sinal. O vetor d_{j-1} contém os coeficientes wavelet da transformada que representam as variações em alta frequência do sinal s_j. Ou seja, as variações de informação entre os níveis j e

$j-1$ são dadas por:

$$\begin{aligned} s_{j-1,l} &= \sum_{k=0}^{D-1} a_k s_{j,2l+k} \\ d_{j-1,l} &= \sum_{k=0}^{D-1} b_k s_{j,2l+k}, \end{aligned} \tag{2.7}$$

com $l = 0, 1, \ldots, 2^{j-1} - 1$.

O processo de transformação começa quando $j = J$. Os demais níveis de decomposição são aplicados sempre sobre o vetor de coeficientes escala s_j. A transformação é completa quando no último nível de decomposição os vetores s_0 e d_0 contém apenas um elemento cada. Na prática, escolhe-se um nível mais grosseiro de representação ($J_0 > 0$) que ainda preserva informações relevantes do sinal original.

2.1.1 Representação Multirresolução

Como já foi introduzido, a análise multirresolução de Mallat (1989) é uma maneira de representar o sinal s_J em vários níveis de resolução, obtido por meio da decimação da TWD e é dado da seguinte forma. Com um nível de decomposição, no qual o tamanho de s_{J-1} e d_{J-1} é metade do tamanho de s_J, temos a seguinte relação:

$$s_J \leftrightarrow (s_{J-1}, d_{J-1}). \tag{2.8}$$

Para vários níveis de resolução, sendo $J-1$ o nível mais fino e J_0 o nível mais grosseiro, temos:

$$s_J \leftrightarrow (s_{J_0}, d_{J_0}, \ldots, d_{J-1}). \tag{2.9}$$

Desta forma, o sinal original $s(t)$ contínuo pode ser aproximado a partir de sua versão discreta s_J na base de funções wavelets de Daubechies, cuja decomposição em apenas um nível é dada da seguinte maneira:

$$s(t) \approx s_J(t) = \sum_{k=0}^{2^{J-1}-1} s_{J-1,k}\phi_{J-1,k}(t) + \sum_{k=0}^{2^{J-1}-1} d_{J-1,k}\psi_{J-1,k}(t), \tag{2.10}$$

em que $s_{J-1} = (s_{J-1,k})$ e $d_{J-1} = (d_{J-1,k})$, obtidos pela Equação (2.7).

Na expansão da Equação (2.11), J_0 é considerado o nível de resolução mais grosso e são apresentadas as decomposições do nível j entre $J-1$ e J_0. Essa expansão é a representação funcional do que é obtido pela TWD em multirresolução associado a repre-

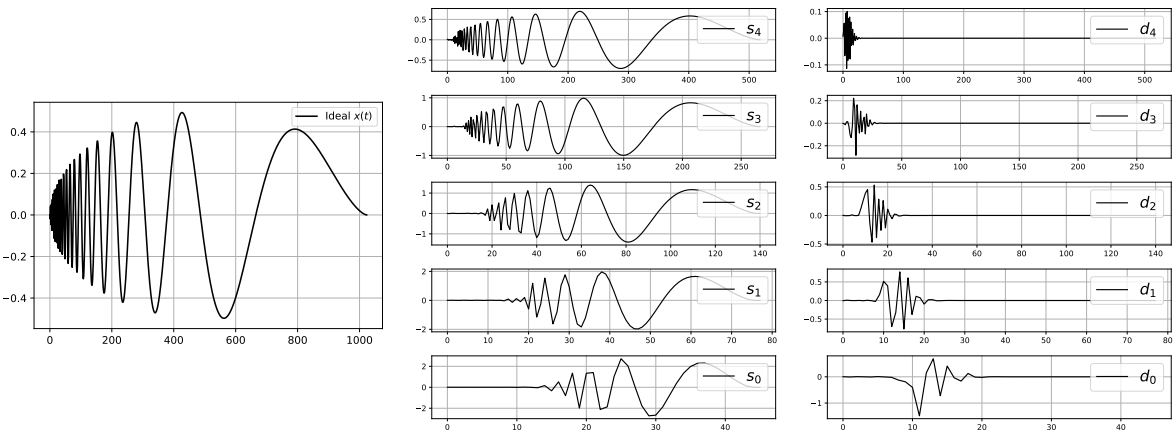
sentação $s_J \leftrightarrow (s_{J_0}, d_{J_0}, \ldots, d_{J-1})$.

$$s(t) = \sum_{k=0}^{2^{J_0}-1} s_{J_0,k}\phi_{J_0,k}(t) + \sum_{j=J_0}^{J-1}\sum_{k=0}^{2^j-1} d_{j,k}\psi_{j,k}(t). \tag{2.11}$$

As Figuras 2.2 e 2.3 mostram exemplos de decomposição em cinco níveis, com $J = 5$ e $J_0 = 0$. As Figuras 2.2 (b) e 2.2 (c) mostram a decomposição para o sinal de teste sem ruído (Figura 2.2 (a)) e as Figuras 2.3 (b) e 2.3 (c) para o mesmo sinal de teste com SNR_{dB} de 9.21 (Figura 2.3 (a)).

Figura 2.2 – Sinal *Doppler* ideal e sua decomposição em cinco níveis

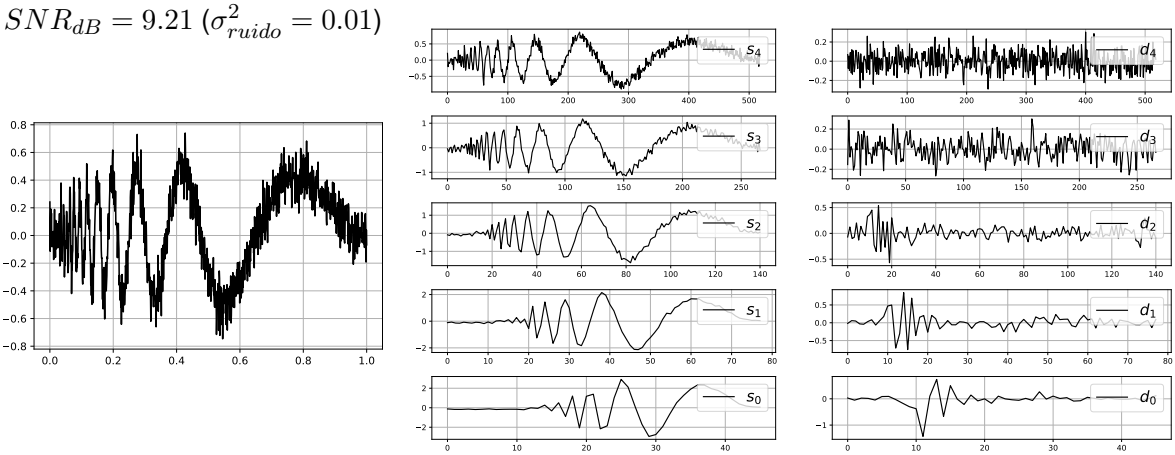
(a) Sinal original (b) Coeficientes escala (c) Coeficientes wavelet



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

Figura 2.3 – Sinal *Doppler* ruidoso e sua decomposição em cinco níveis

(a) Sinal original. $SNR_{dB} = 9.21$ ($\sigma^2_{ruido} = 0.01$) (b) Coeficientes escala (c) Coeficientes wavelet



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

Os dois primeiros vetores de coeficientes de cada uma das Figuras 2.2 e 2.3 são produtos do sinal original e os que seguem são resultado do vetor de coeficientes escala do nível acima.

Ao comparar a diferença dos coeficientes escala e dos coeficientes wavelet entre as figuras é possível observar a propagação do ruído, que foi adicionado ao sinal ideal, em cada nível de resolução. Comparando entre os coeficientes escala e os coeficientes wavelet percebesse a ação da TWD ao remover as informações de alta frequência e calcular um novo vetor de médias do vetor de coeficientes escala de nível maior.

O último vetor de coeficientes escala das figuras são as médias dos sinais originais após os cinco níveis da TWD (nível J_0). Estes são idênticos tanto para o sinal ideal, quanto para o sinal com ruído devido a baixa resolução neste nível.

O Apêndice A apresenta a decomposição dos demais sinais de teste da Figura 2.4 e informações numéricas, em tabelas, da diferença entre os coeficientes wavelet e escala do nível J_0 decompostos dos sinais ideal e do ruidoso. Na próxima seção iremos estudar técnicas de filtragem de ruído através da transformada wavelet.

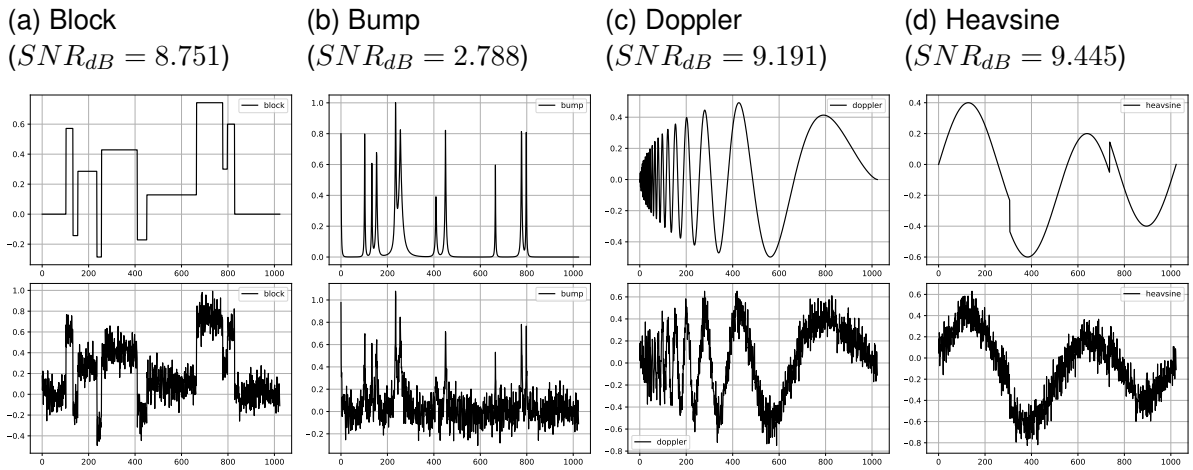
2.1.2 Filtragem via wavelet

A obtenção de sinais discretos via instrumentação ou armazenamento, por exemplo, pode diretamente comprometer seus dados. Esse fato motiva o resultado apresentado na Equação (2.1) que trata o sinal obtido como soma de duas componentes uma que representa o sinal desejado e a outra o ruído. Neste trabalho, serão utilizados dados sintéticos. O ruído será considerado como ocorrências de uma distribuição normal com média zero e variância σ^2. A Figura 2.4 mostra quatro sinais de teste[1] e suas versões ruidosas considerando um $\sigma^2_{ruido} = 0.01$. Os seus algoritmos podem ser encontrados no Apêndice B.

Após realizada a transformada wavelet do sinal s_J decomposto em $(s_{J_0}, d_{J_0}, \ldots, d_{J-1})$, o processo de filtragem é dado pelo truncamento (*threshold*) dos vetores $(d_{J_0}, \ldots, d_{J-1})$ para cada nível de resolução $j = J_0, \ldots, J-1$. Sendo assim, os coeficientes d_j são analisados pela função $trunca(\cdot)$ que compara esses coeficientes segundo algum parâmetro de referência (λ). Esse parâmetro é chamado de valor de truncamento ou limiar de corte, podendo ser definido de forma constante para todos os níveis (λ_{global}), dependente do nível j (λ_j) ou da posição de cada coeficiente ($\lambda_{j,k}$).

[1] Donoho e Johnstone (1994) descrevem que a escolha destes sinais foi feita por serem característicos em dados de imagens, de espectroscopia e de outros tipos de sinais científicos. O sinal *Block* é uma representação de como as comunicações digitais em algumas decodificações (JAY, 1989) ocorrem. O sinal *Bump* é característico para eletrocardiogramas e análises no domínio da frequência em geral. O sinal *Doppler* varre das frequências mais altas as mais baixas. E o sinal *Heavsine* é aparentemente periódico, exceto por suas descontinuidades e mudança de fase em algumas regiões.

Figura 2.4 – Sinais de teste ideais e com ruído ($\sigma^2_{ruido} = 0.01$)



Fonte: Do autor. Utilizando Python e as bibliotecas Numpy e Matplotlob.

As duas heurísticas de truncamento mais utilizadas são dadas da seguinte forma:

$$\tilde{d}_{j,i} = trunca^H_j(d_{j,i}) \quad = \quad \begin{cases} d_{j,i}, & \text{se } |d_{j,i}| > \lambda_j \\ 0, & c.c. \end{cases}, \tag{2.12}$$

$$\tilde{d}_{j,i} = trunca^S_j(d_{j,i}) \quad = \quad \begin{cases} sign(d_{j,i})(|d_{j,i}| - \lambda_j), & \text{se } |d_{j,i}| > \lambda_j \\ 0, & c.c. \end{cases}. \tag{2.13}$$

Na Equação (2.12), quando o valor do coeficiente analisado for maior do que o limiar, este valor será mantido inalterado. Ele será substituído por zero (truncado), apenas quando for considerado não significativo (menor do que o limiar). Esta estratégia é denominada *hard-threshold*, $trunca^H_j(\cdot)$. Na Equação (2.13), os valores significativos sofrem um amortecimento com a magnitude do limiar λ e os coeficientes não significativos são novamente anulados. Esta estratégia é dita *soft-threshold*, $trunca^S_j(\cdot)$.

