

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PLATENY DE BRITO PONCHET

AVALIAÇÃO DE TÉCNICAS DE COMPRESSÃO DE DADOS COM PERDAS DE SÉRIES TEMPORAIS EM APLICAÇÕES DE MONITORAMENTO DE ENERGIA ELÉTRICA

CAMPINA GRANDE - PB
2026

PLATENY DE BRITO PONCHET

AVALIAÇÃO DE TÉCNICAS DE COMPRESSÃO DE DADOS COM PERDAS DE SÉRIES TEMPORAIS EM APLICAÇÕES DE MONITORAMENTO DE ENERGIA ELÉTRICA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande, pertencente à linha de pesquisa Sistemas de Computação e área de concentração Ciência da Computação, como requisito para a obtenção do Título de Mestre em Ciência da Computação.

Orientador(a): Prof. Dr. Andrey Elísio Monteiro Brito

CAMPINA GRANDE - PB

2026

Universidade Federal de Campina Grande - UFCG
Sistema de Bibliotecas - SISTEMOTECA
Catalogação de Publicação na Fonte. UFCG - Biblioteca Central

P795a Ponchet, Plateny de Brito.

Avaliação de técnicas de compressão de dados com perdas de séries temporais em aplicações de monitoramento de energia elétrica / Plateny de Brito Ponchet. – 2026.

160 f. : il. color.

Dissertação (mestrado em Ciência da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2026.

"Orientação: Prof. Dr. Andrey Elísio Monteiro Brito".

Referências.

1. Compressão com Perda. 2. Séries Temporais Elétricas. 3. Avaliação Comparativa. 4. Controle Adaptativo. 5. LoRaWAN. 6. ARC-SDT. 7. Dispositivos Embarcados. I. Brito, Andrey Elísio Monteiro. II. Título.

UFCG/BC CDU 004(043.3)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA ITAPUANA SOARES DIAS GONÇALVES CRB-15/93



MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
POS-GRADUACAO EM CIENCIA DA COMPUTACAO
Rua Aprígio Veloso, 882, Edifício Telmo Silva de Araújo, Bloco CG1, - Bairro Universitário, Campina Grande/PB, CEP 58429-900
Telefone: 2101-1122 - (83) 2101-1123 - (83) 2101-1124
Site: http://computacao.ufcg.edu.br - E-mail: secpg@computacao.ufcg.edu.br

FOLHA DE ASSINATURA PARA TESES E DISSERTAÇÕES

PLATENY DE BRITO PONCHET

AVALIAÇÃO DE TÉCNICAS DE COMPRESSÃO DE DADOS COM PERDAS DE SÉRIES TEMPORAIS EM APLICAÇÕES DE MONITORAMENTO DE ENERGIA ELÉTRICA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação como pré-requisito para obtenção do título de Mestre em Ciência da Computação.

Aprovada em: 26/08/2026

Prof. Dr. **Andrey Elísio Monteiro Brito**, UFCG, Orientador

Profa. Dra. **Lívia Maria Rodrigues Sampaio Campos**, UFCG, Examinadora Interna

Prof. Dr. **Eduardo de Lucena Falcão,** UFRN, Examinador Externo

---



Documento assinado eletronicamente por **ANDREY ELISIO MONTEIRO BRITO**, **PROFESSOR 3 GRAU**, em 30/08/2026, às 14:42, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.



Documento assinado eletronicamente por **Eduardo de Lucena Falcão**, **Usuário Externo**, em 31/08/2026, às 18:17, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.



Documento assinado eletronicamente por **LIVIA MARIA RODRIGUES SAMPAIO CAMPOS**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 01/09/2026, às 09:45, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

A autenticidade deste documento pode ser conferida no site https://sei.ufcg.edu.br/autenticidade, informando o código verificador **6817802** e o código CRC **167CB946**.

Referência: Processo nº 23096.059340/2026-78 SEI nº 6817802

RESUMO

O monitoramento contínuo de grandezas elétricas em ambientes industriais gera volumes de dados que precisam ser transmitidos por redes com severas restrições de largura de banda, como o LoRaWAN (*Long Range Wide Area Network*). Essas redes impõem limites rígidos de carga útil por pacote, muitas vezes inferiores ao volume de uma única leitura bruta do dispositivo de monitoramento, tornando a compressão no próprio dispositivo uma necessidade prática. Diante desse cenário, este trabalho realiza uma avaliação comparativa de cinco abordagens de compressão com perda aplicadas a séries temporais de consumo de energia elétrica: *Swinging Door Trending* (SDT), Ramer-Douglas-Peucker (RDP), Transformada Discreta do Cosseno (DCT), Transformada Wavelet Discreta (DWT), esta última avaliada com três famílias de funções base, e o ARC-SDT (*Adaptive Real-time Controlled* SDT), adaptação do SDT com controle direto sobre a taxa de compressão proposta neste trabalho. Os algoritmos são avaliados sobre 256 séries temporais de instalações públicas e privadas, em nove taxas de compressão alvo e cinco tamanhos de janela, usando métricas de semelhança visual, erro percentual ponto a ponto, energia consumida e preservação de picos de demanda. Os resultados mostram que o RDP é o algoritmo mais robusto nas taxas de compressão agressivas, tanto em fidelidade de forma quanto na preservação de picos, e que a DCT mantém o erro de energia próximo de zero na maioria das configurações, sem que isso constitua uma garantia estrutural nessas mesmas taxas. As variantes ortogonais de Wavelet (*haar*, *sym2*) rivalizam com o RDP nas taxas moderadas, e a *haar* alcança a maior semelhança visual mediana do conjunto avaliado. O SDT reúne propriedades atraentes para uso embarcado, como processamento amostra a amostra e memória constante, mas não controla diretamente a taxa de compressão; o ARC-SDT resolve essa limitação ajustando a tolerância de erro a cada amostra, o que garante minimamente a taxa de compressão exigida.

Palavras-chave: compressão com perda, séries temporais elétricas, avaliação comparativa, controle adaptativo, LoRaWAN, ARC-SDT, dispositivos embarcados.

ABSTRACT

Continuous monitoring of electrical quantities in industrial environments generates data volumes that must be transmitted over networks with severe bandwidth constraints, such as LoRaWAN (*Long Range Wide Area Network*). These networks impose strict payload limits per packet, often smaller than the volume of a single raw reading from the monitoring device, making on-device compression a practical necessity. In this context, this work presents a comparative benchmark of five lossy compression approaches applied to electrical energy time series: *Swinging Door Trending* (SDT), Ramer-Douglas-Peucker (RDP), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), the latter evaluated with three wavelet families, and ARC-SDT (*Adaptive Real-time Controlled* SDT), an adaptation of SDT with direct control over the resulting compression rate, proposed in this work. The algorithms are evaluated on 256 time series from public and private installations, across nine target compression rates and five window sizes, using metrics of visual similarity, point-by-point percentage error, consumed energy, and demand peak preservation. Results show that RDP is the most robust algorithm at aggressive compression rates, both in shape fidelity and in peak preservation, and that DCT keeps energy error close to zero in most configurations, though this does not amount to a structural guarantee at those same rates. The orthogonal Wavelet variants (*haar*, *sym2*) rival RDP at moderate rates, and *haar* reaches the highest median visual similarity of the evaluated set. SDT offers attractive properties for embedded use, such as sample-by-sample processing and constant memory, but does not directly control the resulting compression rate; ARC-SDT addresses this limitation by adjusting the error tolerance at each sample, which minimally guarantees the required compression rate.

Keywords: lossy compression, electrical time series, benchmark, adaptive control, LoRaWAN, ARC-SDT, embedded devices.

Agradecimentos

Aos meus pais, que sempre abriram mão do próprio bem-estar para que eu pudesse estudar. Sei que muitas das escolhas que vocês fizeram ao longo da vida foram pensando em mim, e que isso teve um custo real. Esse trabalho é resultado direto desses sacrifícios, e eu nunca vou esquecer isso.

Ao meu orientador, Prof. Dr. Andrey Elísio Monteiro Brito, pela oportunidade e pela confiança. Agradeço por acreditar no meu potencial, especialmente nos momentos em que eu mesmo duvidei. Ter as portas sempre abertas fez com que eu pudesse trabalhar com mais liberdade e segurança.

Aos meus amigos, pelo suporte de sempre. Vocês tornaram essa jornada mais leve.

Por fim, agradeço à LiteMe, empresa parceira do Laboratório de Sistemas Distribuídos (LSD/UFCG) em projetos de pesquisa e desenvolvimento na área de monitoramento de energia elétrica, que, através de apoio da FAPESQ-PB (Edital nº 22/2024 – SECTIES/FAPESQ/PB Segunda Chamada Desafios Tecnológicos e Inovação – Conectando Startups – Transição Energética) e EMBRAPII (projeto PCEE-2412.0279 – ConectEI), financiou esta pesquisa.

Declaração sobre o Uso de Inteligência Artificial

Esta dissertação foi conduzida e escrita pelo autor, que assume total responsabilidade pelo conteúdo, decisões, experimentos e conclusões. Ferramentas de inteligência artificial generativa foram utilizadas como apoio em algumas etapas específicas, uso este que está sendo registrado em nome da transparência.

Estas ferramentas foram utilizadas em: revisão de escrita; apoio em implementação e revisão dos programas desenvolvidos; e verificação de consistência do texto.

Nenhum resultado experimental apresentado foi produzido por inteligência artificial. Todos os valores, tabelas e figuras decorrem da execução dos algoritmos sobre as coleções de dados utilizadas. Da mesma forma, todas as referências bibliográficas foram localizadas, lidas e verificadas pelo autor.

Sumário

1 Introdução **1**

1.1 Contexto 1

1.2 Perguntas de Pesquisa 6

1.3 Contribuições 6

2 Fundamentação Teórica **7**

2.1 Monitoramento de Grandezas Elétricas 7

2.2 Restrições de Comunicação 9

2.3 Compressão de Dados 16

2.4 Métricas de Avaliação 18

3 Algoritmos de Compressão Avaliados **27**

3.1 Visão Geral 27

3.2 Swinging Door Trending (SDT) 28

3.3 Ramer-Douglas-Peucker (RDP) 32

3.4 Transformada Wavelet Discreta (DWT) 36

3.5 Transformada Discreta do Cosseno (DCT) 41

3.6 Considerações Comparativas 45

4 ARC-SDT: Controle Adaptativo da Taxa de Compressão **47**

4.1 Invariância de Escala do SDT 48

4.2 Caracterização da Magnitude pelo RMS Incremental 49

4.3 Controle da Taxa de Compressão por Controlador PI 51

4.4 Descrição do Algoritmo 63

4.5 Propriedades para Dispositivos Embarcados 66

5 Metodologia de Avaliação **68**

5.1 Visão Geral 68

5.2 Coleções 69

5.3 Ambiente e Implementação 73

5.4 Protocolo Experimental 76

5.5 Métricas de Avaliação 80

5.6 Combinações e Volume Experimental 80

5.7 Estratégia de Análise dos Resultados 80

5.8 Saída e Reprodutibilidade 84

6 Avaliação dos Algoritmos de Compressão **85**

6.1 Seleção das Métricas de Análise 85

6.2 Séries de Alta Compressibilidade 86

6.3 Qualidade de Compressão: Coleção Pública 87

6.4 Qualidade de Compressão: Coleção Privada 92

6.5 Efeito do Tamanho de Janela 95

6.6 Avaliação do ARC-SDT 101

6.7 Síntese dos Resultados 107

7 Conclusão e Trabalhos Futuros **109**

7.1 Conclusão 109

7.2 Trabalhos Futuros 115

Bibliografia **117**

A Avaliação de Algoritmos de Compressão sem Perda **121**

A.1 Configuração Experimental 121

A.2 Resultados dos Algoritmos 122

A.3 Comparação com os Limites do LoRaWAN 123

A.4 Conclusão 124

B Efeito do Tamanho de Janela por Algoritmo e Métrica **127**

C Compressibilidade da Tensão **136**

Lista de Símbolos

Grandezas elétricas

P	—	Potência ativa (W)
Q	—	Potência reativa (var)
S	—	Potência aparente (VA)
FP	—	Fator de potência
V	—	Tensão de fase ou de linha (V)
I	—	Corrente de fase ou de neutro (A)
f	—	Frequência da rede elétrica (Hz)
THD	—	Distorção harmônica total (*total harmonic distortion*)
E_a	—	Energia ativa acumulada (Wh)
E_r	—	Energia reativa acumulada (varh)

Compressão e avaliação

N	—	Número total de amostras da série temporal
N_c	—	Número de elementos retidos após compressão
x_i	—	i-ésima amostra do sinal original
$\hat{x}_i$	—	i-ésima amostra do sinal reconstruído
CR	—	Taxa de compressão: fração do sinal eliminada, em $\%$
MSE	—	Erro quadrático médio (*mean squared error*)
RMSE	—	Raiz do erro quadrático médio (*root mean squared error*)
NRMSE	—	RMSE normalizado pela amplitude do sinal

PRD	—	Diferença quadrática percentual (*percentage root-mean-square difference*)
MAPE	—	Erro percentual absoluto médio (*mean absolute percentage error*)
SNR	—	Relação sinal-ruído (dB)
PSNR	—	Relação sinal-ruído de pico (dB)
SSIM	—	Índice de similaridade estrutural (*structural similarity index*)

SDT e RDP

ε	—	Tolerância de erro vertical do SDT (unidades do sinal)
s^+, s^-	—	Inclinações instantâneas dos pivôs superior e inferior (SDT)
S^+, S^-	—	Máxima e mínima inclinação acumuladas no corredor (SDT)
P^+, P^-	—	Pivôs superior e inferior do corredor de aproximação (SDT)
ξ	—	Distância perpendicular máxima tolerada (RDP)
$d(P, \overline{AB})$	—	Distância perpendicular do ponto P à reta AB (RDP)

DWT

ψ	—	Função wavelet mãe
ϕ	—	Função escala
cA_L	—	Coeficientes de aproximação no nível L
cD_j	—	Coeficientes de detalhe no nível j
L	—	Profundidade de decomposição wavelet
M	—	Número de momentos nulos da wavelet
$h[n]$, $g[n]$	—	Filtros passa-baixa e passa-alta do banco de filtros

DCT

$X[k]$	—	k-ésimo coeficiente da DCT

K	—	Número de coeficientes retidos após truncamento espectral (DCT e DWT)

ARC-SDT

RMS_N	—	Raiz da média quadrática acumulada até a N-ésima amostra
MS_N	—	Média quadrática acumulada até a N-ésima amostra
e_n	—	Erro de controle no passo n: CR alvo $-$ CR medida
$e_{\min}$	—	Piso mínimo absoluto de tolerância do SDT
K_p	—	Ganho proporcional do controlador PI
K_i	—	Ganho integral do controlador PI
K_d	—	Ganho derivativo, avaliado e descartado no ajuste

LoRaWAN

SF	—	Fator de espalhamento (*spreading factor*), de SF7 a SF12
BW	—	Largura de banda (*bandwidth*)
ToA	—	Tempo de ocupação do canal (*time on air*), em segundos
DC	—	Ciclo de trabalho (*duty cycle*)

Lista de Figuras

2.1 Desvio ponto a ponto entre o sinal original x (azul) e o reconstruído $\hat{x}$ (laranja). Cada segmento vermelho representa a diferença $x_i - \hat{x}_i$ em uma amostra, e é a média dos quadrados desses desvios que define o MSE, cuja raiz é o RMSE. Quanto maiores e mais numerosos os desvios, maior o erro global. O sinal reconstruído mostrado corresponde a uma interpolação linear entre *keypoints* (marcados em laranja), como produzida pelos algoritmos geométricos avaliados neste trabalho: os desvios aparecem justamente nas amostras que não foram retidas como *keypoints*. 21

2.2 Por que o erro relativo importa: o mesmo erro absoluto tem significância muito diferente conforme o nível de consumo. Nas duas leituras, a reconstrução erra por 10 W (faixas vermelhas de altura idêntica). Numa leitura de consumo alto (200 W), esse erro representa apenas 5%; numa leitura de consumo baixo (20 W), o mesmo erro representa 50%. Uma métrica de erro absoluto como o RMSE atribui o mesmo peso aos dois casos, enquanto o MAPE, por medir o erro em relação ao valor da leitura, revela que a distorção da leitura baixa é muito mais grave. Essa é a vantagem do MAPE em séries que percorrem faixas de consumo distintas ao longo do dia: ele impede que os erros dos trechos de alta carga mascarem distorções nos trechos de baixa carga. 22

2.3 O que o SSIM mede, ilustrado pela degradação isolada de cada um de seus três componentes a partir do mesmo sinal original. Uma reconstrução fiel atinge SSIM próximo de 1. Deslocar apenas o nível médio (luminância), mantendo forma e amplitude, reduz pouco o índice. Reduzir apenas a amplitude das variações (contraste), mantendo o nível médio, produz queda moderada. Alterar a forma do sinal (estrutura), preservando exatamente a mesma média e a mesma amplitude, faz o SSIM despencar para valores negativos: é a correlação local que mais pesa no índice, razão pela qual o SSIM é sensível à preservação da forma do sinal e não apenas à magnitude do erro. Valores calculados com janela deslizante de 7 amostras, como na implementação adotada. 23

2.4 Fidelidade energética. A energia acumulada é a área sob a curva de potência. Ao cortar os picos de consumo, a reconstrução perde área (energia que faltou, violeta) e, ao passar por cima dos vales, ganha alguma (energia a mais, verde). O EnergyError é a diferença líquida entre as duas: quando a perda nos picos supera o ganho nos vales, sobra um erro de energia positivo. 24

2.5 Fidelidade de picos. Acima: a proeminência mede o quanto cada pico se destaca do contexto local, e um pico só é contado se ela supera o limiar θ_p (o bump raso é descartado). Abaixo: o PeakRecall é a fração dos picos do original recuperados na reconstrução dentro da tolerância de posição; aqui a reconstrução achata o pico central, resultando em PeakRecall de $2/3$. 26

3.1 Janela de abertura do SDT. A cada nova amostra, o algoritmo traça as inclinações da âncora até a faixa de tolerância do ponto, formando um triângulo (a abertura daquele passo); as aberturas anteriores (em cinza) vão se estreitando, e a abertura atual é destacada em azul. Quando a inclinação até a nova amostra cai fora da abertura, o último ponto válido é emitido como *keypoint*. 30

3.2 Recursão do RDP. A cada passo, entre os *keypoints* atuais busca-se o ponto de maior distância perpendicular à reta do segmento (tracejado vermelho); se essa distância supera o limiar ξ, o ponto é mantido e o segmento é dividido. No resultado, os pontos que ficam a menos de ξ de alguma reta mantida são descartados. 34

3.3 Decomposição DWT de um nível e *hard thresholding*. O sinal (1) é decomposto em aproximação cA (2a), que retém a tendência, e detalhe cD (2b), que retém as variações finas e destaca os dois eventos abruptos. O *hard thresholding* (3) mantém os K coeficientes de maior magnitude do vetor $[cA \mid cD]$ (verde) e zera os demais (cinza). A reconstrução (4), a partir apenas dos coeficientes mantidos, preserva a tendência e os eventos significativos, descartando os detalhes pequenos. 39

3.4 *Hard thresholding* por magnitude. No espectro de coeficientes (aqui da DCT), mantêm-se os K de maior magnitude (verde, acima do limiar) e zeram-se os demais (cinza). A seleção é pela magnitude, e não pela posição: um coeficiente de frequência mais alta pode ser mantido se sua magnitude for suficientemente grande. O painel inferior compara o sinal original com a reconstrução usando apenas os K coeficientes mantidos. O mesmo mecanismo se aplica à DWT. . . 43

3.5 Resultado da compressão de 80% sobre série de potência ativa da coleção privada (21/08/2025) pelas quatro abordagens avaliadas. Em cada painel, o original ocupa a faixa superior, em cinza, e o reconstruído a inferior, na cor do algoritmo, ambas na mesma escala; à direita, recorte ampliado de 11h às 12h, a hora de maior erro do dia. A DWT é ilustrada pela `haar`, uma das três famílias avaliadas. 46

4.1 Visão geral do ARC-SDT como ciclo de ajuste. Em azul, a etapa herdada do SDT; em vermelho, as etapas acrescentadas por este trabalho; em cores divididas, a montagem do corredor, cuja geometria vem do SDT e cuja largura passa a ser controlada. 48

4.2 Exemplo de série excluída (esquerda) e mantida (direita) pelo critério de filtragem, ambas reconstruídas pelo SDT puro após a mesma busca binária mirando CR = 80%. A série excluída atinge CR = 99,0% mesmo com a busca tentando reduzi-lo até o alvo (mediana das 18 séries excluídas: 99,2%, variando de 86,5% a 99,9%); ela permanece praticamente nula durante todo o dia, com um único evento isolado por volta das 10h, e um sinal assim é descrito pelo SDT com pouquíssimos pontos qualquer que seja a tolerância. A série mantida atinge CR = 80,3% (mediana das 149 séries mantidas), com variabilidade real e contínua que permite à busca aproximar o CR do alvo com precisão fina. 55

4.3 Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_p ($K_i = 0$, alvo = 80%, 149 séries da Coleção Pública após exclusão das séries de alta compressibilidade). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%. 56
4.4 Distribuição do CR final por valor de K_i com $K_p = 32$ fixo, alvo = 80% e janela de 1.440 pontos. Cada *boxplot* resume 149 séries da Coleção Pública, após exclusão das séries de alta compressibilidade. O eixo y aplica compressão logarítmica centrada no alvo ($\text{sign}(\delta) \cdot \ln(1 + |\delta|)$, com $\delta = \text{CR} - 80\%$), preservando a escala linear na vizinhança do alvo e comprimindo os extremos para melhor visualização. 57
4.5 Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_i ($K_p = 32$, alvo = 80%, 149 séries da Coleção Pública após exclusão das séries de alta compressibilidade). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%. 59
4.6 Distribuição do CR final por valor de K_d com $K_p = 32$ e $K_i = 1$ fixos, alvo = 80% e janela de 1.440 pontos. Cada *boxplot* resume 149 séries da Coleção Pública, após exclusão das séries de alta compressibilidade. Eixo y com compressão logarítmica centrada no alvo (ver Figura 4.4). 60
4.7 Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_d ($K_p = 32$, $K_i = 1$, alvo = 80%, 149 séries da Coleção Pública). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%. 61

6.1 Identificação de séries de alta compressibilidade na Coleção Pública. Esquerda: CR obtido pelo SDT após busca binária mirando 80% por série, destacando séries que ultrapassam 85% (linha tracejada). Direita: distribuição do CR pós-busca por série. 87
6.2 Distribuição do SSIM por algoritmo na Coleção Pública. Esquerda: todas as séries. Direita: excluindo séries de alta compressibilidade. 88
6.3 Mediana das métricas de qualidade em função do CR alvo: Coleção Pública, janela de 1440 pontos (todos os dados). 89