Considerando $\tilde{d}_j$ como sendo o vetor com os coeficientes wavelet de cada nível j truncados por uma destas estratégias, a análise multirresolução correspondente é denotada por:

$$\tilde{s}_J \leftrightarrow (s_{J_0}, \tilde{d}_{J_0}, \ldots, \tilde{d}_{J-1}), \tag{2.14}$$

em que $\tilde{s}_J$ é uma estimativa para o sinal sem ruído.

2.1.3 Determinação do limiar de corte

Na clássica referência de Donoho e Johnstone (1994) os autores apresentaram o método Universal Threshold, também chamado de *VisuShrink*, para obtenção do limiar λ para o truncamento dos coeficientes wavelet do sinal ruidoso. O método é dito universal por assumir todos os valores λ_j iguais ($\lambda = \lambda_{global}$) e em sua formulação, dada pela

Equação (2.15), considera o tamanho do vetor de coeficientes wavelet do nível mais fino de decomposição, $N_{J-1} = 2^{J-1}$ e um desvio absoluto mediano, também chamado de estimador robusto $\hat{\sigma} = \frac{mediana(|d_{J-1}|)}{0.6745}$. O relato de Donoho e Johnstone para o uso deste estimador é que neste nível de coeficientes, em específico, a presença de ruído puro é significativamente alto, logo o ruído tem um viés superior ao dado neste nível e com um estimador robusto esse viés é efetivamente controlado. Gross (2003) descreve que este estimador é especialmente razoável quando se sabe que os elementos no vetor d_{J-1} são simétricos e distribuídos em torno de zero:

$$\lambda_{global} = \hat{\sigma}\sqrt{2\log N_{J-1}}. \tag{2.15}$$

Em Donoho e Johnstone (1995) é apresentado o método *SureShrink*, cuja abordagem é adaptativa, ou seja, os valores λ_j podem ser diferentes, dependendo do módulo dos coeficientes wavelet e de cada um dos níveis da transformação. A Equação (2.16) utiliza o princípio de Stein (1981) para a obtenção do limiar, também chamado de SURE threshold por Donoho e Johnstone (1995). Novamente, N_j é o tamanho do vetor d_j e t é um parâmetro para definição do λ_j na Equação (2.17).

O termo $\#\{i : |d_{j,i}| \leq t\}$ cria um conjunto de todos os valores de d_j que são menores ou iguais ao parâmetro t e retorna a quantidade de elementos neste conjunto ($\#\{\cdot\}$ = cardinalidade):

$$SURE(t, d_j) = N_j - 2 \times \#\{i : |d_{j,i}| \leq t\} + \sum_{i=0}^{N_j-1} \min(|d_{j,i}|, t)^2. \tag{2.16}$$

O limiar λ_j para cada vetor d_j é:

$$\lambda_j = \text{argmin}_{0 \leq t \leq \sqrt{2\log N_j}} SURE(t, d_j). \tag{2.17}$$

O Algoritmo 1 mostra o processo de remoção do ruído (filtragem) a partir do truncamento dos coeficientes wavelet do sinal de entrada s_J. $\tilde{s}_J = TIW(\hat{s}_J)$ é obtido via transformada inversa do sinal decomposto com os coeficientes wavelet truncados $\hat{s}_J$ (Equação 2.14). Assim, uma expansão para $\tilde{s}_J$ é obtida de forma análoga à Equação (2.11), quando são considerados os coeficientes $\tilde{d}_{j,k}$ truncados ao invés dos originais $d_{j,k}$.

Uma das contribuições deste trabalho é a proposta de uma nova heurística de determinação do limiar λ_j, que ao invés de considerar o vetor de coeficientes wavelet d_j diretamente para sua obtenção, considera gráficos de controle CUSUM. Na próxima seção este conceito será apresentado, assim como o algoritmo correspondente à metodologia proposta.

Algoritmo 1: PROCEDIMENTO DE FILTRAGEM $filtering(\cdot)$

Entrada: s_J: Sinal com ruído.
Saída: $\hat{s}_J$: Sinal filtrado.

início
  $TWD(s_J) = (s_{J_0}, d_{J_0}, \ldots, d_{J-1})$
  $trunca(d_{J_0}, \ldots, d_{J-1}) = (\tilde{d}_{J_0}, \ldots, \tilde{d}_{J-1})$
  $TIW(\hat{s}_J) = TIW(s_{J_0}, \tilde{d}_{J_0}, \ldots, \tilde{d}_{J-1}) = \tilde{s}_J$
fim
retorna $\hat{s}_J$

2.2 GRÁFICO DE CONTROLE DE SOMA ACUMULADA (CUSUM)

O gráfico de controle de soma acumulada (CUSUM) é um método de controle estatístico de processo (SPC) para pequenas variações dos sinais analisados. Gráficos de controle, em geral, possuem três informações básicas destes dados: uma reta que indica a média e outras duas linhas chamadas de limites de controle superior (S^+) e inferior (S^-), como ilustrado nas Figuras 2.5 (b) e 2.5 (d).

Neste método, assume-se que $X = (X_i)_{i=0}^{N-1}$ têm média $\overline{X} = \sum_{i=0}^{N-1} \frac{X_i}{N}$, têm desvio-padrão $\sigma = \sqrt{\frac{\sum_{i=0}^{N-1}(X_i - \overline{X})^2}{N}}$ e o parâmetro K é definido por $K = \frac{1}{2}\sigma$. E ainda, os limites de controle S_i^+ e S_i^- são dados por:

$$S_i^+ = \max(0, (X_i - \overline{X} - K + S_{i-1}^+)) \quad (2.18)$$

$$S_i^- = \min(0, (X_i - \overline{X} + K + S_{i-1}^-)) \quad (2.19)$$

$$S_{-1}^+ = S_{-1}^- = 0.$$

Para cada valor X_i analisado, podemos interpretar que primeiro o gráfico desconta a média $\overline{X}$ dos valores X_i. Desta forma, os valores são trazidos para o entorno da média zero. Após, o valor de K é descontado do limite S_i^+ e incrementado do limite S_i^-. Com isso, o gráfico monta uma faixa de tolerância dos dados utilizando-se da métrica do desvio-padrão. E, por fim, incrementa-se o limite de controle do dado anterior (S_{i-1}^+ ou S_{i-1}^-) que passa pela mesma aritmética. O último termo da operação dá a característica de acumulação a este gráfico de controle.

Se os valores de X_i se distanciam demais da média e, além disso, acima do meio desvio-padrão (K) que o gráfico tolera, por exemplo, uma progressão linear dos dados torna-se uma progressão geométrica para os limites de controle, como pode ser visto na Figura 2.5. No caso contrário, pode-se simplesmente zerar o limite de controle pelo fato de tal amostra X_i estar próximo da média e com um desvio aceitável, dado pelas operações $min(\cdot)$ e $max(\cdot)$.

O CUSUM, em especial, possui outras duas linhas características chamadas de intervalo de decisão superior (H^+) e inferior (H^-). Esses intervalos, quando comparados

com os valores S_i^+ e S_i^-, caracterizam-se os dados analisados estão dentro ou fora de controle e é comumente dado por (MONTGOMERY, 2009):

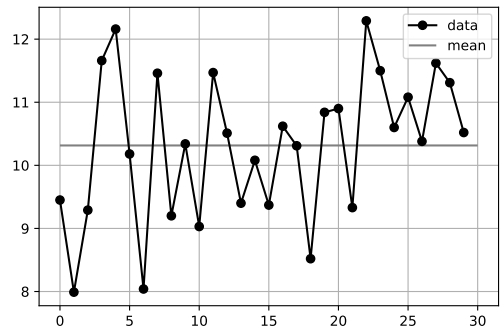
$$H^+ = -H^- = h \times \sigma, \tag{2.20}$$

com $h = 5$. Os valores de H^+ e H^-, definidos na Equação (2.20) ainda serão discutidos para ajustes do gráfico de controle para a filtragem dos coeficientes wavelet (Equação (2.23)).

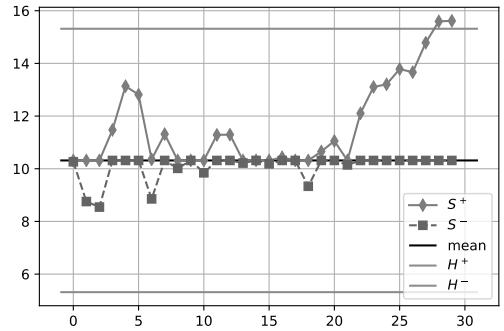
As Figuras 2.5 (a) e 2.5 (c) ilustram exemplos a serem analisados pelo gráfico. Utilizou-se em (a) dados aleatórios e em (c) dados amostrados de uma reta inclinada. As Figuras 2.5 (b) e 2.5 (d) apresentam os limites de controle (S_i^+ e S_i^-) ao redor da média ($mean$) para cada vetor de exemplo, presente ainda nas mesmas figuras os intervalos de decisão para o S^+ (H^+) e para o S^- (H^-). Nos exemplos podemos observar que os limites de controle ultrapassam os intervalos de decisão em algum momento, são estes os momentos que consideramos que os dados estão fora de controle.

Figura 2.5 – Dois exemplos de análise pelo CUSUM

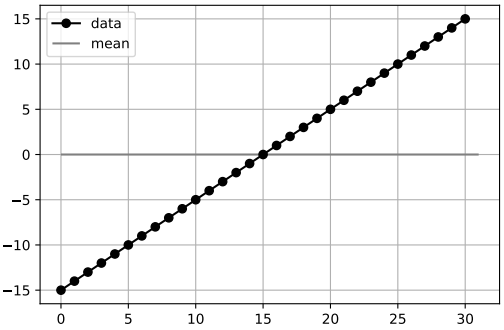
(a) Vetor de dados aleatórios



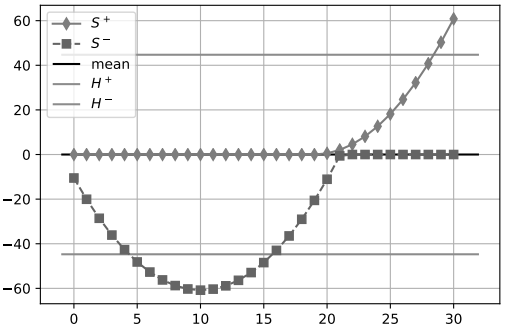
(b) Linhas da análise do CUSUM da parte (a)



(c) Vetor de dados da reta inclinada



(d) Linhas da análise do CUSUM da parte (c)



Fonte: Do autor. Utilizando Python e as bibliotecas Numpy e Matplotlob.

2.2.1 CUSUM e os coeficientes wavelet

O CUSUM, como método de truncamento para a filtragem via wavelet, faz algo semelhante ao método *SureShrink*, no qual os limiares de corte λ_j são determinados de forma adaptativa, ou seja, dependem de cada vetor de coeficientes analisados. Para cada vetor de coeficientes d_j, são calculados os limites superior (S^+) e inferior (S^-) de controle. Reescrevemos as Equações (2.18) e (2.19) agora com vetor de dados d_j sendo X, $\overline{d}_j$ sendo a média $\overline{X}$ dos coeficientes wavelet no nível j e σ_j sendo o desvio-padrão no nível j:

$$\begin{aligned} S^+_{j,i} &= \text{max}(0, (d_{j,i} - \overline{d}_j - \frac{1}{2}\sigma_j + S^+_{j,i-1})) && (2.21)\\ S^-_{j,i} &= \text{min}(0, (d_{j,i} - \overline{d}_j + \frac{1}{2}\sigma_j + S^-_{j,i-1})) && (2.22)\\ S^+_{j,-1} = S^-_{j,-1} &= 0. \end{aligned}$$

Depois de se obter os limites de controle para cada vetor d_j, passamos para a fase de truncamento dos coeficientes que assim como no *SureShrink*, os intervalos de decisão (H^+ e H^- agora chamados de H^+_j e H^-_j) serão diferenciados para cada nível j e dado por:

$$H^+_j = -H^-_j = h_j \times \sigma_j, \tag{2.23}$$

em que na Equação (2.20) o h agora h_j foi dado igual a cinco.

A Equação (2.27) apresentará uma configuração para o valor de h_j de acordo com uma definição de parâmetros descrita adiante.

2.2.2 Truncamento via CUSUM

No *VisuShrink* e no *SureShrink* o truncamento baseia-se apenas na comparação entre λ e os coeficientes wavelet. Já no CUSUM o *threshold* é feito a partir dos limites de controle $S^+_{j,i}$ e $S^-_{j,i}$ dos coeficientes e dos intervalos de decisão H^+_j e H^-_j.

O *threshold* do CUSUM ($trunca^{cusum}_j(\cdot)$) trunca os elementos na forma:

$$\tilde{d}_{j,i} = trunca^{cusum}_j(d_{j,i}) = \begin{cases} d_{j,i}, & \text{se } (S^+_{j,i} > H^+_j) \text{ ou } (S^-_{j,i} < H^-_j) \\ 0, & \text{se } (S^+_{j,i} \leq H^+_j) \text{ e } (S^-_{j,i} \geq H^-_j) \end{cases}. \tag{2.24}$$

Sendo assim, se o elemento analisado $d_{j,i}$ estiver dentro de controle, este é zerado, considera-se que a natureza do valor truncado é puramente aleatória. Podendo ser descartado sem causar danos para as informações relevantes contidas no sinal original.