6.4 Mediana das métricas de qualidade em função do CR alvo: Coleção Pública, janela de 1440 pontos (excluindo séries de alta compressibilidade). 90
6.5 Distribuição do SSIM por algoritmo na Coleção Privada. 93
6.6 Mediana das métricas de qualidade em função do CR alvo: Coleção Privada, janela de 1440 pontos. 94
6.7 Efeito do tamanho de janela sobre o SSIM: Coleção Pública. Cada linha é um algoritmo; o eixo y é a mediana sobre todos os CRs alvo, excluindo as séries de alta compressibilidade. 96
6.8 Efeito do tamanho de janela sobre o SSIM: Coleção Privada. 96
6.9 Efeito do tamanho de janela sobre o PeakRecall: Coleção Pública (excluindo as séries de alta compressibilidade). 97
6.10 Efeito do tamanho de janela sobre o PeakRecall: Coleção Privada. 97
6.11 Efeito do tamanho de janela sobre o Erro de Energia: Coleção Pública (excluindo as séries de alta compressibilidade). 98
6.12 Efeito do tamanho de janela sobre o Erro de Energia: Coleção Privada. 98
6.13 Efeito do tamanho de janela sobre o MAPE: Coleção Pública (excluindo as séries de alta compressibilidade). 99
6.14 Efeito do tamanho de janela sobre o MAPE: Coleção Privada. 99
6.15 Distribuição do erro de controle do ARC-SDT (CR real − CR alvo): Coleção Pública, janela de 1440 pontos, excluindo séries de alta compressibilidade. O traço horizontal sobre cada violino marca a mediana. A linha tracejada indica erro zero; as linhas pontilhadas cobrem ±5%. 102
6.16 Distribuição do erro de controle do ARC-SDT (CR real − CR alvo): Coleção Privada, janela de 1440 pontos. 103
6.17 Distribuição de métricas de qualidade e do erro do CR (CR real − CR alvo) para ARC-SDT e SDT calibrado com frações iniciais da janela (10%, 25%, 50%): Coleção Pública, janela de 1.440 pontos, excluindo séries de alta compressibilidade. 105
6.18 Distribuição de métricas de qualidade e do erro do CR (CR real − CR alvo) para ARC-SDT e SDT calibrado com frações iniciais da janela (10%, 25%, 50%): Coleção Privada, janela de 1.440 pontos. 106

7.1 Guia informal de escolha do algoritmo, segundo o requisito prioritário da aplicação. Não é uma projeção matemática das métricas; a posição de cada algoritmo reflete qualitativamente as conclusões deste capítulo. 113

A.1 Razão de compressão total para os algoritmos sem perda com melhor desempenho, por intervalo de agregação τ. A família *brotli* lidera em todas as configurações avaliadas. 122

A.2 Bytes diários após compressão sem perda para a Configuração I (todas as grandezas de monitoramento). A linha vermelha indica o limite diário de um plano LoRaWAN com *DataRate* = 3. Nenhum algoritmo, em nenhum intervalo de agregação avaliado, consegue trazer os dados abaixo do limite. 124

A.3 Bytes diários após compressão sem perda para a Configuração II (grandezas de potência). Nenhum algoritmo consegue ficar abaixo do limite LoRaWAN em nenhum dos intervalos avaliados. 125

A.4 Bytes diários após compressão sem perda para a Configuração III (cinco grandezas mínimas). O *brotli* fica abaixo do limite apenas a partir de $\tau = 360$ minutos, exigindo agregação em janelas de 6 horas ou mais. 126

B.1 Efeito do tamanho de janela sobre o SSIM mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade). 128

B.2 Efeito do tamanho de janela sobre o SSIM mediano por algoritmo: Coleção Privada. 129

B.3 Efeito do tamanho de janela sobre o PeakRecall mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade). 130

B.4 Efeito do tamanho de janela sobre o PeakRecall mediano por algoritmo: Coleção Privada. 131

B.5 Efeito do tamanho de janela sobre o Erro de Energia mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade). 132

B.6 Efeito do tamanho de janela sobre o Erro de Energia mediano por algoritmo: Coleção Privada. 133

B.7 Efeito do tamanho de janela sobre o MAPE mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade). 134

B.8 Efeito do tamanho de janela sobre o MAPE mediano por algoritmo: Coleção Privada. 135

C.1 Taxa de compressão do SDT aplicado à tensão, com tolerância de 3 V, sobre 267 janelas diárias das três fases de quinze instalações. A distribuição concentra-se em torno de 99,65%. 137

C.2 Pior caso de compressão do experimento, com razão de 97,22%: a série original de um dia completo (1.440 pontos) e sua reconstrução pelo SDT com quarenta pontos. 138

Lista de Tabelas

2.1 Grandezas elétricas de um dispositivo de monitoramento trifásico típico (30 valores por leitura) 8
2.2 Viabilidade de transmissão por SF no plano AU915 14

3.1 Parâmetros do SDT 32
3.2 Parâmetros do RDP 36
3.3 Famílias de wavelets utilizadas no experimento 37
3.4 Parâmetros do compressor Wavelet 41
3.5 Parâmetros do compressor DCT 44

4.1 Métricas de convergência para diferentes valores de K_i ($K_p = 32$, alvo = 80%, janela de 1.440 pontos, 149 séries da Coleção Pública). Desvio final, pico e vale referem-se à trajetória média do CR ao longo da janela; RMSE é o erro quadrático médio dessa mesma trajetória em relação ao alvo; mediana e IQR referem-se à distribuição do CR final medido individualmente em cada série. . 58
4.2 Métricas de convergência da trajetória média para diferentes valores de K_d ($K_p = 32$, $K_i = 1$, alvo = 80%, janela de 1.440 pontos, 149 séries da Coleção Pública). 62
4.3 Parâmetros do ARC-SDT 64

5.1 Coleções de dados utilizadas na avaliação comparativa 69
5.2 Complexidade algorítmica dos algoritmos avaliados 75
5.3 Tamanhos de janela avaliados e interpretação temporal 77
5.4 Dimensões do experimento e total de combinações 80

6.1 Mediana das métricas de qualidade para CR alvo = 80% — Coleção Pública . . 91

6.2 Mediana das métricas de qualidade para CR alvo = 80% — Coleção Pública (sem outliers SDT) 92
6.3 Mediana das métricas de qualidade para CR alvo = 80% — Coleção Privada . . 95

Lista de Códigos Fonte

3.1 Pseudocódigo do SDT 30
3.2 Pseudocódigo do RDP 33
4.1 Pseudocódigo do ARC-SDT 64

Capítulo 1

Introdução

1.1 Contexto

Em pesquisa publicada pelo IoT Analytics [28], os gastos globais com Internet das Coisas (do inglês, *Internet of Things*, abreviada como IoT) saltaram de US$ 99 bilhões em 2018 para US$ 201 bilhões em 2022, com previsão de US$ 721 bilhões em 2030. Uma das expressões desse crescimento é a digitalização da rede elétrica, conhecida como *smart grid* (rede elétrica inteligente), que demanda sensoriamento contínuo e comunicação constante de grandezas elétricas ao longo de toda a infraestrutura de geração, transmissão e distribuição [27; 24]. Com o avanço da Indústria 4.0 e a popularização desses conceitos, o monitoramento contínuo de grandezas elétricas passou a ser uma necessidade prática em plantas industriais, redes de distribuição de energia e instalações de infraestrutura crítica. Em ambientes industriais, parâmetros como potência ativa e reativa por fase, fator de potência, tensões e correntes de linha, distorção harmônica total e valores de pico são monitorados continuamente para garantir eficiência operacional, conformidade com normas de qualidade de energia e identificação precoce de falhas.

Esses ambientes, no entanto, impõem desafios consideráveis à infraestrutura de comunicação. Estruturas metálicas, máquinas pesadas, paredes de concreto e fontes de interferência eletromagnética dificultam ou inviabilizam o uso de redes WiFi convencionais, que exigem alta densidade de pontos de acesso para cobrir grandes áreas. Redes celulares também podem ser insuficientes em áreas internas de grandes plantas ou em zonas industriais afastadas. Neste contexto, o LoRaWAN (*Long Range Wide Area Network*) surge como uma alternativa viável [10;

44]: opera em faixas de radiofrequência sub-GHz, o que garante maior penetração em ambientes com obstáculos físicos, e oferece alcance de vários quilômetros sem depender de infraestrutura WiFi ou cobertura celular. Em aplicações de monitoramento de energia, os dispositivos de medição tipicamente se alimentam da própria rede monitorada, o que dispensa preocupações com consumo energético do rádio; a vantagem do LoRaWAN está, portanto, principalmente na cobertura e na viabilidade de implantação em locais sem infraestrutura de comunicação convencional.

O LoRaWAN, porém, impõe restrições significativas ao volume de dados transmitido, independentemente do modelo de implantação adotado. Em redes públicas, políticas de uso justo limitam o tempo de *uplink* de cada dispositivo a dezenas de segundos por dia, sem garantia de entrega ou nível de serviço. Em redes privadas com servidor de rede gerenciado por terceiros, elimina-se essa política, mas mantém-se a dependência de infraestrutura externa. Mesmo na configuração mais autônoma, com *gateway* e servidor de rede executados localmente, ainda persistem restrições impostas pela camada física e pela regulamentação espectral: a modulação LoRa (*chirp spread spectrum*) opera em faixas ISM (*Industrial, Scientific and Medical*) sujeitas a limites de ciclo de trabalho (*duty cycle*), e o tamanho máximo da carga útil por pacote varia entre aproximadamente 11 e 222 bytes, conforme o fator de espalhamento utilizado. Em ambientes industriais com maior atenuação, fatores de espalhamento mais elevados são frequentemente necessários, reduzindo tanto a taxa de dados quanto a carga útil disponível por pacote.

Diante dessas restrições, a transmissão direta dos dados brutos pode ser inviável mesmo com intervalos de medição relativamente espaçados. O caso de uso considerado neste trabalho adota um intervalo de medição de um minuto com cerca de 30 grandezas elétricas por leitura; codificadas em ponto flutuante de 32 bits, essas grandezas totalizam aproximadamente 130 bytes por transmissão (120 bytes de dados de medição mais 10 bytes de sobrecarga mínima por leitura para identificação do dispositivo, *timestamp* e campos auxiliares), volume que supera a carga útil disponível nos fatores de espalhamento tipicamente necessários em ambientes industriais. O Capítulo 2 enumera as grandezas monitoradas e quantifica, para cada configuração do LoRaWAN, a taxa mínima de compressão exigida. Sendo assim, a compressão dos dados no próprio dispositivo embarcado torna-se uma necessidade prática para viabilizar a transmissão confiável por LoRaWAN nesse cenário.

Diante desse dimensionamento, é natural questionar se não bastaria transmitir menos grandezas, já que parte delas é derivável das demais: a potência ativa total é a soma das potências por fase, e a reativa total, o fator de potência total e os fatores de potência por fase seguem o mesmo princípio. Cortar essas cerca de seis grandezas, porém, reduz a leitura de 130 para aproximadamente 106 bytes, uma redução inferior a 20%, muito abaixo da taxa de compressão de pelo menos 75% exigida nos fatores de espalhamento mais restritivos (quantificada no Capítulo 2, Tabela 2.2). Eliminar as grandezas derivadas, portanto, não chega perto da compressão necessária, que permanece indispensável. Soma-se a isso que reconstruir essas grandezas no receptor introduziria erro adicional, contaminando a própria avaliação do ganho de compressão que este trabalho busca isolar. Por esse motivo, a decisão sobre quais grandezas transmitir é entendida como uma escolha da aplicação e não faz parte do escopo deste trabalho: adota-se o conjunto completo para caracterizar o pior caso, e estuda-se a compressão em si, cujo resultado se aplica a qualquer subconjunto.

No entanto, potência ativa e tensão, isoladamente, já atendem boa parte das necessidades de monitoramento. É a potência ativa (e a energia dela derivada) que sustenta o faturamento de consumo e a gestão de demanda; é também a grandeza central na identificação não intrusiva de cargas (do inglês, *Non-Intrusive Load Monitoring*, abreviada como NIALM), cujos métodos frequentemente dispensam a potência reativa e operam apenas sobre a potência agregada [51]. A tensão, por sua vez, é o que caracteriza a conformidade do fornecimento, pela permanência dentro da faixa de referência. Reduzida a esse conjunto essencial (as três potências ativas e as três tensões), a carga útil cai de 130 para 34 bytes, o que representa uma compressão de cerca de 74%, ainda abaixo dos 75% exigidos no caso mais restritivo; mesmo o conjunto essencial, portanto, não dispensa a compressão. Como ela permanece necessária, o trabalho mantém seu valor ao recair justamente sobre a grandeza mais importante do monitoramento. A tensão, por se esperar que varie pouco, é ainda mais favorável à compressão, um problema semelhante e mais simples cuja análise sistemática fica como trabalho futuro, embora o Apêndice C já traga uma demonstração inicial de que ela é comprimida com folga.

Surge então a pergunta: até onde é possível comprimir sem comprometer a utilidade dos dados? Como ponto de partida, algoritmos de compressão sem perda (do inglês, *lossless*) foram avaliados neste trabalho como etapa preliminar. Os resultados, detalhados no Apêndice A, mostram que algoritmos sem perda como o *brotli* podem atingir razões de compressão de até 5,3

com janelas diárias (quociente entre o tamanho original e o comprimido, equivalente a uma taxa de compressão de 81,1% na definição adotada neste trabalho), o que é suficiente para alguns cenários de transmissão. Contudo, essa abordagem apresenta limitações estruturais relevantes para o caso de uso considerado. Primeiro, a razão de compressão obtida não é controlável: ela depende das características estatísticas do sinal no período considerado e não é garantida de uma série para outra ou de um ambiente para outro, tornando inviável seu uso como mecanismo confiável para satisfazer restrições de carga útil. Segundo, janelas de agregação de várias horas são necessárias para que a compressão sem perda atinja razões expressivas, e isso impõe latências incompatíveis com qualquer forma de monitoramento em tempo próximo ao real. Terceiro, manter um *buffer* de horas de dados brutos em memória, junto com as estruturas internas do algoritmo de compressão, representa custo de RAM proibitivo para a maioria dos sistemas embarcados. Para cenários com requisitos mais restritivos de carga útil ou de latência de entrega, é necessário algo que comprima mais do que a compressão sem perda consegue oferecer de forma confiável e previsível.

A compressão com perda (do inglês, *lossy*) permite razões maiores e controle explícito sobre o volume produzido, mas levanta uma questão que a compressão sem perda não precisa responder: quais características do sinal podem ser descartadas sem prejudicar a aplicação. A resposta depende do contexto: monitoramentos voltados à qualidade de energia não podem descartar transitórios nem componentes harmônicas, enquanto aplicações focadas em consumo energético podem tolerar algum grau de suavização, desde que grandezas como energia acumulada e pico de demanda sejam preservadas com precisão adequada.

O LoRaWAN representa apenas um exemplo contemporâneo de cenário onde a transmissão de dados brutos de monitoramento se torna inviável ou custosa; o objetivo da compressão não é dar suporte a uma tecnologia específica, mas permitir que informações sejam descartadas de forma inteligente e controlada. Mesmo em cenários onde a largura de banda seria suficiente para transmitir os dados sem compressão, o uso de algoritmos com perda ainda traz benefícios concretos: menor ocupação de banda, redução do custo de armazenamento e a possibilidade de aumentar a granularidade temporal das medições sem ultrapassar os limites da rede. Há ainda um benefício menos óbvio, mas relevante em ambientes industriais e rurais com sinal WiFi ou móvel instável: a confiabilidade da transmissão. Quando a taxa de perda de pacotes é alta por condições ruins de sinal, usar menos pacotes por leitura amplia a janela de oportunidade para

retransmissões. Um sistema que precisa enviar dez pacotes por leitura consome quase todo o orçamento de banda ou de ciclo de trabalho disponível apenas com dados novos, deixando pouca margem para novas tentativas em caso de falha; o mesmo sistema, após compressão, usando dois ou três pacotes, libera capacidade suficiente para retransmitir em caso de perda, reduzindo o risco de lacunas no histórico de monitoramento.

Entre as abordagens de compressão com perda avaliadas, o SDT (*Swinging Door Trending*, do inglês) processa o sinal amostra a amostra, sem exigir o armazenamento do histórico completo, respondendo diretamente à restrição de memória já apontada para a compressão sem perda. Apresenta características particularmente favoráveis para uso em dispositivos embarcados: tem custo computacional e de memória compatíveis com hardware restrito e, do ponto de vista legal, sua patente original [15] expirou em 2005, permitindo seu uso livre sem restrições de propriedade intelectual. No entanto, o SDT em sua formulação original não oferece controle direto sobre a taxa de compressão produzida, pois o único parâmetro de configuração é uma tolerância de erro absoluta cujo efeito depende das características específicas de cada sinal, dificultando o uso em cenários com restrições de largura de banda, onde é necessário garantir que o volume transmitido não ultrapasse um limite definido. Diante dessa lacuna, este trabalho propõe o ARC-SDT (*Adaptive Real-time Controlled* SDT), uma adaptação do SDT com controle direto sobre a taxa de compressão, detalhada no Capítulo 4.

Não existe, porém, uma resposta universal sobre qual algoritmo de compressão com perda é mais adequado para sinais de monitoramento de energia, seja entre as abordagens já estabelecidas na literatura, seja para a adaptação proposta. Alguns algoritmos clássicos, baseados em transformadas, aproximação por segmentos e codificação preditiva, são utilizados em outros domínios, mas têm desempenho relativo pouco avaliado nesse contexto específico. Escolher um algoritmo sem uma avaliação criteriosa pode resultar em soluções subótimas em termos de taxa de compressão, fidelidade do sinal reconstruído ou custo computacional para os dispositivos embarcados. Este trabalho realiza essa avaliação: uma comparação entre alguns algoritmos clássicos de compressão com perda e o ARC-SDT proposto, cobrindo taxa de compressão, fidelidade do sinal reconstruído e custo computacional para dispositivos embarcados.

1.2 Perguntas de Pesquisa

As questões que motivam este trabalho podem ser organizadas em três perguntas centrais, formuladas antes de qualquer experimento e cujas respostas não eram conhecidas a priori:

1. A compressão com perda é viável para séries temporais de grandezas elétricas, considerando os requisitos de fidelidade de aplicações de monitoramento de consumo e qualidade de energia?

2. Dentre os algoritmos de compressão com perda avaliados, algum apresenta desempenho consistentemente superior nas métricas de fidelidade relevantes para monitoramento de energia, ou os resultados são comparáveis entre as abordagens?

3. Para dispositivos embarcados com recursos limitados, existe algum algoritmo de compressão com perda, ou adaptação de algum, que se destaque em relação aos demais?

1.3 Contribuições

As principais contribuições deste trabalho são:

- Uma avaliação comparativa de algoritmos de compressão com e sem perda aplicados a séries temporais de grandezas elétricas, avaliando taxa de compressão e fidelidade do sinal reconstruído;

- A definição de um conjunto de métricas de fidelidade para avaliação de sinais de grandezas elétricas comprimidos, cobrindo diferentes dimensões de fidelidade, como a preservação da forma do sinal, a fidelidade energética e outras características relevantes para o monitoramento elétrico;

- A análise do custo-benefício entre taxa de compressão e qualidade do sinal reconstruído para diferentes classes de algoritmos no contexto de monitoramento de energia;

- A proposição do algoritmo ARC-SDT (*Adaptive Real-time Controlled* SDT), uma adaptação do SDT com controle direto sobre a taxa de compressão produzida, cujo desempenho é avaliado comparativamente aos demais algoritmos estudados neste trabalho.

Capítulo 2

Fundamentação Teórica

2.1 Monitoramento de Grandezas Elétricas

O monitoramento contínuo de grandezas elétricas é condição necessária para diagnosticar problemas de qualidade de energia, como distorções harmônicas, variações de tensão e baixo fator de potência, e para subsidiar a otimização do uso da energia em instalações industriais e comerciais [1]. Na prática, esse monitoramento é realizado por redes elétricas inteligentes (do inglês, *smart grids*), sistemas de medição avançada (do inglês, *Advanced Metering Infrastructure*, abreviada como AMI) e dispositivos embarcados instalados em campo, que registram e transmitem as grandezas de interesse com resolução suficiente para detectar anomalias, acionar alarmes e viabilizar faturamento baseado em consumo real. As grandezas primárias mensuradas são a tensão e a corrente, das quais derivam as demais grandezas de interesse.

A potência ativa (P) representa a energia efetivamente convertida em trabalho e é essencial em aplicações de faturamento e controle de carga. Já a potência reativa (Q), associada à energia armazenada e devolvida ciclicamente por elementos indutivos e capacitivos, não realiza trabalho útil, mas circula continuamente pela rede, ocupando capacidade de transmissão e elevando as perdas por efeito Joule. Esse comportamento impacta diretamente o fator de potência (FP $= P/S$), que mede a proporção da energia fornecida que é efetivamente convertida em trabalho. Valores baixos indicam alta circulação de corrente reativa, o que sobrecarrega a infraestrutura elétrica sem benefício energético direto. É importante destacar que, no Brasil, a Agência Nacional de Energia Elétrica (ANEEL) exige fator de potência mínimo de 0,92 em instalações de média e alta tensão, e consumidores que não atendem a esse limite estão sujeitos

a penalizações tarifárias [2]. Sendo assim, o monitoramento contínuo do fator de potência é essencial em ambientes industriais e comerciais, onde cargas indutivas como motores e transformadores são predominantes. A frequência da rede é monitorada em sistemas de proteção e controle, pois desvios indicam desequilíbrios entre geração e consumo. A análise harmônica, por sua vez, é necessária em ambientes com cargas não lineares, como inversores e fontes chaveadas, onde a distorção harmônica total (*Total Harmonic Distortion*, THD) compromete a qualidade da energia.

A Tabela 2.1 apresenta as grandezas que compõem uma leitura típica de um dispositivo de monitoramento trifásico, totalizando 30 valores por leitura, organizadas por fase e por grandezas globais do sistema.

Tabela 2.1: Grandezas elétricas de um dispositivo de monitoramento trifásico típico (30 valores por leitura)

Grandeza	**Descrição**
Por fase (Fase A, Fase B e Fase C): 24 valores	
Tensão de fase (V_A, V_B, V_C)	Valor eficaz (RMS) da tensão entre cada fase e o neutro. Indica o nível de tensão entregue às cargas e é referência para detecção de subtensão, sobretensão e desequilíbrio.
Corrente de fase (I_A, I_B, I_C)	Valor eficaz da corrente em cada fase. Permite identificar sobrecarga, desequilíbrio de cargas e correntes de falta.
Potência ativa (P_A, P_B, P_C)	Potência efetivamente convertida em trabalho em cada fase, medida em watts (W). Base para faturamento e análise de eficiência energética.
Potência reativa (Q_A, Q_B, Q_C)	Potência associada à energia armazenada e devolvida por cargas reativas em cada fase, medida em var. Determina o impacto de cada fase no fator de potência do sistema.
Fator de potência (FP_A, FP_B, FP_C)	Relação entre potência ativa e aparente em cada fase. Indica a eficiência com que a energia fornecida é utilizada.
THD de tensão (THD_{V_A}, THD_{V_B}, THD_{V_C})	Distorção harmônica total da tensão em cada fase. Quantifica a presença de componentes harmônicas na tensão, indicando qualidade da energia fornecida.

continua na próxima página

Tabela 2.1 – continuação

Grandeza	**Descrição**
THD de corrente (THD_{I_A}, THD_{I_B}, THD_{I_C})	Distorção harmônica total da corrente em cada fase. Reflete a presença de cargas não lineares como inversores, fontes chaveadas e dispositivos eletrônicos.
Tensão de pico ($V_{\text{pico},A}$, $V_{\text{pico},B}$, $V_{\text{pico},C}$)	Valor máximo de tensão registrado em cada fase durante o intervalo de medição. Relevante para detecção de surtos e sobretensões transitórias.
Grandezas globais do sistema: 6 valores	
Potência ativa total (P_T)	Soma das potências ativas das três fases. Principal grandeza para faturamento de energia e controle de demanda.
Potência reativa total (Q_T)	Soma das potências reativas das três fases. Determina a necessidade de correção de fator de potência no sistema.
Fator de potência total (FP_T)	Fator de potência resultante do sistema trifásico. Parâmetro regulado pela ANEEL para apuração de penalidades tarifárias.
Corrente do neutro (I_N)	Corrente no condutor neutro, resultado do desequilíbrio entre as fases. Valores elevados indicam desequilíbrio de cargas ou presença de harmônicos ímpares de ordem 3.
Frequência (f)	Frequência fundamental da rede elétrica (nominal: 60 Hz no Brasil). Desvios indicam desequilíbrio entre geração e consumo e são críticos para sistemas de proteção.
Corrente de pico (I_{pico})	Valor máximo de corrente registrado em qualquer fase durante o intervalo de medição. Relevante para análise de partidas de equipamentos e eventos de sobrecarga.