Apesar da literatura oferecer um embasamento de uso dos valores K e H^+, defini-

dos nas Equações (2.18) e (2.20), como sendo (MONTGOMERY, 2009):

$$K = \frac{1}{2}\sigma \tag{2.25}$$

$$H^{+} = 5\sigma, \tag{2.26}$$

estes valores são insuficientes para filtragem via wavelet. Uma das contribuições deste trabalho é adaptar esses valores, no caso valores de h_j definidos na Equação (2.23), para um melhor resultado da filtragem via wavelet usando o CUSUM como método de filtragem dos coeficientes. A adaptação desses valores será baseado no trabalho de Chen e Han (2005) como será mostrado a seguir.

2.2.2.1 Adaptação do gráfico de controle CUSUM para a filtragem via wavelet

No trabalho de Chen e Han (2005) foi elaborado um teste utilizando imagens (Figura 2.6) demonstrando que o desvio-padrão médio[2] dos coeficientes wavelet $\overline{\sigma}_j$ possuíam um padrão de decaimento logarítmico em função do nível wavelet j, como pode ser visto na Figura 2.7. O Anexo A apresenta os resultados originais do artigo do autor.

Figura 2.6 – Imagens utilizadas para extração dos coeficientes wavelet e para o teste do decaimento do desvio-padrão

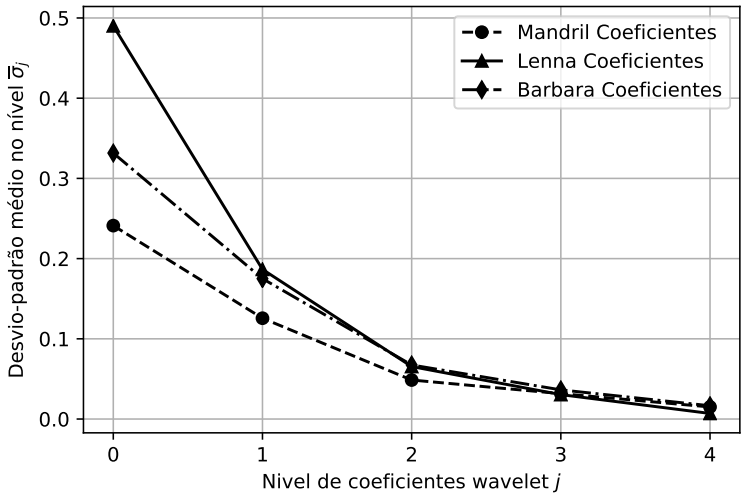
(a) Lena (b) Barbara (c) Mandrill

Fonte: Public-Domain Test Images for Homeworks and Projects. Disponível em: *https://homepages.cae.wisc.edu/ ece533/images/*. Acessado em 15 de outubro de 2018.

A Figura 2.8 mostra o mesmo teste feito por Chen, mas com os sinais de teste que já foram apresentados neste trabalho (Figura 2.4). O teste mostra que a análise de Chen se confirma parcialmente para sinais 1D. Os coeficientes apresentam uma evolução no sentido logarítmico, porém com algumas diferenças quando comparado ao gráfico de decaimento do desvio-padrão das imagens de teste.

[2]Desvio-padrão médio porque os coeficientes resultante de uma transformação wavelet 2D são dados em três vetores diferentes, chamados horizontais, verticais e diagonais.

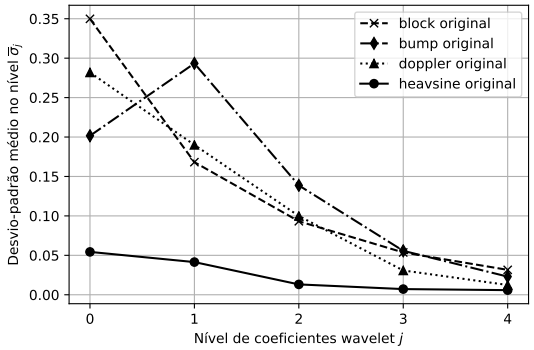
Figura 2.7 – Gráfico com a média dos desvios-padrão das imagens de teste, utilizando transformada wavelet Daubechies com oito momentos nulos



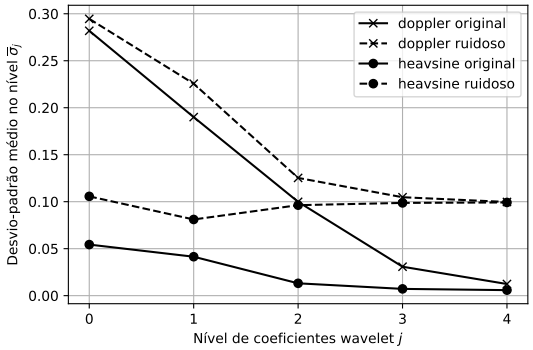
Fonte: Do autor. Utilizando *Python* e as bibliotecas *Numpy*, *PyWavelets* e *Matplotlib*.

Figura 2.8 – Gráfico com a média dos desvios-padrão dos sinais de teste em cada nível j da transformada wavelet. $\sigma^2_{ruido} = 0.01$

(a) Para os sinais ideais



(b) Para alguns dos sinais ideais e com ruído



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Numpy* e *Matplotlib*.

Tendo em vista que o algoritmo de threshold do CUSUM utiliza como parâmetro de truncamento o desvio-padrão dos coeficientes (Equações (2.20) e (2.23)) e levando em conta o já citado ajuste necessário para analisar e truncar este tipo de coeficientes, viu-se na ideia de Chen uma forma de obter melhores resultados para a filtragem wavelet.

A proposta visa ajustar os intervalos de decisão (Equação (2.23)) devido ao decaimento logarítmico dos desvios-padrão dos coeficientes wavelet. Então foi proposto o modelo da Equação (2.27) que desenha um variação exponencial para ajustar o valor h_j dos intervalos de decisão de cada nível j, utilizando cinco níveis de coeficientes wavelet e

dois parâmetros abertos chamados de a e b, em que:

$$h_j = a\ \log_{10}(b(5-j)), \tag{2.27}$$

com j variando de 0 a 4 sendo o nível do coeficiente wavelet a ser analisado pelo CUSUM ($j = 0$ é o nível mais grosseiro e $j = 4$ é o nível mais fino). Então o método para a escolha dos parâmetros abertos da Equação (2.27) foi o de Monte Carlo (Seção 2.4). Foram variados os valores de a e b num intervalo e de acordo com a figura de mérito (Seção 2.3) foram eleitos os melhores parâmetros a e b para se adaptar o gráfico CUSUM para a filtragem via wavelet com cinco níveis, a partir dos sinais de teste apresentados na Figura 2.4 e do ruído adicional descrito na Equação (2.1).

2.3 FIGURA DE MÉRITO: RAZÃO SINAL-RUÍDO

A razão sinal-ruído (SNR) será a métrica utilizada para validar a técnica proposta na Equação (2.27). O SNR já foi apresentado nas Equações (2.4) e (2.5). O Algoritmo 2 mostra como a qualidade de filtragem de sinal será medida.

Algoritmo 2: FUNÇÃO PARA CÁLCULO DA QUALIDADE DA FILTRAGEM $SNR_{funcao}(\cdot)$

Entrada: σ^2_{ideal}: Variância do sinal ideal
σ^2_{ruido}: Variância do ruído aplicado inicialmente
$\sigma^2_{residuo}$: Variancia da diferença do sinal filtrado pelo ideal.
Saída: $SNR_{dB,final}$: Razão Sinal-Ruído do sinal filtrado.

1 **início**
  `// A função` $SNR_{dB}(\cdot)$ `refere-se a Equação (2.5)`
2 $SNR_{dB}(\sigma^2_{ideal}, \sigma^2_{ruido}) = SNR_{dB,ruidoso}$
3 $SNR_{dB}(\sigma^2_{ideal}, \sigma^2_{residuo}) = SNR_{dB,filtrado}$
4 $SNR_{dB,filtrado} - SNR_{dB,ruidoso} = SNR_{dB,final}$
5 **fim**
6 **retorna** $SNR_{dB,final}$

Existem várias formas de se calcular esta métrica e a escolhida para este trabalho baseia-se na razão das variâncias do sinal e dos níveis de ruído (WELVAERT; ROSSEEL, 2013). O $SNR_{dB,ruidoso}$ (no Algoritmo 2) é a métrica calculada do sinal com ruído e é dado pela razão da variância do sinal ideal e a variância do ruído aplicado. O $SNR_{dB,filtrado}$ é a métrica calculada do sinal que já passou pelo processo de transformação e filtragem e é dado pela razão da variância do sinal ideal e a variância do ruído ainda existente no sinal filtrado. O $SNR_{dB,final}$ conterá a quantidade de ruído que pôde ser removido deste sinal no processo.

Uma possível simplificação para o $SNR_{dB,final}$ do Algoritmo 2 é utilizando a Equação (2.5):

$$SNR_{dB,ruidoso} = 10(\log_{10}\sigma^2_{ideal} - \log_{10}\sigma^2_{ruido}) \quad (2.28)$$

$$SNR_{dB,filtrado} = 10(\log_{10}\sigma^2_{ideal} - \log_{10}\sigma^2_{residuo}), \quad (2.29)$$

logo, operando o $SNR_{dB,final}$:

$$\begin{aligned} SNR_{dB,final} &= SNR_{dB,filtrado} - SNR_{dB,ruidoso} \\ &= 10(\log_{10}\sigma^2_{ideal} - \log_{10}\sigma^2_{residuo}) - 10(\log_{10}\sigma^2_{ideal} - \log_{10}\sigma^2_{ruido}) \\ &= 10(-\log_{10}\sigma^2_{residuo}) - 10(-\log_{10}\sigma^2_{ruido}) \\ &= 10(\log_{10}\sigma^2_{ruido} - \log_{10}\sigma^2_{residuo}). \end{aligned} \quad (2.30)$$

Ou ainda:

$$SNR_{dB,final} = 10\log_{10}\left(\frac{\sigma^2_{ruido}}{\sigma^2_{residuo}}\right). \quad (2.31)$$

2.4 MÉTODO DE MONTE CARLO PARA VARIAÇÃO DOS PARÂMETROS

Para a determinação dos valores a e b da Equação (2.27) foi utilizado um algoritmo semelhante ao Algoritmo 3. A cada simulação foi configurado um tipo de sinal de teste (Figura 2.4) e uma variação σ^2_{ruido} do ruído, chamado de cenário, para rodar no Algoritmo.

Para cada cenário foram testados os valores de a e b e simulado com 1000 réplicas de Monte Carlo e se obteve um valor médio de cada cenário para cada valor de a e b chamado no Algoritmo 3 de $\overline{SNR}_{dB,final}(a,b)$. Com essa tabela de valores $\overline{SNR}_{dB,final}(a,b)$ foi utilizado um gráfico chamado mapa de calor, ou também *heatmap*, para representar estes dados. Os gráficos estão presentes no Apêndice C. Haverá um *heatmap* para cada cenário de simulação (sinal de teste e a variância σ^2_{ruido} do ruído).

Algoritmo 3: PROCEDIMENTO DE MONTE CARLO PARA OBTENÇÃO DOS VALORES DE $\overline{SNR}_{dB,final}$.