Fonte: elaborada pelo autor.

2.2 Restrições de Comunicação

Aplicações de monitoramento distribuído em ambientes industriais, remotos e urbanos frequentemente operam em condições desfavoráveis à comunicação sem fio. A ausência ou insuficiência de infraestrutura de conectividade impõe restrições severas à transmissão dos dados

coletados, motivando o uso de tecnologias de comunicação especializadas. Esta seção apresenta as limitações das tecnologias convencionais nesses contextos, as alternativas disponíveis e as características técnicas do LoRaWAN, tecnologia adotada como referência neste trabalho.

2.2.1 Limitações das Tecnologias de Comunicação Convencionais

Redes WiFi, embora amplamente disponíveis em ambientes comerciais e residenciais, apresentam limitações significativas em cenários industriais e remotos. A tecnologia opera nas faixas de 2,4 GHz e 5 GHz, frequências mais suscetíveis à atenuação por obstáculos físicos como paredes de concreto, estruturas metálicas e maquinário pesado. Em plantas industriais, a combinação de barreiras físicas densas com fontes intensas de interferência eletromagnética, como motores de indução, inversores de frequência e equipamentos de solda, compromete severamente o alcance e a confiabilidade do sinal WiFi. O alcance prático em ambientes internos com obstáculos raramente supera algumas dezenas de metros por ponto de acesso, o que implica a necessidade de uma infraestrutura cara e densa para cobrir grandes áreas industriais. Em locais remotos, como subestações de energia, campos agrícolas ou instalações de extração mineral, a ausência de infraestrutura elétrica e de rede torna a implantação de WiFi inviável.

Redes celulares de segunda e quarta geração (2G e 4G) oferecem cobertura geográfica mais ampla, mas também apresentam limitações práticas relevantes. Em ambientes subterrâneos, como túneis, minas, porões industriais e câmaras de subestações enterradas, o sinal celular é fortemente atenuado ou completamente ausente, pois as ondas de radiofrequência nessas faixas não penetram eficientemente em estruturas de concreto e metal. Em zonas rurais e industriais afastadas dos centros urbanos, a cobertura celular pode ser inexistente ou de qualidade insuficiente para comunicação confiável de dados. Mesmo em áreas com cobertura nominal, a penetração do sinal em estruturas densas pode ser insatisfatória, criando pontos cegos no interior de edificações industriais.

2.2.2 Redes de Longa Distância e Baixa Potência (LPWAN)

Diante das limitações das tecnologias convencionais, surgiram as redes de longa distância e baixa potência (*Low Power Wide Area Networks*, LPWAN), projetadas especificamente para comunicação de dados em aplicações de IoT com requisitos de baixa taxa de dados e longo

alcance. Entre as principais tecnologias LPWAN disponíveis estão o NB-IoT, o Sigfox e o LoRaWAN [44].

O NB-IoT depende da infraestrutura das operadoras celulares, implicando custo recorrente por dispositivo e indisponibilidade onde a cobertura não foi implantada. O Sigfox, com carga útil de apenas 12 bytes por mensagem e 140 mensagens por dia, é inadequado para múltiplas grandezas por leitura. O LoRaWAN distingue-se por admitir os dois modelos: pode ser contratado de um provedor de rede, como ocorre com o NB-IoT e o Sigfox, ou implantado de forma completamente privada, com *gateway* e servidor de rede próprios e sem dependência de terceiros.

2.2.3 LoRaWAN

O LoRaWAN é um protocolo de rede de longa distância e área ampla desenvolvido e mantido pela LoRa Alliance, composto por duas camadas distintas: a camada física LoRa, baseada na modulação *Chirp Spread Spectrum* (CSS), e o protocolo MAC (*Medium Access Control*) LoRaWAN, que define a arquitetura de rede e as regras de acesso ao meio [10].

Camada Física LoRa

A modulação LoRa opera em faixas de radiofrequência não licenciadas, conhecidas como faixas ISM (*Industrial, Scientific and Medical*). No Brasil e nas Américas, a faixa utilizada é a de 915 MHz, enquanto na Europa é utilizada a faixa de 868 MHz. A operação em frequências sub-GHz confere ao LoRa maior capacidade de penetração em obstáculos e maior alcance em comparação com tecnologias que operam em 2,4 GHz ou 5 GHz.

O comportamento da camada física LoRa é determinado por dois parâmetros principais:

Fator de espalhamento (*Spreading Factor*, SF): varia de SF7 a SF12 e é o principal parâmetro de configuração da camada física. Um símbolo é a unidade básica de informação transmitida pelo rádio, e sua duração é determinada pelo SF. Quanto maior o SF, mais longa é a duração de cada símbolo, o que aumenta a sensibilidade do receptor e o alcance do sinal. Em contrapartida, maior SF resulta em menor taxa de dados efetiva e maior tempo de ocupação do canal (*time on air*, abreviado como ToA). A relação entre SF e taxa de dados é aproximadamente exponencial: o aumento de um nível de SF dobra o tempo de transmissão de um mesmo pacote. Fatores de espalhamento mais altos, como SF11 e SF12, são tipicamente necessários em

ambientes com maior atenuação, como interiores de plantas industriais ou locais com múltiplos obstáculos.

Largura de banda (*Bandwidth*, BW): tipicamente 125 kHz ou 500 kHz. Maior largura de banda resulta em maior taxa de dados e menor tempo de transmissão, porém reduz a sensibilidade do receptor e, consequentemente, o alcance. A largura de banda de 125 kHz é a configuração padrão para a maioria das aplicações LoRaWAN.

A combinação desses parâmetros determina a taxa de dados efetiva, o alcance e o tamanho máximo da carga útil por transmissão. Em configurações de SF mais alto, o tamanho máximo da carga útil por pacote é reduzido pelo plano de frequências regional, variando de cerca de 11 bytes em SF10 até 222 bytes em SF7, conforme o plano de banda adotado.

Protocolo e Arquitetura de Rede

O protocolo LoRaWAN define uma arquitetura em estrela com quatro elementos principais: dispositivos finais, *gateways*, servidor de rede e servidor de aplicação.

Dispositivos finais são os nós sensores ou atuadores da rede, responsáveis pela coleta de dados e transmissão via rádio LoRa. Os dispositivos de monitoramento elétrico operam em **Classe C**, que mantém o receptor em escuta contínua (adequado para equipamentos com alimentação permanente pela rede elétrica monitorada).

Gateways recebem as transmissões LoRa e as encaminham por conexão IP ao servidor de rede. Um único *gateway* com antena externa pode cobrir dezenas de quilômetros em área aberta e vários andares em ambientes internos, servindo simultaneamente a milhares de dispositivos.

O servidor de rede (*Network Server*) é responsável pela autenticação dos dispositivos, deduplicação de pacotes e agendamento de *downlinks*. Pode ser executado em infraestrutura própria (ex: ChirpStack embarcado no próprio *gateway*), eliminando dependência de terceiros.

O servidor de aplicação recebe os dados decodificados pelo servidor de rede e os disponibiliza à aplicação final, tipicamente por meio de APIs ou protocolos como MQTT e HTTP.

Modelos de Implantação e Suas Limitações

A escolha do modelo de implantação afeta o grau de controle, a privacidade dos dados e as restrições operacionais a que a rede está sujeita. Em redes públicas, como a The Things Network, políticas de uso justo (*fair use policy*) limitam o tempo de *uplink* de cada dispositivo a 30 se-

gundos por dia, o que equivale, dependendo do SF, a poucas dezenas de transmissões diárias, sem garantia de entrega nem nível de serviço contratado. Em redes privadas com servidor de rede gerenciado por terceiros, eliminam-se as restrições de uso justo, mas mantém-se a dependência de infraestrutura externa e eventuais restrições contratuais, além de implicar o trânsito dos dados por servidores fora do controle do operador do sistema.

A configuração mais autônoma consiste em *gateway* privado com servidor de rede executado localmente, inclusive no próprio hardware do *gateway*. Essa configuração elimina toda dependência de terceiros. Ainda assim, continuam existindo restrições impostas pela modulação LoRa e pela regulamentação espectral, que fazem parte do próprio padrão, independentemente do modelo de implantação. No plano de frequências AU915 (915–928 MHz), o protocolo LoRaWAN impõe, por padrão, duas restrições de transmissão: um tempo máximo de permanência no canal (*dwell time*) de 400 ms por pacote, e um intervalo mínimo recomendado de 20 segundos entre transmissões consecutivas [35]. A primeira restrição é imposta pela regulação da faixa espectral; a segunda corresponde ao intervalo de 20 segundos explicitamente recomendado pela Seção 3.8.3 do documento RP002-1.0.5 [35] (o texto da especificação determina literalmente: *"Twenty seconds (20s) are RECOMMENDED between two uplink transmissions when UplinkDwellTime = 1"*), valor que pode ser ajustado conforme a regulação local. Embora o protocolo preveja uma configuração alternativa com *UplinkDwellTime*=0 que eliminaria ambas as restrições, ela exige habilitação explícita pelo servidor de rede via comando MAC *TxParamSetupReq*, e só pode ser utilizada quando a regulação local da faixa espectral o permite [35]. Por padrão de inicialização, todo dispositivo AU915 opera com *UplinkDwellTime*=1; a análise que se segue adota essa configuração como referência conservadora, representando o cenário válido sem dependência de autorização regulatória específica.

Exemplo Concreto: Monitoramento Trifásico com LoRaWAN

Para ilustrar o impacto dessas restrições e mostrar que o cenário de uso deste trabalho é um caso onde a compressão de dados é necessária, considera-se um dispositivo de monitoramento elétrico trifásico com intervalo de medição de um minuto. Cada leitura compreende as seguintes grandezas: potência ativa por fase e total (4 valores), potência reativa por fase e total (4), fator de potência por fase e total (4), tensão das três fases (3), corrente das três fases (3), corrente do neutro (1), frequência da rede (1), distorção harmônica total de tensão por fase (3), distorção

harmônica total de corrente por fase (3), corrente de pico no intervalo (1) e tensão de pico por fase (3). Esse conjunto totaliza 30 grandezas por leitura. Representadas como valores em ponto flutuante de 32 bits (4 bytes cada), apenas os dados de medição ocupam $30 \times 4 = 120$ bytes; acrescidos de 10 bytes de sobrecarga mínima por leitura para identificador do dispositivo, *timestamp* da medição e campos auxiliares de alerta, cada leitura totaliza 130 bytes a serem transmitidos.

A Tabela 2.2 consolida, para cada fator de espalhamento do plano AU915 com *UplinkDwellTime*=1, o limite máximo de carga útil por pacote (valores reproduzidos da Tabela 44 de [35]), o número de pacotes necessários para transmitir os 130 bytes da leitura, o ToA por pacote, calculado pela fórmula padrão LoRa para BW = 125 kHz e taxa de codificação 4/5, e o intervalo mínimo de transmissão resultante com 20 segundos entre pacotes consecutivos.

Tabela 2.2: Viabilidade de transmissão de uma leitura de 130 bytes por SF no plano AU915 com *UplinkDwellTime*=1: carga útil máxima conforme Tabela 44 de [35], ToA por pacote e intervalo mínimo de 20 s entre pacotes (BW = 125 kHz, taxa de codificação 4/5)

SF	**Carga útil máx. (bytes)**	**Pacotes por leitura**	**ToA por pacote**	**Intervalo mín. (20 s × pacotes)**	**Redução necessária dos dados**
SF12	N/A*	N/A	N/A	N/A	Inviável
SF11	N/A*	N/A	N/A	N/A	Inviável
SF10	11	12	≈289 ms	240 s (= 4 min)	≥75%
SF9	53	3	≈329 ms	60 s (= 1 min)	0%†
SF8	125	2	≈369 ms	40 s	Não necessária
SF7	222	1	≈349 ms	20 s	Não necessária

Fonte: elaborada pelo autor.

*Em SF12 e SF11, a carga útil máxima é N/A no plano AU915 com *UplinkDwellTime*=1: o tempo de permanência no canal excede o limite de 400 ms mesmo com carga útil mínima, tornando essas taxas de dados indisponíveis nessa configuração [35].

†Em SF9, os 130 bytes cabem em 3 pacotes de 53 bytes (capacidade total = 159 bytes) e o intervalo

mínimo de $3 \times 20\,\text{s} = 60\,\text{s}$ coincide exatamente com o limite de 1 minuto, sem qualquer margem. Em um sistema de monitoramento com *uplinks* confirmados (necessários para garantir que nenhum dado seja perdido [34]), qualquer retransmissão por falha na confirmação de recebimento (ACK) consome uma janela adicional de 20 s, ultrapassando o limite. Em SF8, os 130 bytes exigem 2 pacotes de 125 bytes (intervalo base de 40 s), sobrando apenas uma janela para retransmissão dentro de 60 s; SF7, com 1 pacote de 222 bytes e intervalo base de 20 s, suporta até 2 retransmissões dentro de 60 s, tornando-o o limiar confiável para transmissões confirmadas.

Em ambientes industriais com obstáculos e interferências, o ADR (*Adaptive Data Rate*) tende a selecionar fatores de espalhamento mais altos para garantir a comunicação. Esses são exatamente os cenários mais realistas na prática: plantas industriais com estruturas metálicas, paredes de concreto e múltiplos andares frequentemente exigem SF9 a SF11; SF12 é comum em situações de máxima atenuação, como ambientes subterrâneos ou grandes distâncias com obstáculos. Com *UplinkDwellTime*=1, SF11 e SF12 sequer estão disponíveis, de modo que SF10 e SF9 representam os fatores de espalhamento mais restritivos alcançáveis. SF7 e SF8, que dispensariam compressão, correspondem a condições de linha de visada ou ambientes abertos com curta distância ao *gateway*, situações menos comuns no contexto industrial motivador deste trabalho.

Sendo assim, nos fatores de espalhamento alcançáveis com *UplinkDwellTime*=1, a compressão é obrigatória nos casos mais restritivos. Em SF10, os 130 bytes de uma leitura exigem 12 pacotes de 11 bytes cada, totalizando intervalo mínimo de $12 \times 20\,\text{s} = 240\,\text{s}$, quatro vezes acima do limite de 1 minuto; para viabilizar a transmissão dentro de 60 s são necessários no máximo $\lfloor 60/20 \rfloor = 3$ pacotes, o que impõe comprimir os dados a no máximo $3 \times 11 = 33$ bytes, equivalente a uma taxa de compressão de $1 - 33/130 \approx 75\%$. Em SF9, os dados cabem em 3 pacotes de 53 bytes sem compressão alguma, mas o intervalo de exatamente 60 s não deixa margem para retransmissões; com *uplinks* confirmados, em SF8 os 130 bytes exigem 2 pacotes com intervalo base de 40 s, sobrando apenas uma janela para retransmissão dentro de 60 s, enquanto SF7 comporta tudo em 1 pacote com intervalo base de 20 s, suportando até duas retransmissões dentro de 60 s. Vale notar que, mesmo em SF7 e SF8, onde a transmissão sem compressão seria viável para intervalos de um minuto, a redução do intervalo de coleta para ordens de grandeza menores tornaria a compressão igualmente necessária: 130 bytes a cada segundo correspondem a 7.800 bytes por minuto, volume incompatível com qualquer configuração LoRaWAN. A compressão deixa de ser, nesse contexto, apenas um requisito de cobertura e passa a ser também um requisito de granularidade temporal.

2.3 Compressão de Dados

Diante da necessidade de reduzir o volume de dados gerados pelo monitoramento elétrico sem descartar informação relevante, como formalizar precisamente o que significa comprimir uma série temporal? Compressão de dados, também chamada de codificação de fonte, é definida como o processo de converter um fluxo de dados de entrada em outro fluxo de dados de saída com tamanho menor [17]. Um princípio teórico frequentemente citado para orientar esse processo é a chamada teoria do poder de simplicidade, segundo a qual o objetivo da compressão é eliminar redundância mantendo o maior poder descritivo possível sobre os dados originais. No contexto específico de séries temporais, três propriedades costumam ser exploradas pelos algoritmos de compressão para atingir esse objetivo:

- **Redundância**: segmentos de uma série temporal podem se repetir, seja dentro da própria série, seja entre séries relacionadas; nas grandezas elétricas monitoradas neste trabalho, essa redundância aparece como correlação entre amostras consecutivas e como padrões de consumo que se repetem ao longo do dia;
- **Aproximabilidade**: em muitos casos, o sinal produzido por um sensor pode ser aproximado por uma função matemática simples (um polinômio, uma soma de senoides, uma base de *wavelets*), o que permite representar um segmento inteiro por um pequeno conjunto de coeficientes;
- **Previsibilidade**: quando o comportamento futuro do sinal pode ser inferido a partir do seu histórico recente, é possível transmitir apenas os desvios em relação ao valor previsto, em vez do valor bruto.

De modo mais formal, um algoritmo de compressão é composto por duas partes complementares: um codificador, responsável por receber a série temporal original e produzir uma representação comprimida de tamanho menor, e um decodificador, responsável por reconstruir uma série a partir dessa representação comprimida [17]. É exatamente esse par codificador-decodificador que define as duas categorias fundamentais de algoritmos de compressão: sem perda e com perda. Se a série reconstruída pelo decodificador é idêntica à série original, o algoritmo é dito sem perda; caso contrário, é dito com perda. Os algoritmos avaliados neste trabalho são descritos individualmente em capítulo próprio; nesta seção são definidos apenas os conceitos que os classificam. O objetivo deste trabalho não é determinar uma taxa de compressão alvo específica, mas sim avaliar o potencial de cada técnica em termos de taxa de compressão e fidelidade do sinal reconstruído, permitindo identificar quais abordagens são mais adequadas para o caso de uso de monitoramento de grandezas elétricas.

2.3.1 Compressão sem Perda

Algoritmos de compressão sem perda permitem a reconstrução exata do sinal original a partir dos dados comprimidos: a reconstrução é idêntica, bit a bit, sem introdução de qualquer erro. Isso é possível porque esses algoritmos exploram apenas a redundância presente nos dados (repetições e correlações), sem tocar na informação em si.

A limitação dessa abordagem é que a taxa de compressão atingível é restrita pela entropia do sinal. Sinais com alta variabilidade e baixa correlação temporal, como as grandezas elétricas amostradas em alta frequência, oferecem pouca redundância explorável, resultando em taxas de compressão modestas. Quando o objetivo é reduzir significativamente o volume de dados para transmissão em redes com restrições severas de carga útil, a compressão sem perda pode ser insuficiente, como demonstrado empiricamente no Apêndice A.

2.3.2 Compressão com Perda

Algoritmos de compressão com perda relaxam a exigência de reconstrução idêntica, permitindo reduções maiores ao custo de introduzir erro na reconstrução do sinal. Ao contrário da compressão sem perda, o processo não é invertível: o sinal reconstruído é uma aproximação do original, e a informação descartada não pode ser recuperada. O descarte é feito de forma controlada, explorando tanto a redundância quanto a aproximabilidade e a previsibilidade descritas anteriormente, ou seja, componentes do sinal que podem ser removidos ou aproximados sem comprometer significativamente a utilidade dos dados para a aplicação.

A noção de perda aceitável depende diretamente do contexto. Para monitoramento de qualidade de energia, componentes harmônicas e transitórios são informativos e não devem ser descartados. Para monitoramento de consumo energético com granularidade de um minuto, algum grau de suavização pode ser tolerado desde que grandezas integradas como energia acumulada e pico de demanda sejam preservadas com fidelidade adequada. Sendo assim, a escolha do algoritmo e de seus parâmetros envolve um compromisso entre taxa de compressão e qualidade do sinal reconstruído, avaliado pelas métricas definidas na seção seguinte.

2.3.3 Classificação dos Algoritmos Avaliados

Onde os algoritmos avaliados nesta dissertação se posicionam dentro do panorama mais amplo de técnicas de compressão de séries temporais? Chiarot e Silvestri [17] organizam os algoritmos da literatura em quatro grupos: os baseados em dicionário, que guardam trechos repetidos da série para reutilizá-los

depois; os de aproximação por funções, que dividem a série em pedaços e representam cada um por uma função matemática simples, como uma reta ou um polinômio; os autoencoders, redes neurais treinadas para reconstruir o sinal a partir de uma versão reduzida dele; e os algoritmos sequenciais, que combinam técnicas simples de codificação, uma atrás da outra.

Os algoritmos sem perda avaliados no Apêndice A pertencem ao grupo baseado em dicionário, já que todos reaproveitam trechos repetidos do fluxo de bytes. Os algoritmos com perda avaliados no Capítulo 3, incluindo a adaptação proposta neste trabalho (Capítulo 4), pertencem ao grupo de aproximação por funções: todos dividem a série em segmentos e aproximam cada um por uma reta dentro de um erro máximo definido antecipadamente, mesmo que o levantamento de Chiarot e Silvestri não cite esses algoritmos pelo nome.

Os outros dois grupos ficam fora do escopo desta avaliação por delimitação definida antes dos experimentos, e não por desempenho medido. Os autoencoders pressupõem uma etapa de treinamento prévio sobre dados representativos e a manutenção dos pesos resultantes em memória durante toda a operação, o que é incompatível com a premissa adotada aqui de um dispositivo que comprime o sinal à medida que ele chega, sem conhecimento prévio do perfil de consumo da instalação em que opera. Os algoritmos sequenciais, por sua vez, são técnicas sem perda, e a compressão sem perda foi posta de lado no início desta pesquisa: como a aplicação admite perda controlada, as abordagens com perda se mostraram o caminho mais promissor para caber na carga útil disponível. O Apêndice A registra a avaliação preliminar que sustentou essa decisão.

2.4 Métricas de Avaliação

Uma única métrica é insuficiente para caracterizar o comportamento de um algoritmo de compressão com perda em todos os aspectos relevantes para o monitoramento elétrico: um método pode preservar bem a forma geral do sinal mas distorcer picos de demanda, ou conservar a energia acumulada mas introduzir erros locais expressivos. Por esse motivo, este trabalho adota um conjunto de métricas organizadas em quatro dimensões: taxa de compressão, fidelidade global do sinal, fidelidade energética e fidelidade de picos.

Nas definições a seguir, $\mathbf{x} = (x_1, \ldots, x_N)$ denota o sinal original e $\hat{\mathbf{x}} = (\hat{x}_1, \ldots, \hat{x}_N)$ o sinal reconstruído, ambos com N amostras.