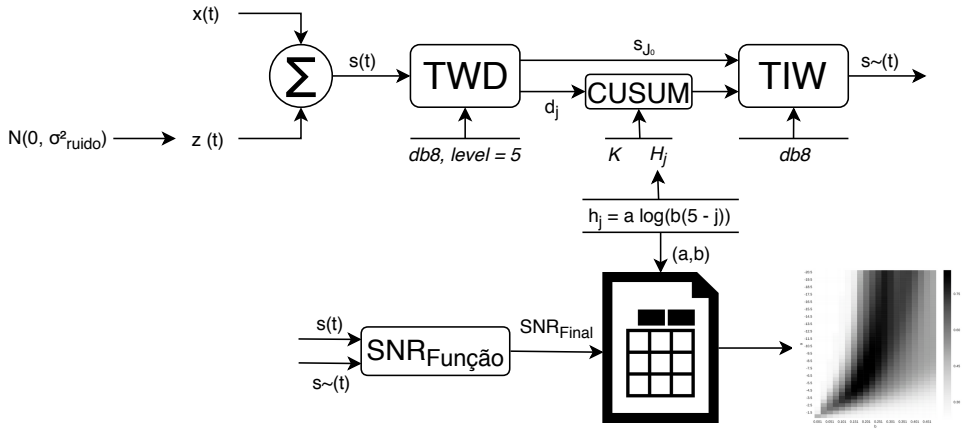
Entrada: x: Sinal de teste ideal a ser analisado
σ^2_{ruido}: Variância do ruído a ser aplicado ao sinal
a_{lista}, b_{lista}: Intervalo de valores a serem testados
Saída: $\overline{SNR}_{dB,final}$: Tabela com valores de SNR_{dB} médio para cada valor a e b testado no algoritmo referente ao x e σ^2_{ruido} dado na **Entrada**.

início
  // Utiliza as funções de filtragem ($filtering(\cdot)$, Algoritmo 1) e de cálculo do $SNR_{dB,final}$ ($SNR_{funcao}(\cdot)$, Algoritmo 2)
  // $\sigma^2(\cdot)$ simboliza a variância de $(\cdot)$
  $n_{iteracoes} = 1000$
  para a ***em*** a_{lista} **faça**
    para b ***em*** b_{lista} **faça**
      $h = a \log_{10}(b(5-j))$
      $it = 0$
      $acc_{SNR_{dB,final}} = 0$
      enquanto $it < n_{iteracoes}$ **faça**
        $z = N(0, \sigma^2_{ruido})$ // Nova amostra a cada iteração
        $s = x + z$
        $\tilde{s} = filtering(s)$
        $\tilde{z} = \tilde{s} - x$
        $\sigma^2_x = \sigma^2(x)$; $\sigma^2_{\tilde{s}} = \sigma^2(\tilde{s})$; $\sigma^2_{\tilde{z}} = \sigma^2(\tilde{z})$
        $SNR_{funcao}(\sigma^2_x, \sigma^2_{ruido}, \sigma^2_{\tilde{s}}, \sigma^2_{\tilde{z}}) = SNR_{dB,final}$
        $acc_{SNR_{dB,final}} = acc_{SNR_{dB,final}} + SNR_{dB,final}$
        $it++$
      fim
      $\overline{SNR}_{dB,final}(a,b) = \frac{acc_{SNR_{dB,final}}}{n_{it}}$
    fim
  fim
fim
retorna $\overline{SNR}_{dB,final}$

3 RESULTADOS OBTIDOS

A Figura 3.1 apresenta um fluxo simplificado da metodologia aplicada apresentada anteriormente.

Figura 3.1 – Fluxograma básico da metodologia aplicada em cada cenário



Fonte: Do Autor.

Cada sinal $x(t)$, na Figura 3.1, é um sinal de teste apresentado na Figura 2.4 e as variâncias σ^2_{ruido} de ruído, também na Figura 3.1, que foram utilizadas são 10 no total com valores de 0.001 a 0.01. Com 4 sinais de teste e 10 variâncias de ruído a quantidade de cenários (Seção 2.4) gerados foi igual a 40.

O Apêndice C apresenta os gráficos para cada cenário que foi testado, em que cada um apresenta os índices de $\overline{SNR}_{dB,final}$ em função dos parâmetros a e b.

3.1 PARÂMETROS ESCOLHIDOS PARA O GRÁFICO CUSUM

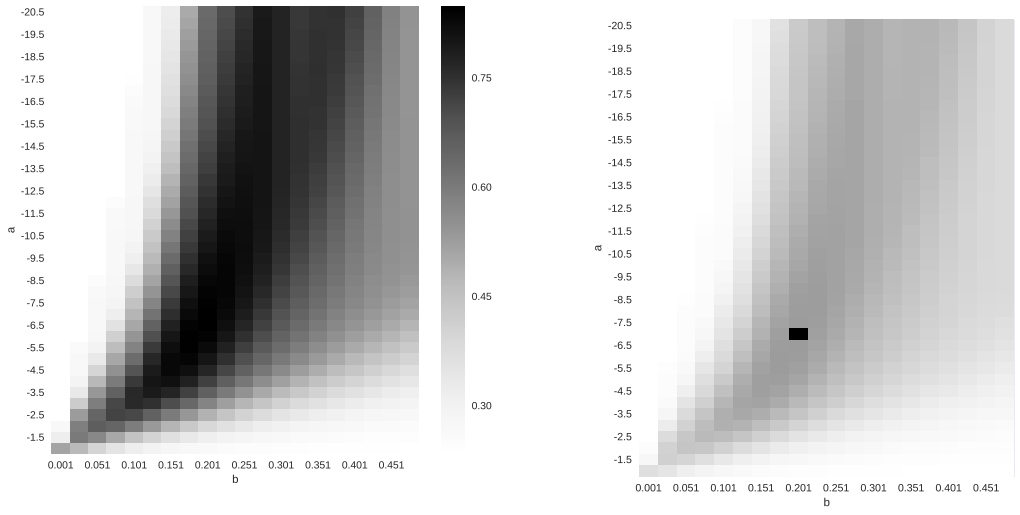
Para a escolha dos parâmetros da Equação (2.27) todos os gráficos do Apêndice C, gerados por Monte Carlo, foram acumulados e obteve-se o gráfico médio da Figura 3.2 (a); quanto mais escuro, maior é a qualidade da filtragem (Seção 2.3). A Figura 3.2 (b) apresenta o ponto máximo das médias da Figura 3.2 (a), ou seja, o melhor valor entre todos os cenários testados, dado pelos valores $a = -7.0$ e $b = 0.201$. Estes valores para a e b são os que melhor ajustam h_j da Equação (2.27) para a filtragem de sinais utilizando o gráfico CUSUM, chamando-o de CUSUM adaptado.

Os parâmetros ($a = -7.0$ e $b = 0.201$), obtidos a partir da Figura 3.2, foram fixados para calcular o valor de h_j da Equação (2.27). O gráfico é apresentado na Figura 3.3.

Figura 3.2 – Gráficos com todos os mapas de calor dos cenários acumulados, para a na vertical e para b na horizontal, valores médios e normalizados

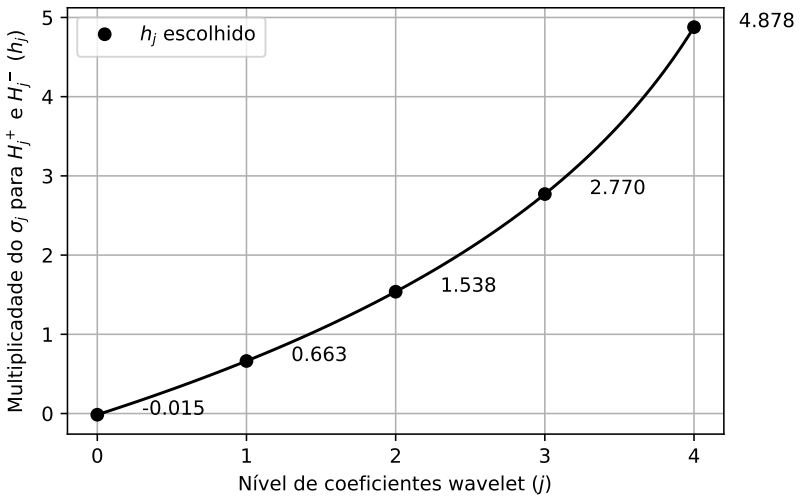
(a) Gráfico original

(b) Gráfico com valor máximo evidenciado



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura 3.3 – Gráfico que mostra os valores de h_j para os cinco níveis da transformada wavelet, utilizando a Equação (2.27) e os parâmetros encontrados na Figura 3.2 ($a = -7.0$ e $b = 0.201$)



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Numpy* e *Matplotlib*.

Os valores h_j da Figura 3.3 serão os parâmetros dos intervalos de decisão na Equação (2.23) para cada nível j dos coeficientes wavelet. A curva mostra um valor negativo para o h_j no nível mais grosseiro, o h_0 fazendo com que os intervalos de decisão H_0^+ e H_0^- sejam negativos e positivos, respectivamente. Com os limites de controle $S_{0,i}^+$ e $S_{0,i}^-$ não podendo assumir valores negativos e positivos por causa das funções $min(\cdot)$ e $max(\cdot)$ os coeficientes wavelet $d_{0,i}$ nunca estarão fora de controle, logo não serão truncados.

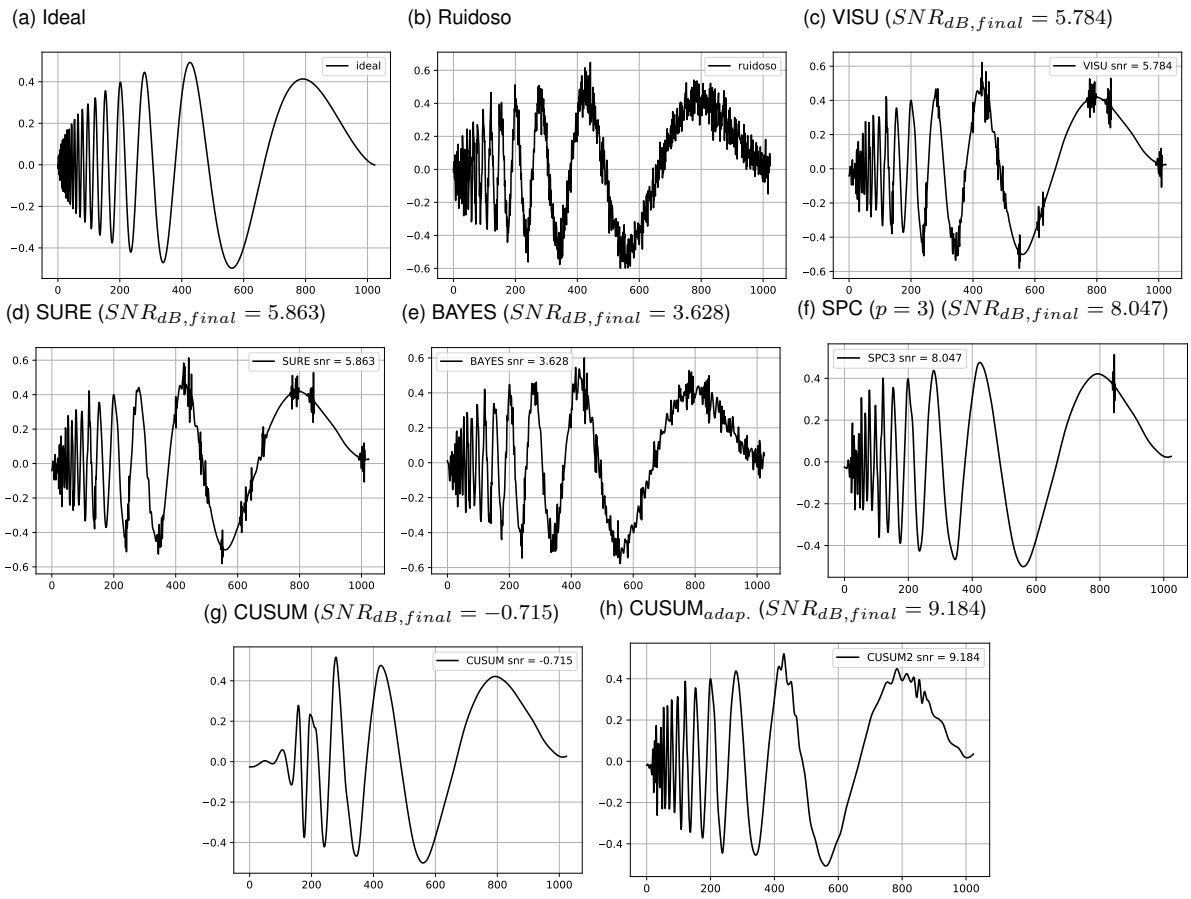
Inicialmente, a ideia era utilizar um valor que acompanhasse o decaimento logarítmico dos desvios-padrão dos coeficientes, entretanto o que se mostrou eficiente para o ajuste foi uma variação exponencial do h_j de forma que equilibrasse os valores de desvio-padrão em decaimento, como mostrou a Figura 2.8, para os intervalos de decisão (Equação (2.23)).

3.2 COMPARAÇÃO DO CUSUM ADAPTADO COM OUTROS MÉTODOS EM TESTE UNITÁRIO

Na Figura 3.4 está o sinal de teste *Doppler* ideal (Figura 3.4 (a)) com uma variância σ^2_{ruido} de ruído igual a 0.005 (Figura 3.4 (b)). Este sinal foi filtrado utilizando os algoritmos *VisuShrink* e *SureShrink* apresentados na Seção 2.1.3, Figuras 3.4 (c) e 3.4 (d), respectivamente. Outros algoritmos de filtragem via wavelet foram testados como o *BayesShrink* (CHANG; YU; VETTERLI, 2000) (Figura 3.4 (e)) e *SPC-Threshold* (BAYER; KOZAKEVICIUS, 2010) com p = 3 (Figura 3.4 (f)). Seus pseudo-códigos estão presentes no Apêndice D. Também foi testado o *CUSUM Tradicional*, com $H = 5\sigma$ (Figura 3.4 (g)) e o CUSUM adaptado, proposto neste trabalho, com $H^+ = -H^- = -7.0\log_{10}(0.201(5-j)) \times \sigma_j$ (Figura 3.4 (h)). Para cada filtragem, nas legendas, estão presentes os índices $SNR_{dB,final}$ de qualidade.

Pode-se observar que, neste cenário (sinal de teste *Doppler* e $\sigma^2_{ruido} = 0.005$, Seção 2.4), o método utilizando o CUSUM adaptado se mostrou melhor que os outros métodos de filtragem. Nas Figuras da 3.4 as partes de alta frequência do sinal de teste *Doppler* foram preservadas exceto no CUSUM Tradicional, Figura 3.4 (g), mostrando que o gráfico com os valores que a literatura recomenda (Equações (2.25) e (2.26)) para o design do CUSUM não são suficientes para a filtragem wavelet em todos os casos. O método *SPC-Threshold* teve um segundo melhor resultado, em $SNR_{dB,final}$, seguido do *VisuShrink*, *SureShrink* e do *BayesShrink*.