2.4.1 Taxa de Compressão

A taxa de compressão (*Compression Rate*, CR) quantifica a redução de dados obtida pelo algoritmo, expressa como a fração do volume original que foi eliminada:

$$\text{CR} = 100 \cdot \left(1 - \frac{N_c}{N}\right) \tag{2.1}$$

onde N é o número de amostras da série original e N_c o número de elementos retidos após a compressão (*keypoints*, no caso do SDT e do RDP, ou coeficientes, no caso da DCT e da DWT). Uma CR de 80% indica que o dado comprimido ocupa apenas 20% do tamanho original.

No SDT e no RDP, cada elemento retido ocupa 8 bytes, exatamente a mesma estrutura de um ponto do dado bruto, de modo que a razão em bytes coincide com a razão N_c/N. Na DCT e na DWT, o custo é calculado diretamente em bytes na implementação, somando 8 bytes por coeficiente mantido e um *overhead* fixo com os metadados obrigatórios de reconstrução; a fórmula acima continua valendo, desde que N_c e N sejam lidos como volumes em bytes (detalhes na Seção 5.4.4).

2.4.2 Fidelidade Global do Sinal

As métricas de fidelidade global avaliam a diferença entre o sinal original e o reconstruído de forma agregada, sem privilegiar regiões específicas do sinal.

Métricas de Erro Global

As métricas de erro global quantificam a magnitude média do desvio ponto a ponto entre o sinal original e o reconstruído. A base dessa família é o erro quadrático médio (MSE, *Mean Squared Error*) e sua raiz (RMSE, *Root Mean Squared Error*), ambas apresentadas por Chiarot e Silvestri [17] entre os índices de qualidade usuais para compressão de séries temporais:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{2.2}$$

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{2.3}$$

A Figura 2.1 ilustra esse desvio ponto a ponto entre o sinal original e o reconstruído. As demais métricas do grupo são transformações diretas do mesmo erro quadrático, diferindo apenas em normalização e escala. O NRMSE é simplesmente uma versão normalizada do RMSE, dividida pela amplitude

do sinal para permitir comparação entre séries de escalas diferentes; o PRD é definido por Deepu, Heng e Lian [20] no contexto de compressão de dados em sensores sem fio para IoT; e o SNR e o PSNR são apresentados por Chiarot e Silvestri [17]:

$$\text{NRMSE} = \frac{\text{RMSE}}{\max(\mathbf{x}) - \min(\mathbf{x})} \tag{2.4}$$

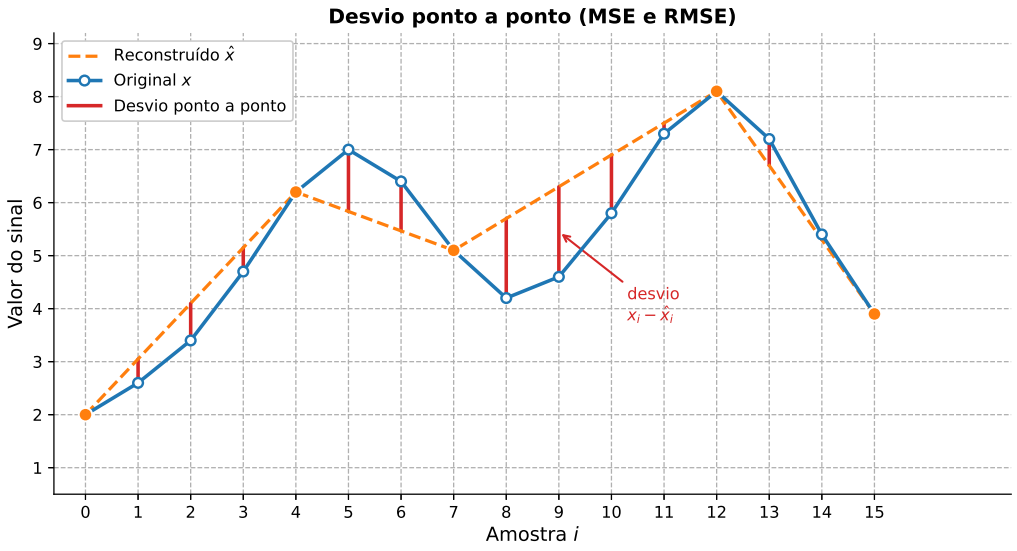
$$\text{PRD} = 100 \cdot \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{N} x_i^2}} \tag{2.5}$$

$$\text{SNR} = 10\log_{10}\left(\frac{\sum_{i=1}^{N} x_i^2}{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}\right) \text{ (dB)} \tag{2.6}$$

$$\text{PSNR} = 10\log_{10}\left(\frac{\max(\mathbf{x})^2}{\text{MSE}}\right) \text{ (dB)} \tag{2.7}$$

Essas seis variantes (MSE, RMSE, NRMSE, PRD, SNR, PSNR) capturam a mesma dimensão de informação em escalas distintas e são tratadas como um grupo redundante neste trabalho; apenas o MAPE, por medir o erro relativo ponto a ponto, traz informação complementar às demais.

Figura 2.1 — Desvio ponto a ponto entre o sinal original x (azul) e o reconstruído $\hat{x}$ (laranja). Cada segmento vermelho representa a diferença $x_i - \hat{x}_i$ em uma amostra, e é a média dos quadrados desses desvios que define o MSE, cuja raiz é o RMSE. Quanto maiores e mais numerosos os desvios, maior o erro global. O sinal reconstruído mostrado corresponde a uma interpolação linear entre *keypoints* (marcados em laranja), como produzida pelos algoritmos geométricos avaliados neste trabalho: os desvios aparecem justamente nas amostras que não foram retidas como *keypoints*.



Fonte: elaborada pelo autor.

MAPE

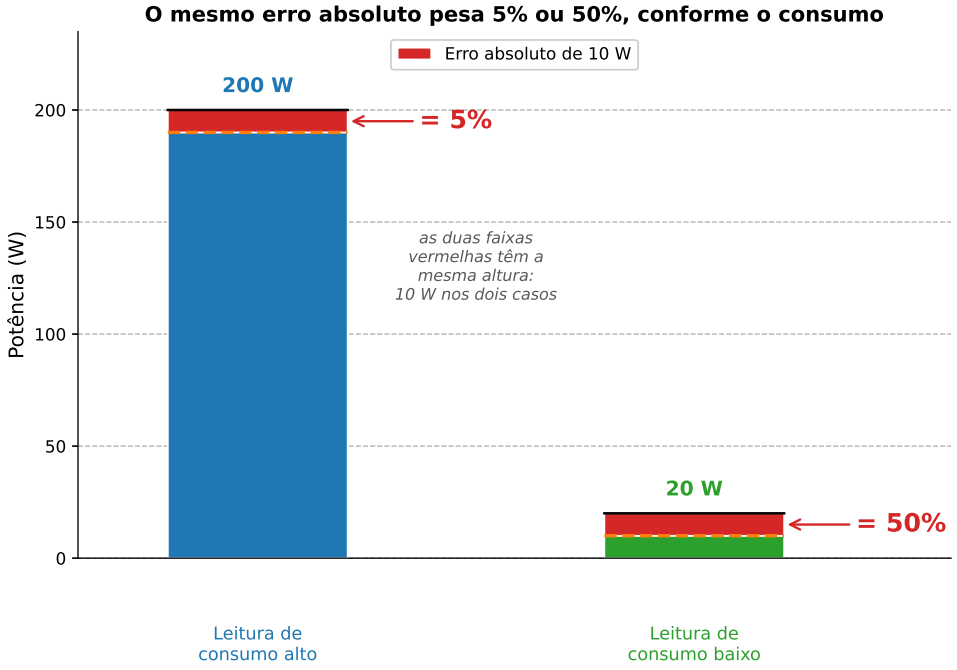
O erro percentual absoluto médio (*Mean Absolute Percentage Error*, MAPE) quantifica o erro relativo ponto a ponto, excluindo amostras nulas para evitar divisão por zero [19]:

$$\text{MAPE} = \frac{100}{N'} \sum_{\substack{i=1 \\ x_i \neq 0}}^{N} \frac{|x_i - \hat{x}_i|}{|x_i|} \tag{2.8}$$

onde N' é o número de amostras com $x_i \neq 0$. Em séries de potência com variação significativa ao longo do dia, o MAPE pondera os erros proporcionalmente ao valor instantâneo do sinal, evitando que erros em períodos de alta carga dominem a métrica e mascarem distorções em períodos de baixa carga. A Figura 2.2 ilustra esse princípio: um mesmo erro absoluto de 10 W pesa 5% numa leitura alta e 50% numa leitura baixa. Por outro lado, em instalações com períodos de consumo próximo de zero (equipamentos em *standby*, ambientes desocupados à noite), a exclusão de $x_i = 0$ exato não é suficiente: valores muito pequenos mas não nulos ($x_i \approx 0{,}1$ W) fazem o denominador amplificar erros absolutos desprezíveis em contribuições percentuais elevadas. Nesses casos, o MAPE pode ser dominado por esses pontos de baixo consumo, produzindo valores elevados mesmo quando a reconstrução é visualmente fiel

ao sinal. Os resultados do MAPE apresentados neste trabalho devem ser interpretados com essa ressalva.

Figura 2.2 — Por que o erro relativo importa: o mesmo erro absoluto tem significância muito diferente conforme o nível de consumo. Nas duas leituras, a reconstrução erra por 10 W (faixas vermelhas de altura idêntica). Numa leitura de consumo alto (200 W), esse erro representa apenas 5%; numa leitura de consumo baixo (20 W), o mesmo erro representa 50%. Uma métrica de erro absoluto como o RMSE atribui o mesmo peso aos dois casos, enquanto o MAPE, por medir o erro em relação ao valor da leitura, revela que a distorção da leitura baixa é muito mais grave. Essa é a vantagem do MAPE em séries que percorrem faixas de consumo distintas ao longo do dia: ele impede que os erros dos trechos de alta carga mascarem distorções nos trechos de baixa carga.



Fonte: elaborada pelo autor.