Figura 3.4 – Gráfico com a filtragem do sinal de teste *Doppler* com uma $\sigma^2_{ruido} = 0.005$



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *Numpy* e *Matplotlib*.

3.3 COMPARAÇÃO DO CUSUM ADAPTADO COM OUTROS MÉTODOS EM TESTES ITERATIVOS

As Tabelas 3.1, 3.2, 3.3 e 3.4 mostram o valor de $\overline{SNR}_{dB,final}$ obtido via simulação de Monte Carlo (10000 réplicas) para cada algoritmo de truncamento, incluindo o CUSUM adaptado, e com diferentes valores de variância σ^2_{ruido} do ruído. Os valores em destaque são os melhores de cada cenário. A segunda coluna apresenta o SNR_{dB} inicial de cada sinal com ruído e a coluna do CUSUM adaptado possui um índice que indica a posição entre os demais métodos testados.

Tabela 3.1 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Block**

σ^2_{ruido}	$SNR_{dB,ruidoso}$	VISU	SURE	BAYES	SPC ($p = 3$)	CUSUM	CUSUM$_{adap.}$
0.001	18.50	**3.22**	1.96	0.77	-8.42	-10.21	-0.15[4]
0.003	13.72	**3.39**	2.44	2.21	-4.27	-5.55	2.57[2]
0.005	11.51	3.60	2.65	2.89	-2.42	-3.38	**3.62**[1]
0.007	10.05	3.74	2.78	3.35	-1.20	-1.94	**4.23**[1]
0.009	8.95	3.87	2.92	3.74	-0.29	-0.88	**4.63**[1]

Fonte: Do Autor.

Tabela 3.2 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Bump**

σ^2_{ruido}	$SNR_{dB,ruidoso}$	VISU	SURE	BAYES	SPC ($p = 3$)	CUSUM	CUSUM$_{adap.}$
0.001	12.80	**3.35**	1.27	1.52	-3.82	-9.27	-0.70[4]
0.003	8.03	**3.62**	2.76	2.69	-0.47	-4.61	2.84[2]
0.005	5.81	3.86	3.32	3.32	0.92	-2.44	**4.19**[1]
0.007	4.35	4.06	3.64	3.74	1.80	-1.01	**4.89**[1]
0.009	3.26	4.21	3.89	4.05	2.45	0.05	**5.31**[1]

Fonte: Do Autor.

Tabela 3.3 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Doppler**

σ^2_{ruido}	$SNR_{dB,ruidoso}$	VISU	SURE	BAYES	SPC ($p = 3$)	CUSUM	CUSUM$_{adap.}$
0.001	19.30	5.76	5.43	2.84	**5.89**	-7.45	5.29[4]
0.003	14.53	5.76	5.46	4.25	6.78	-3.13	**7.27**[1]
0.005	12.31	5.90	5.56	4.79	7.13	-1.25	**7.44**[1]
0.007	10.85	6.01	5.62	5.14	7.19	0.02	**7.40**[1]
0.009	9.75	6.06	5.66	5.40	7.18	0.94	**7.32**[1]

Fonte: Do Autor.

Tabela 3.4 – Valores de $\overline{SNR}_{dB,final}$ comparando com outros algoritmos de truncamento para o sinal **Heavsine**

σ^2_{ruido}	$SNR_{dB,ruidoso}$	VISU	SURE	BAYES	SPC ($p = 3$)	CUSUM	CUSUM$_{adap.}$
0.001	19.45	6.18	5.63	6.79	**6.89**	5.14	6.72[3]
0.003	14.68	6.62	5.92	8.28	8.86	**9.12**	6.92[4]
0.005	12.46	6.87	6.04	9.01	9.72	**10.68**	6.98[4]
0.007	11.00	7.03	6.11	9.55	10.23	**11.57**	7.01[5]
0.009	9.91	7.14	6.16	9.94	10.57	**12.16**	7.03[5]

Fonte: Do Autor.

O método do CUSUM adaptado foi o que obteve os melhores resultados na maioria dos cenários. Nos que foi melhor a variância σ^2_{ruido} do ruido era igual ou acima de 0.005 e nos cenários em que não foi o melhor, ainda assim, ficou bem colocado. Com variâncias mais baixas os métodos mais conservadores (*VisuShrink*) ou baseado no gráfico de controle de Shewhart (*SPC-Threshold*) se mostraram melhores, nestas variâncias o $\overline{SNR}_{dB,final}$ do CUSUM adaptado em alguns momentos se mostrou negativo, resultado de uma extração mais agressiva do ruído prejudicando parte que era do sinal ideal, porém valores abaixo de uma unidade de $SNR_{dB,final}$.

Os outros métodos baseados em gráficos de controle também fizeram esta extração mais agressiva do ruído nas variâncias mais baixas e o CUSUM tradicional, em especial, os fez na maioria dos cenários com valores variando de uma a até 10 unidades negativas de $SNR_{dB,final}$.

Para o sinal de teste *Heavsine* o CUSUM com valores tradicionais se mostrou melhor que o adaptado. Apesar do método adaptado estar com piores resultados, ao ser comparado com outros métodos, seus índices de qualidade ainda são bons e próximos aos demais.

Analisando a evolução dos desvios-padrão (Figura 2.8) podemos notar que o sinal *Heavsine* possui uma suavidade dos desvios em função do nível j da transformada wavelet, o que tornou o modelo menos adaptado para esse sinal de teste. O sinal *Bump* mesmo apresentando uma disparidade do nível mais fino para o segundo mais fino respondeu bem ao CUSUM que foi proposto.

4 CONSIDERAÇÕES FINAIS

A proposta deste trabalho foi a adaptação do gráfico de controle CUSUM para a filtragem de coeficientes wavelet para sinais unidimensionais. Neste processo, foi feita a modelagem de sinais sintéticos ideais e de várias versões de ruído que somadas resultam em sinais com informações comprometidas. Nesses sinais ruidosos foram realizados testes de filtragem do CUSUM adaptado em comparação aos métodos de filtragem wavelet consagrados.

Para a decomposição do sinal e obtenção dos valores de qualidade do método proposto, a transformada wavelet discreta foi aplicada nos sinais com ruído e decompostos em cinco níveis. Cada nível j dos coeficientes wavelet foi filtrado utilizando várias configurações do intervalo de decisão do gráfico CUSUM (Equação (2.23)) via simulação de Monte Carlo. A avaliação da qualidade de filtragem foi dada pelo SNR_{db}, figura de mérito deste trabalho (Seção 2.3).

A metodologia que foi montada para adaptar o gráfico CUSUM foi baseado na análise de Chen e Han (2005). Na Seção 2.2.2.1 desta tese, foi verificado que o desvio-padrão dos coeficientes wavelet das imagens possuem um decaimento logarítmico (Figura 2.7) e constatado evolução semelhante para os sinais de teste da Figura 2.4 (Figura 2.8).

A partir dos Resultados (Seção 3), foi mostrado que o gráfico CUSUM para a filtragem via wavelet carece de adaptações por não se adequar de forma geral aos coeficientes. O gráfico apresenta ter um desempenho superior quando os intervalos de decisão são preparados para truncar os coeficientes wavelet, ou seja, que o CUSUM quando adaptado funciona de forma mais eficiente do que o CUSUM tradicional (Equações (2.25) e (2.26)) para a maioria dos cenários.

Os resultados também mostraram que o CUSUM tem competitividade, em índice de SNR_{db}, perante aos algoritmos consagrados como o *VisuShrink* e o *SureShrink* de Donoho e Jonhstone. Percebeu-se, ainda, que os gráficos de controle (considerando também o método *SPC-Threshold*) se mostram bastante qualificados, pela avaliação medida em SNR_{db}, para filtragem via wavelet.

O aprimoramento do CUSUM adaptado pode ainda ser feito por outros ajustes nos intervalos de decisão do gráfico CUSUM. Outro parâmetro que pode impactar no acúmulo dos limites de controle do gráfico é o valor de K, já apresentado na Seção 2.2. Neste trabalho foi utilizado um valor fixo recomendado por Montgomery (2009), entretanto esse valor pode ser modificado de forma a aperfeiçoar os alertas do gráfico de controle, deixando-o mais ou menos sensível aos coeficientes que analisa.

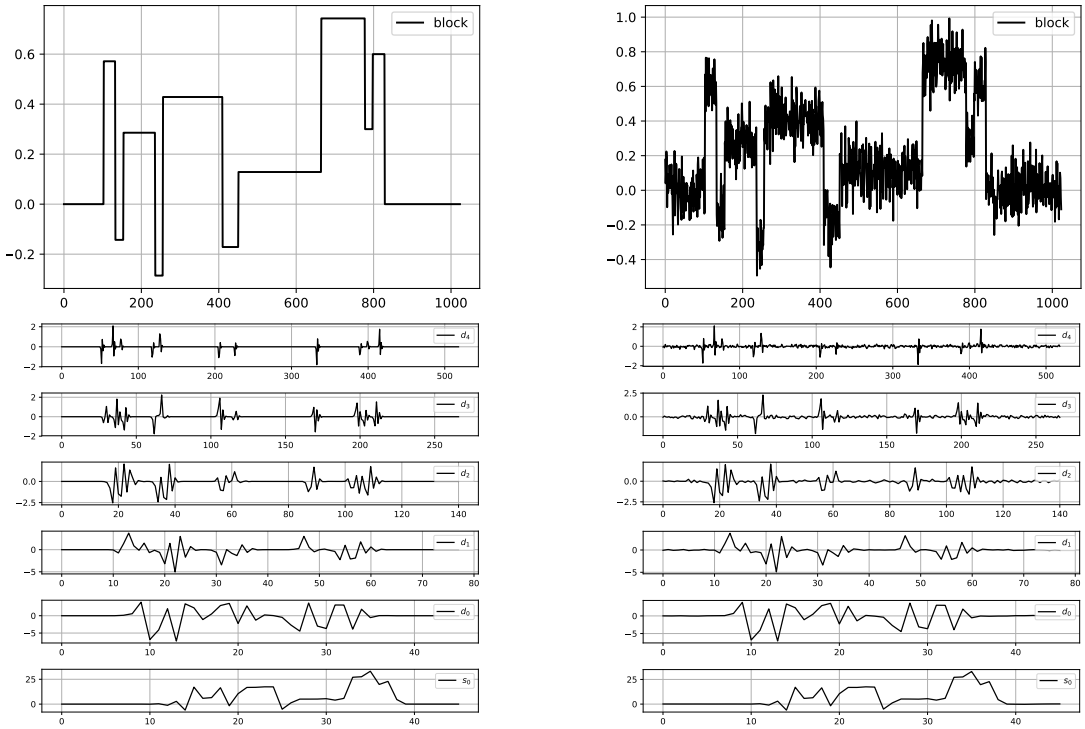
Um trabalho futuro, já em planejamento, é a disponibilização dos algoritmos utilizados neste trabalho para toda a comunidade acadêmica, em especial a de filtragem estatística de sinais. A implementação dos algoritmos consagrados de filtragem estatística, dos sinais de teste e da métrica SNR_{db}, utilizada nesta tese, para avaliação de qualidade da filtragem são inexistentes nos repositórios da linguagem Python. E, além da biblioteca com estas funções implementadas, pode-se ainda oferecer um ambiente em que um desenvolvedor ou pesquisador possa inserir seus sinais de teste ou usar os sinais disponíveis na biblioteca, realizar sua transformada wavelet discreta, testar seus algoritmos de filtragem, comparar com os métodos consagrados e, ainda, aferir seu algoritmo com um índice de qualidade, como foi realizado neste trabalho.

APÊNDICE A – COEFICIENTES WAVELET E ESCALA DOS SINAIS DE TESTE

Este Apêndice apresenta a decomposição dos sinais de teste da Figura 2.4 com e sem ruído. Também incluso, para cada figura, uma tabela com informações de diferenças entre os coeficientes wavelet com ruído e sem ruído como valores mínimo e máximo, variância e amplitude máxima da diferença. No fim, a Tabela A.5 mostra a variância dos sinais de teste ideais.

Figura A.1 – Sinal: *Block* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0}

(a) Ideal. Sem ruído (b) Com ruído ($\sigma^2_{ruido} = 0.01$, $SNR_{dB} = 8.751$)



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

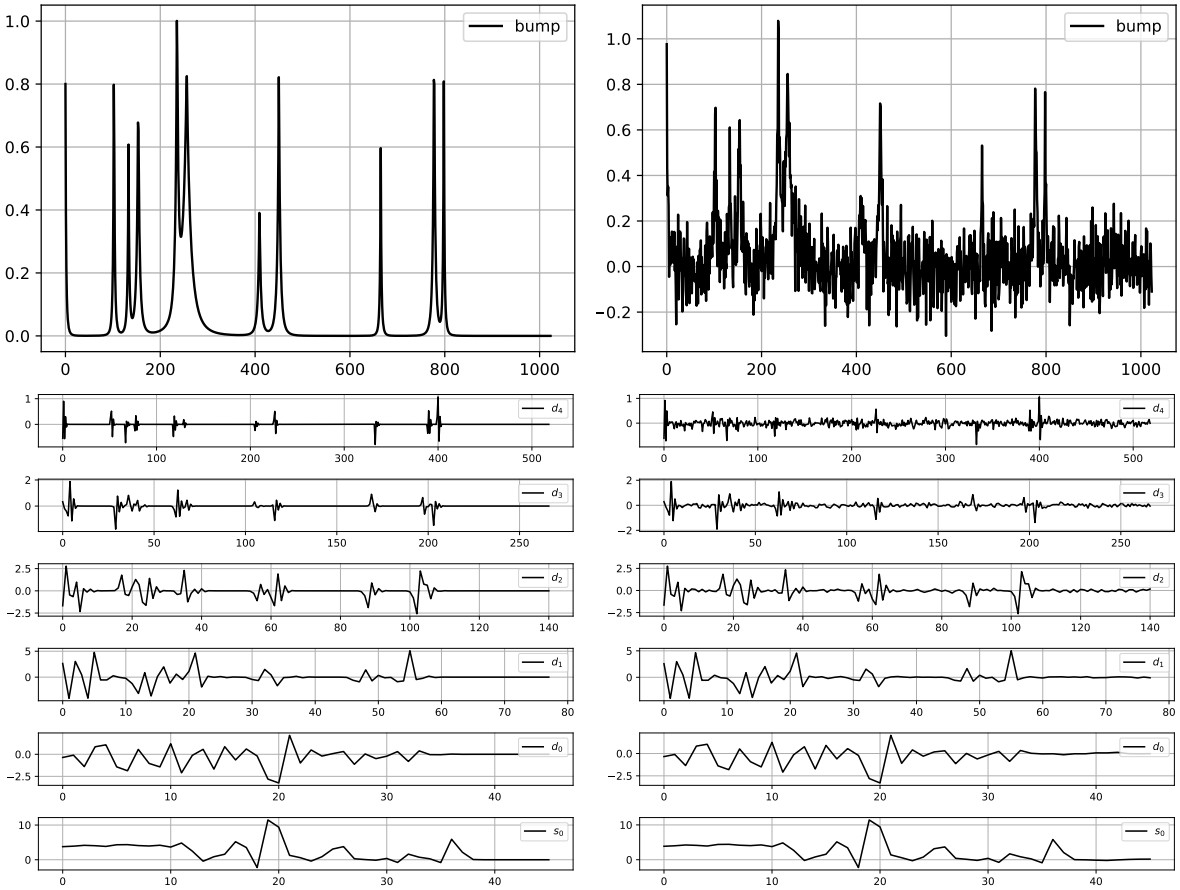
Tabela A.1 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Block** ($\sigma^2_{ideal} = 0.0748$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0836$)

	mínimo	máximo	σ^2_j	amplitude
s_0	-0.016	-0.001	0.015	0.016
d_0	-0.031	0.078	0.009	0.108
d_1	-0.005	0.030	0.010	0.036
d_2	-0.082	0.096	0.010	0.178
d_3	-0.097	0.071	0.010	0.166
d_4	-0.093	0.065	0.010	0.158

Fonte: Do Autor.

Figura A.2 – Sinal: *Bump* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0}

(a) Ideal. Sem ruído (b) Com ruído ($\sigma^2_{ruido} = 0.01$, $SNR_{dB} = 2.788$)



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

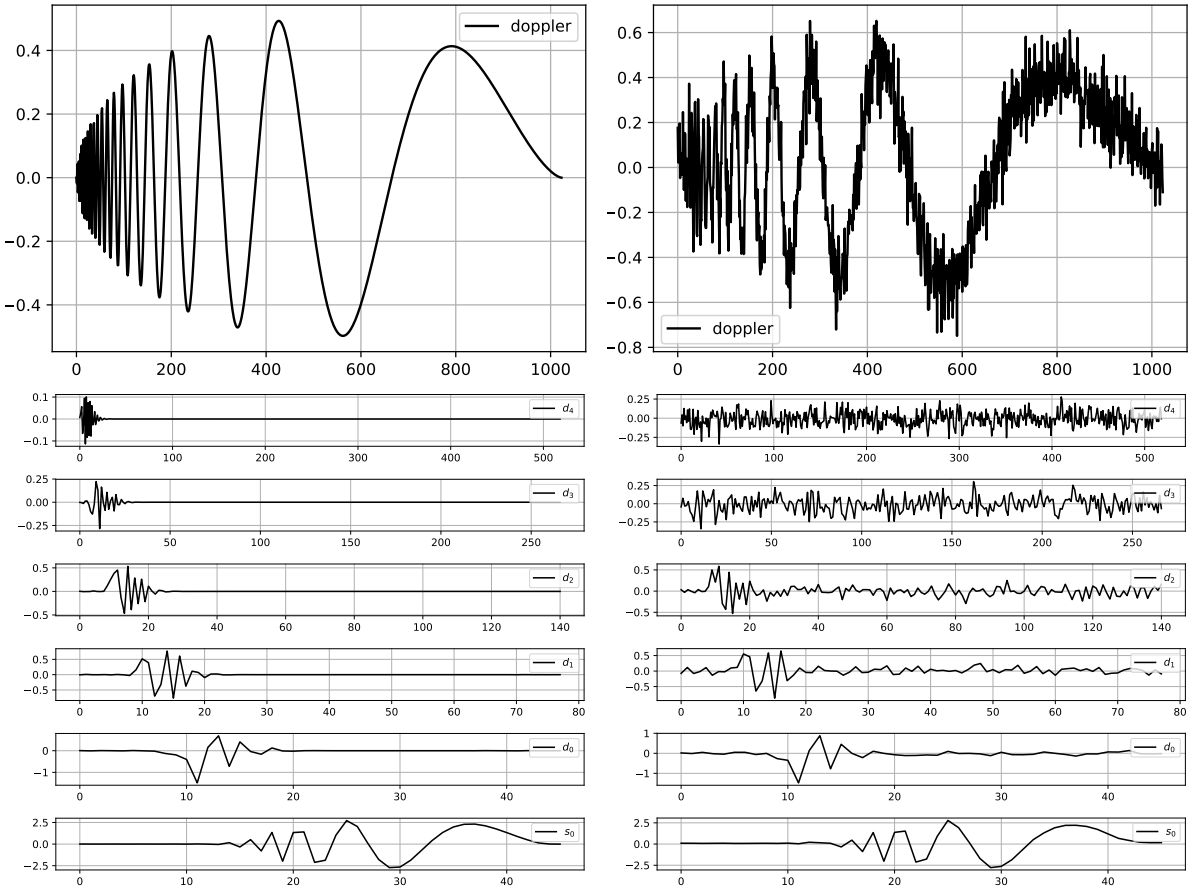
Tabela A.2 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Bump** ($\sigma^2_{ideal} = 0.0191$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0278$)

	mínimo	máximo	σ^2_j	amplitude
s_0	-0.024	0.001	0.016	0.026
d_0	-0.024	0.010	0.008	0.035
d_1	-0.043	0.060	0.010	0.103
d_2	-0.053	0.045	0.010	0.097
d_3	-0.052	0.033	0.010	0.085
d_4	-0.161	0.115	0.010	0.276

Fonte: Do Autor.

Figura A.3 – Sinal: *Doppler* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0}

(a) Ideal. Sem ruído (b) Com ruído ($\sigma^2_{ruido} = 0.01$, $SNR_{dB} = 9.191$)



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

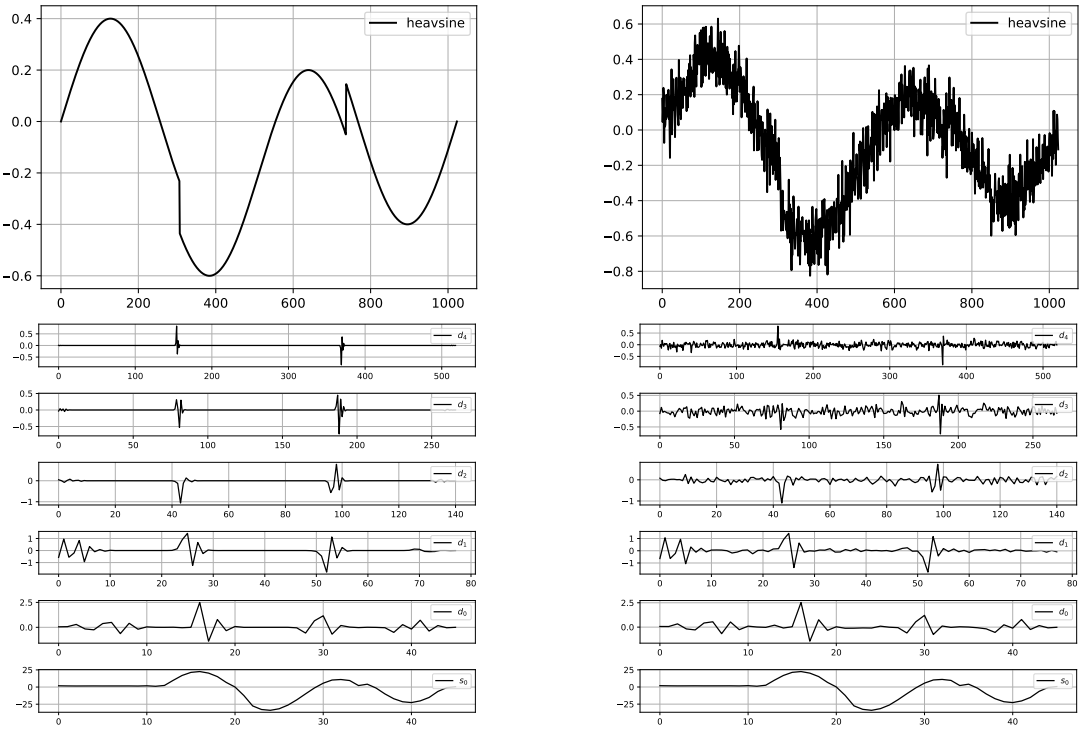
Tabela A.3 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Doppler** ($\sigma^2_{ideal} = 0.0834$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0970$)

	mínimo	máximo	σ^2_j	amplitude
s_0	-0.031	0.000	0.016	0.031
d_0	-0.001	0.000	0.008	0.003
d_1	-0.030	0.010	0.010	0.040
d_2	-0.028	0.032	0.010	0.060
d_3	-0.046	0.090	0.010	0.136
d_4	-0.196	0.207	0.010	0.403

Fonte: Do Autor.

Figura A.4 – Sinal: *Heavsine* e a sua decomposição em cinco níveis, incluindo os coeficientes de escala no nível mais grosseiro: s_{J_0}

(a) Ideal. Sem ruído (b) Com ruído ($\sigma^2_{ruido} = 0.01$, $SNR_{dB} = 9.445$)



Fonte: Do Autor. Utilizando *Python* e as bibliotecas *PyWavelets*, *Numpy* e *Matplotlib*.

Tabela A.4 – Informações da diferença entre os coeficientes wavelet com e sem ruído do sinal **Heavsine** ($\sigma^2_{ideal} = 0.0882$, $\sigma^2_{ruido} = 0.0097$, $\sigma^2_{ideal+ruido} = 0.0989$)

	mínimo	máximo	σ^2_j	amplitude
s_0	-0.017	0.016	0.015	0.032
d_0	-0.095	0.035	0.008	0.131
d_1	-0.099	0.127	0.010	0.225
d_2	-0.166	0.190	0.010	0.355
d_3	-0.213	0.240	0.010	0.454
d_4	-0.223	0.223	0.010	0.445

Fonte: Do Autor.

Tabela A.5 – Variância dos sinais de teste ideais

Sinal	σ^2
block	0.075
bump	0.019
doppler	0.083
heavisine	0.088

Fonte: Do Autor.

APÊNDICE B – ALGORITMOS QUE MODELAM OS SINAIS DE TESTE

Este Apêndice apresenta uma descrição em pseudo-código dos algoritmos utilizados para modelar os sinais de teste da Figura 2.4.

Algoritmo 4: FUNÇÃO MATEMÁTICA PARA O SINAL DOPPLER ($dopplerFunction(\cdot)$)

Entrada: x: Coordenada das abcissas.
Saída: y: Coordenada das ordenadas.

início
  // Para valores de $0 \leq x \leq 1$
  $e = 0.05$
  $y = \sqrt{x(1-x)}\sin\left(\frac{2\pi(1+e)}{x+e}\right)$
fim
retorna y

Algoritmo 5: FUNÇÃO MATEMÁTICA PARA O SINAL BLOCK ($blockFunction(\cdot)$)

Entrada: x: Coordenada das abcissas.
Saída: y: Coordenada das ordenadas.

início
  // Para valores de $0 \leq x \leq 1$
  $h = [0,\ 4,\ -5,\ 3,\ -4,\ 5,\ -4.2,\ 2.1,\ 4.3,\ -3.1,\ 2.1,\ -4.2]$
  $t = [0,\ 0.1,\ 0.13,\ 0.15,\ 0.23,\ 0.25,\ 0.40,\ 0.44,\ 0.65,\ 0.76,\ 0.78,\ 0.81]$
  $K(t) = (1 + sign(t))/2$
  para cada h_i *e* t_i ***em*** h *e* t **faça**
    $y = \sum h_i K(x - t_i)$
  fim
fim
retorna y

Algoritmo 6: FUNÇÃO MATEMÁTICA PARA O SINAL BUMP ($bumpFunction(\cdot)$)

Entrada: x: Coordenada das abcissas.
Saída: y: Coordenada das ordenadas.

início
  `// Cuidar os limites de representação da função Bump, ela estoura infinitamente no eixo y.`
  `// Para valores de` $0 \leq x \leq 1$
  $h = [4, 5, 3, 4, 5, 4.2, 2.1, 4.3, 3.1, 5.1, 4.2]$
  $w = [0.005, 0.005, 0.006, 0.01, 0.01, 0.03, 0.01, 0.01, 0.005, 0.008, 0.005]$
  $t = [0, 0.1, 0.13, 0.15, 0.23, 0.25, 0.40, 0.44, 0.65, 0.76, 0.78, 0.81]$
  $K(t) = (1 + abs(t))^{-4}$
  para cada h_i, t_i *e* w_i ***em*** h, t *e* w **faça**
    $y = \sum h_i K((x - t_i)/w_i)$
  fim
fim
retorna y

Algoritmo 7: FUNÇÃO MATEMÁTICA PARA O SINAL HEAVSINE ($heavsineFunction(\cdot)$)

Entrada: x: Coordenada das abcissas.
Saída: y: Coordenada das ordenadas.

início
  `// Para valores de` $0 \leq x \leq 1$
  $y = 4 * sin(4\pi x) - sign(x - 0.3) - sign(0.72 - x)$
fim
retorna y

Algoritmo 8: FUNÇÃO SIGN

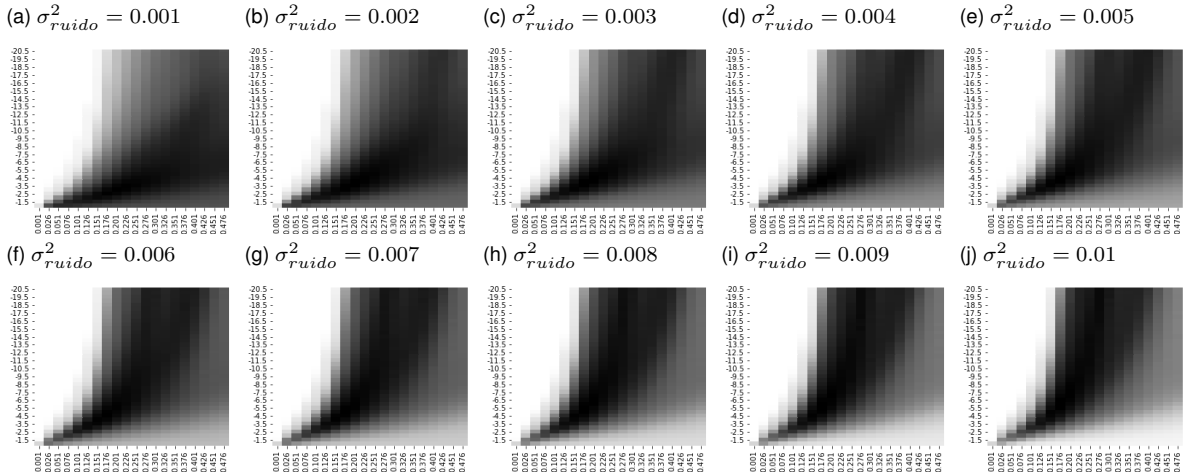
Entrada: x: Valor real.
Saída: y: -1, 0 ou 1.

início
  if $x < 0$ **then**
    $y = -1$
  else
    if $x == 0$ **then**
      $y = 0$
    else
      $y = 1$
    end
  end
fim
retorna y

APÊNDICE C – MAPAS DE CALOR PARA CADA CENÁRIO TESTADO

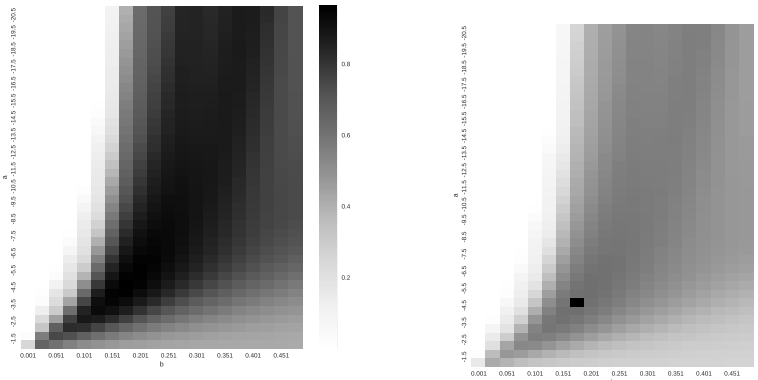
Este Apêndice apresenta todos os resultados obtidos na simulação pelo método de Monte Carlo do Algoritmo 3. Cada mapa de calor pertence a um cenário (Seção 2.4) em específico. Abaixo de cada aglomerado destes mapas está uma mapa médio, do sinal que foi analisado pelas 10 variâncias testadas.

Figura C.1 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Block** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados



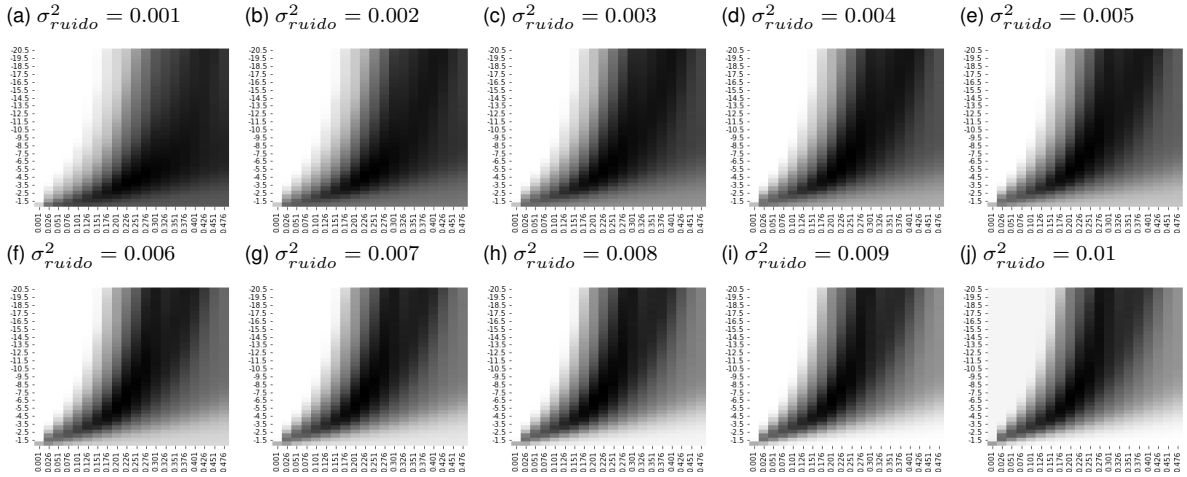
Fonte: Do Autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.2 – Mapa de calor médio de todos os mapas de calor da Figura C.1



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.3 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Bump** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados

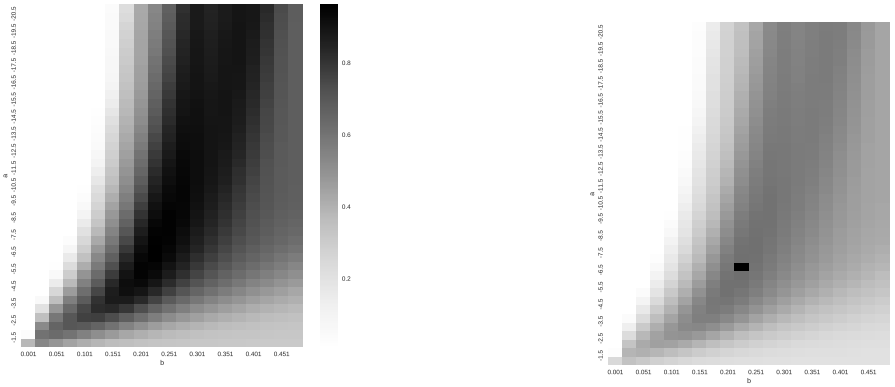


Fonte: Do Autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.4 – Mapa de calor médio de todos os mapas de calor da Figura C.3

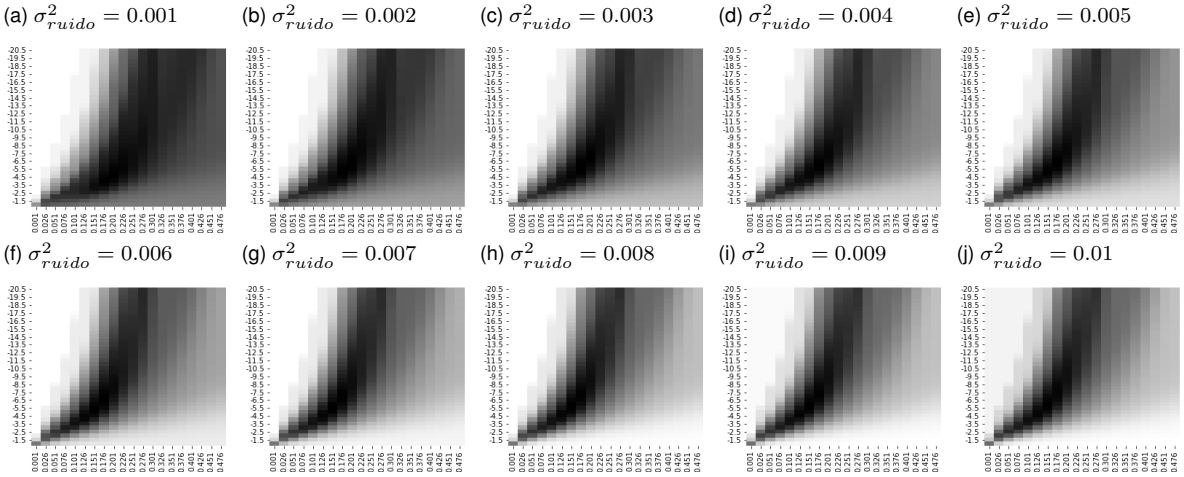
(a) Gráfico Original

(b) Gráfico com valor máximo evidenciado. $a = -6.5$ e $b = .226$



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.5 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Doppler** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados

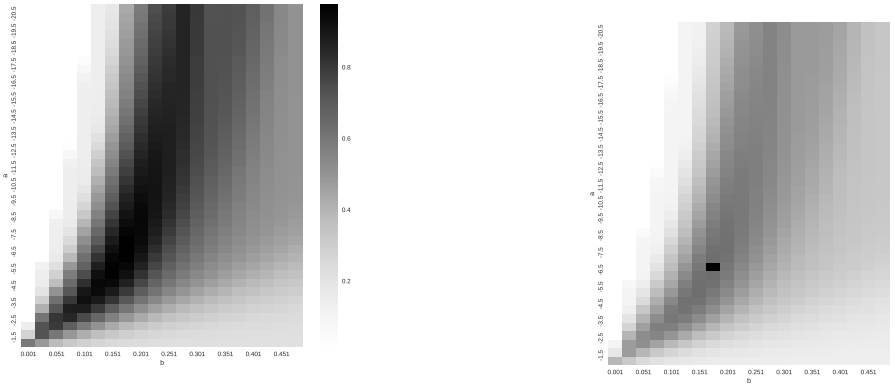


Fonte: Do Autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.6 – Mapa de calor médio de todos os mapas de calor da Figura C.5

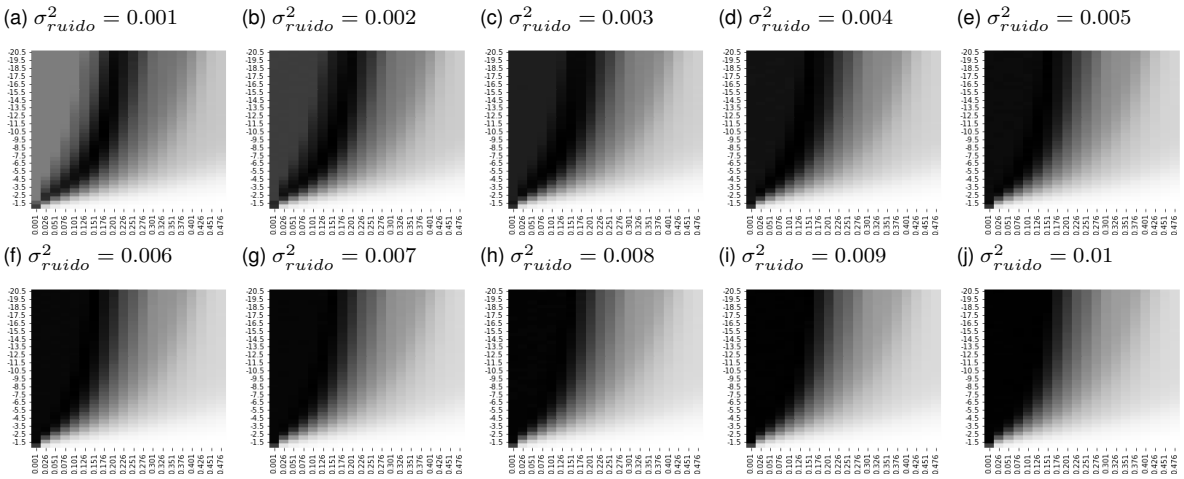
(a) Gráfico Original

(b) Gráfico com valor máximo evidenciado. $a = -6.5$ e $b = .176$



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.7 – Gráficos que relacionam os parâmetro a e b (Equação (2.27)) à qualidade da filtragem para o sinal de teste **Heavsine** para 10 variâncias diferentes, valores de $\overline{SNR}_{db,final}$ (Algoritmo 3) normalizados

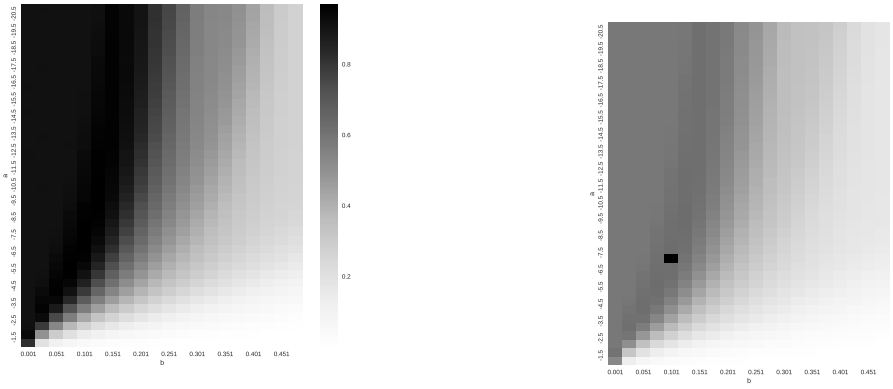


Fonte: Do Autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

Figura C.8 – Mapa de calor médio de todos os mapas de calor da Figura C.7

(a) Gráfico Original

(b) Gráfico com valor máximo evidenciado. $a = -7$ e $b = .101$



Fonte: Do autor. Utilizando *Python* e as bibliotecas *Pandas* e *Seaborn*.

APÊNDICE D – ALGORITMOS DOS MÉTODOS DE TRUNCAMENTO CONSAGRADOS

Este Apêndice apresenta o algoritmo em pseudo-código dos métodos consagrados utilizados para comparar o modelo do CUSUM adaptado neste trabalho.

Algoritmo 9: VISUSHRINK

Entrada: d_{J-1}: Vetor de coeficientes wavelet do nível mais fino.
Saída: λ_{global}: Limiar de corte para os coeficientes wavelet.

início
  $N = size(d_{J-1})$
  $\hat{\sigma} = \frac{median(|d_{J-1}|)}{0.6745}$
  $\lambda_{global} = \hat{\sigma} * \sqrt{2 \log N}$
fim
retorna λ_{global}

Algoritmo 10: SURE

Entrada: d_j: Vetor de coeficiente wavelet.
t: Parâmetro setado pelo SureShrink.
Saída: $SURE_{value}$: Utilizado para o λ do SureShrink.

início
  $N = size(d_j)$
  $acc_1 = acc_2 = 0$
  para $d_{j,k}$ ***em*** d_j **faça**
    if $d_{j,k} \leq t$ **then**
      $acc_1 = acc_1 + 1$
    end
    $acc_2 = acc_2 + min(d_{j,k}, t)^2$
  fim
  $SURE_{value} = N - 2 * acc_1 + acc_2$
fim
retorna $SURE_{value}$

Algoritmo 11: SURESHRINK

Entrada: d_j: Vetor de coeficiente wavelet.
Saída: λ_j: Limiar de corte para os coeficientes wavelet do nível j.

início
  $N_j = size(d_j)$
  $\hat{\sigma} = \frac{median(|d_{J-1}|)}{0.6745}$
  $t = 0, ..., \hat{\sigma} * \sqrt{2 \log N_j}$
  $SURE_{values} = SURE(d_j, t)$
  $\lambda_j = t[argmin(SURE_{values})]$
fim
retorna λ_j

Algoritmo 12: BAYESSHRINK

Entrada: d_{J-1}: Vetor de coeficientes wavelet do nível mais fino.
d_j: Vetor de coeficientes wavelet do nível j.
Saída: λ_j: Limiar de corte para os coeficientes wavelet do nivel j.

início
  $\hat{\sigma} = \frac{median(|d_{J-1}|)}{0.6745}$
  $N_j = size(d_j)$
  $\hat{\sigma}^2_{d_j} = \sum_k \frac{d^2_{j,k}}{N_j}$
  $\sigma_x = \sqrt{max(\hat{\sigma}^2_{d_j} - \hat{\sigma}^2, 0)}$
  $\lambda_j = \frac{\hat{\sigma}^2}{\sigma_x}$
fim
retorna λ_j

Algoritmo 13: SPC-THRESHOLD

Entrada: d_j: Vetor de coeficientes wavelet do nível j.
p: Parâmetro interno do algoritmo, geralmente valores entre 2 e 3.
Saída: λ_j: Limiar de corte para os coeficientes wavelet do nivel j.

início
  $flag = False$
  $N_j = size(d_j)$
  $d_{j.tmp} = d_j$
  repita
    $N_j = size(d_{j.tmp})$
    $S_j = \sqrt{\frac{1}{N_j - 1} \sum (d_{j.tmp} - |d_{j.tmp}|)^2}$
    $flag = True$
    para $d_{j,k}$ ***em*** $d_{j.tmp}$ **faça**
      if $|d_{j,k}| \geq p * S_j$ **then**
        $flag = False$
        $d_{j.tmp} = remove(d_{j.tmp}, d_{j,k})$
      end
    fim
  até $!flag\ and\ N_j > 0$;
  $\lambda_j = p * S_j$
fim
retorna λ_j

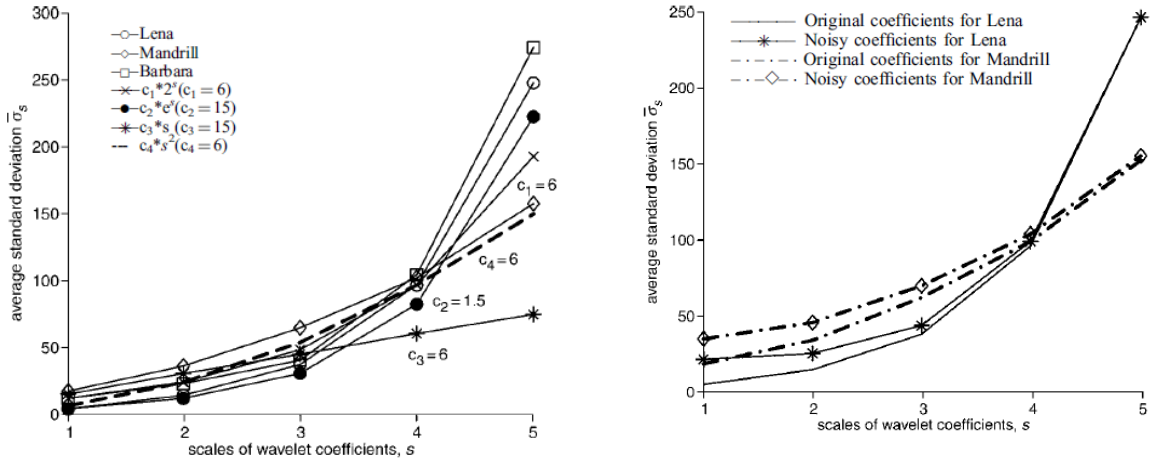
ANEXO A – GRÁFICOS ORIGINAIS DE CHEN PARA A EVOLUÇÃO DOS DESVIOS-PADRÃO DOS VETORES DE COEFICIENTES WAVELET EM CADA NÍVEL

Este Anexo apresenta os resultados do artigo original de Chen. Consideramos até então que quanto menor o nível mais grosseiro é o vetor de coeficientes, porém a notação que o autor usa é diferente da utilizada até o momento. Nos gráficos da Figura A.1 o vetor de coeficientes wavelet com $s = 0$, em que s é o nível wavelet dos coeficientes, é o vetor de coeficientes do nível mais fino $J - 1$.

Figura A.1 – Gráficos que relacionam o nível wavelet dos coeficientes e o desvio-padrão médio no nível

(a) Desvio-padrão médio dos coeficientes wavelet pelo nível wavelet das imagens de teste em comparação com algumas funções

(b) Desvio-padrão médio dos coeficientes wavelet pelo nível wavelet das imagens de teste com e sem ruído



Fonte: Adaptive wavelet threshold for image denoising (CHEN; HAN, 2005)

A parte (a) da Figura A.1, mostra o desvio padrão médio dos coeficientes wavelet de três imagens de teste (Figura 2.6) em comparação com algumas funções (6×2^s, $15 \times e^s$, $15 \times s$ e $6 \times s^2$) e esta descrito em seu texto que os coeficientes wavelet das 3 imagens evoluem de forma semelhante a equação 6×2^s levando em conta a ideia dos quadrados mínimos. Na parte (b) da imagem ele utiliza algumas imagens usadas na parte (a), mas comparando-as com e sem ruído.

REFERÊNCIAS BIBLIOGRÁFICAS

ALEXANDER, C. K.; SADIKU, M. N. O. **Fundamental of Eletric Circuits**. Fourth edition. United States: Mc Graw Hill, 2009.

BAYER, F. M.; KOZAKEVICIUS, A. J. SPC-threshold: uma proposta de limiarização para filtragem adaptativa de sinais. **Tendências em Matemática Aplicada e Computacional**, v. 11, n. 2, p. 121–132, 2010. In portuguese.

BENDAT, J.; PIERSOL, A. G. **Engineering Applications of Correlation and Spectral Analysis**. Second edition. United States: J. Willey, 1993.

CHAGANTI, V. R. **Edge Detection of Noisy Images using 2-D Discrete Wavelet Transform**. 2005. Tese (Doutorado) — THE FLORIDA STATE UNIVERSITY, 2005.

CHANG, S. G.; YU, B.; VETTERLI, M. Adaptive wavelet thresholding for image denoising and compression. **IEEE Transactions on Image Processing**, v. 9, p. 1532–1546, 2000.

CHEN, Y.; HAN, C. Adaptive wavelet threshold for image denoising. **Electronics Letters**, v. 41, 2005.

DESAINTE-CATHERINE, M.; MARCHAND, S. High-precision Fourier analysis of sounds using signal derivatives. **Journal of the Audio Engineering Society**, v. 48, n. 7/8, p. 654–667, 2000.

DONOHO, D. L.; JOHNSTONE, I. M. Ideal spatial adaptation via wavelet shrinkage. **Biometrika**, v. 81, p. 425–455, 1994.

DONOHO, D. L.; JOHNSTONE, I. M. Adapting to unknown smoothness via wavelet shrinkage. **Journal of the American Statistical Association**, v. 90, n. 432, p. 1200–1224, 1995.

ESCOFET, J.; MILLAN, M. S.; RALLO, M. Modeling of woven fabric structures based on Fourier image analysis. **Applied Optics**, OSA, v. 40, n. 34, p. 6170–6176, Dec 2001.

GROSS, J. **Linear Regression**. First edition. Deutschland: Springer, 2003.

JAY, C. Manchester encoder circuit for local area networks. **Microprocessors and Microsystems**, v. 13, n. 1, p. 28–32, 1989.

KOZAKEVICIUS, A. et al. Adaptive ECG filtering and QRS detection using orthogonal wavelet transform. In: **Proceedings of International Conference on BioMedical Engineering (IASTED)**. Austria: Innsbruck, 2005.

KOZAKEVICIUS, A. D. J.; BAYER, F. M. Filtragem de sinais via limiarização de coeficientes wavelet. **Ciência e Natura**, v. 36, p. 37–51, 2014.

MALLAT, S. A theory for multiresolution signal decomposition: the wavelet representation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 11, n. 7, p. 674–693, 1989.

MOCCAGATTA, I.; COBAN, M. Z.; CHEN, H. H. Wavelet-based image coding: comparison of MPEG-4 and JPEG-2000. **Conference Record of the Thirty-Third Asilomar Conference on Signals, Systems, and Computers**, 1999.

MONTGOMERY, D. C. **Introduction to Statistical Quality Control**. Sixth edition. United States: John Wiley & Sons, Inc., 2009. 733 p.

NIELSEN, O. **Wavelets in Scientific Computing**. 1998. Tese (Doutorado) — Technical University of Denmark, 1998.

OPPENHEIM, A. V.; SCHAFER, R. W. **Discrete-Time Signal Processing**. Thirth edition. United States: Prentice Hall Press, 2009.

RAZAVI, B. **Fundamentals of Microelectronics**. First edition. United States: John Wiley & Sons, 2008.

SHEWHART, W. A. **Economic Control of Quality of Manufactured Product**. New York: MacMillan, 1931.

STEIN, C. Estimation of the mean of a multivariate normal distribution. **Annals of Statistics**, v. 9, p. 1135–1151, 1981.

TANG, X.; QU, C. Facial image recognition based on fractal image encoding. **Bell Labs Technical Journal**, v. 15, p. 209–214, 2010.

WELVAERT, M.; ROSSEEL, Y. On the definition of signal-to-noise ratio and contrast-to-noise ratio for fMRI data. **PLoS ONE**, v. 8, n. 11, p. 1–10, 2013.