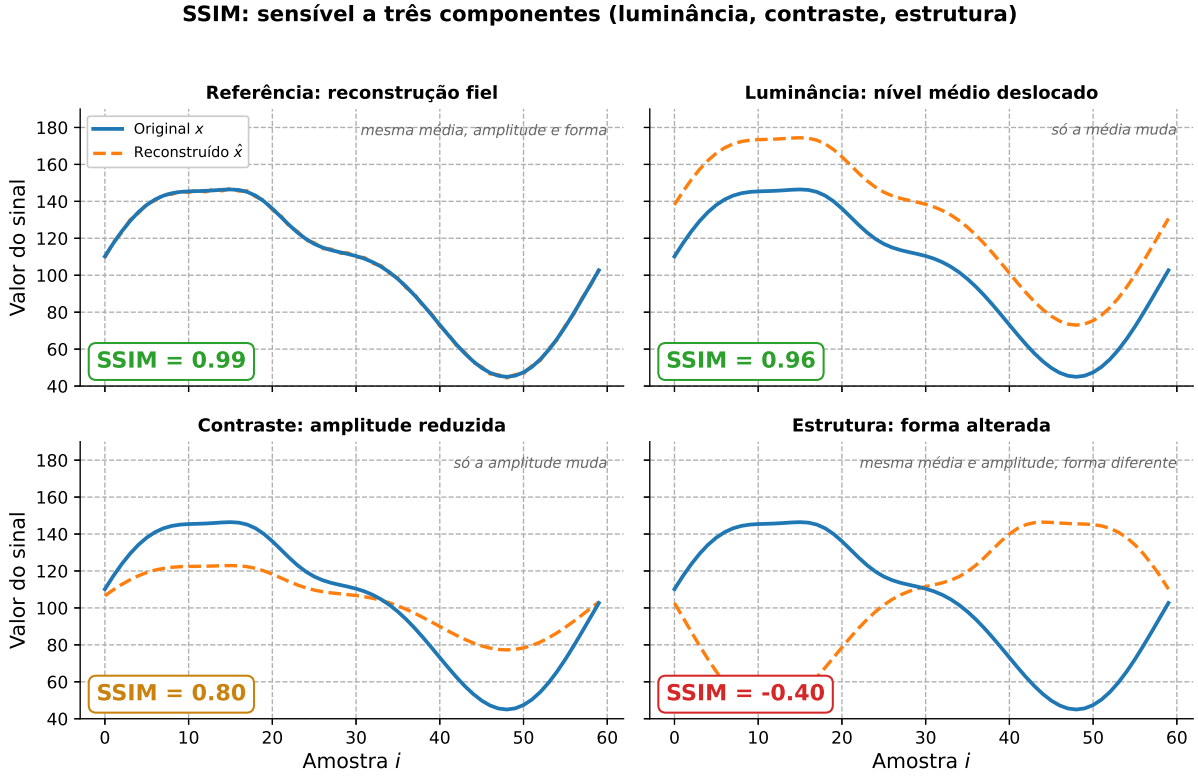
SSIM

O índice de similaridade estrutural (*Structural Similarity Index*, SSIM) foi originalmente proposto para avaliação de qualidade de imagens [48] e captura aspectos de similaridade que métricas puramente algébricas não capturam, como a preservação de padrões locais de variação. Sua formulação é baseada em janela deslizante aplicada localmente ao sinal, o que permite sua extensão direta a qualquer sinal unidimensional; o SSIM combina três componentes: luminância (comparação das médias locais), contraste (comparação dos desvios padrão locais) e estrutura (correlação entre os sinais). O resultado é um valor em $[-1, 1]$, onde 1 indica sinais idênticos.

Por capturar correlações locais e não apenas diferenças ponto a ponto, o SSIM é sensível à preservação da forma do sinal. A Figura 2.3 demonstra essa sensibilidade, degradando isoladamente cada um dos três componentes do índice. Dois sinais podem ter o mesmo RMSE mas SSIMs muito diferentes

se um deles tiver os erros distribuídos uniformemente e o outro concentrados de forma que destruam padrões locais. Em monitoramento elétrico, essa métrica é relevante para avaliar se o perfil de carga ao longo do tempo permanece reconhecível após a compressão. Na implementação adotada (biblioteca `scikit-image`), o SSIM é calculado com janela deslizante de 7 amostras (padrão para dados unidimensionais) e `data_range` igual ao intervalo de amplitude do sinal original ($\max(\mathbf{x}) - \min(\mathbf{x})$), que normaliza a fórmula para a escala dos dados.

Figura 2.3 — O que o SSIM mede, ilustrado pela degradação isolada de cada um de seus três componentes a partir do mesmo sinal original. Uma reconstrução fiel atinge SSIM próximo de 1. Deslocar apenas o nível médio (luminância), mantendo forma e amplitude, reduz pouco o índice. Reduzir apenas a amplitude das variações (contraste), mantendo o nível médio, produz queda moderada. Alterar a forma do sinal (estrutura), preservando exatamente a mesma média e a mesma amplitude, faz o SSIM despencar para valores negativos: é a correlação local que mais pesa no índice, razão pela qual o SSIM é sensível à preservação da forma do sinal e não apenas à magnitude do erro. Valores calculados com janela deslizante de 7 amostras, como na implementação adotada.



Fonte: elaborada pelo autor.

2.4.3 Fidelidade Energética

No contexto de monitoramento de potência com intervalo de medição de um minuto, cada amostra representa a potência média do intervalo em watts. A energia acumulada em watt-hora é obtida somando as amostras e convertendo a unidade de tempo:

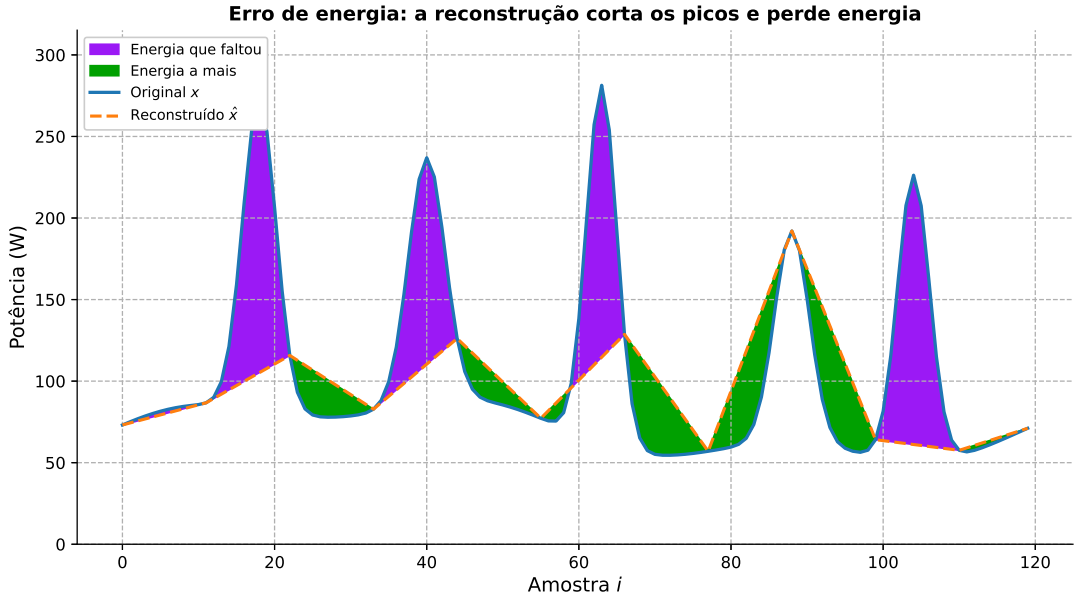
$$E(\mathbf{x}) = \frac{1}{60} \sum_{i=1}^{N} x_i \tag{2.9}$$

O erro energético (*EnergyError*) quantifica o desvio relativo da energia acumulada entre o sinal original e o reconstruído:

$$\text{EnergyError} = 100 \cdot \frac{|E(\mathbf{x}) - E(\hat{\mathbf{x}})|}{|E(\mathbf{x})|} \tag{2.10}$$

Essa métrica responde diretamente à pergunta de quanto o valor faturável de energia seria afetado pela compressão, sendo essencial em aplicações onde os dados comprimidos servirão de base para cálculo de consumo ou demanda contratada. A Figura 2.4 ilustra essa diferença entre a energia do sinal original e a do reconstruído.

Figura 2.4 — Fidelidade energética. A energia acumulada é a área sob a curva de potência. Ao cortar os picos de consumo, a reconstrução perde área (energia que faltou, violeta) e, ao passar por cima dos vales, ganha alguma (energia a mais, verde). O EnergyError é a diferença líquida entre as duas: quando a perda nos picos supera o ganho nos vales, sobra um erro de energia positivo.



Fonte: elaborada pelo autor.

2.4.4 Fidelidade de Picos

Eventos de pico em grandezas elétricas, como picos de demanda, sobretensões ou correntes de partida de equipamentos, são relevantes para diagnóstico e proteção. Métricas globais podem não capturar adequadamente a preservação desses eventos, pois os picos representam uma fração pequena das amostras

totais.

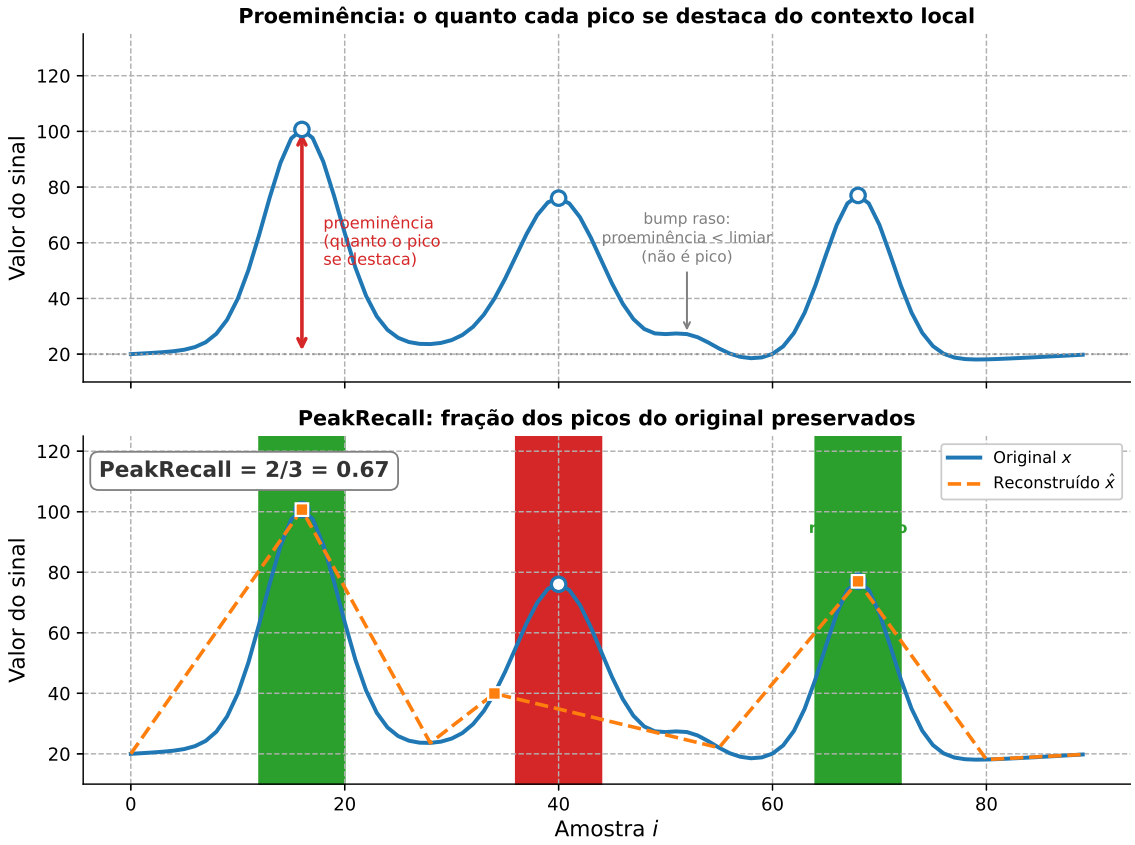
Um pico é detectado quando sua proeminência supera o limiar $\theta_p = 0{,}1 \cdot (\max(x) - \min(x))$. A proeminência de um pico, conforme definida pela função `scipy.signal.find_peaks` [47], é a diferença entre o valor do pico e o nível mais alto do vale entre o pico e o vizinho de maior amplitude em cada lado: ela quantifica o quanto o pico se destaca em relação ao contexto local, sendo mais informativa do que um limiar de amplitude absoluta. Dois picos são considerados correspondentes se a diferença de posição entre eles não excede uma tolerância Δ_p (padrão: 1% do comprimento da série).

O *PeakRecall* mede a fração dos picos do sinal original preservados na série reconstruída:

$$\text{PeakRecall} = \frac{|\{p \in \mathcal{P}_x : \exists\, \hat{p} \in \mathcal{P}_{\hat{x}},\ |p - \hat{p}| \leq \Delta_p\}|}{|\mathcal{P}_x|} \tag{2.11}$$

onde $\mathcal{P}_x$ e $\mathcal{P}_{\hat{x}}$ são os conjuntos de posições de picos detectados no sinal original e no reconstruído. Um *PeakRecall* igual a 1 indica que todos os picos foram preservados em posição. Em séries de potência ativa ou corrente, essa métrica é relevante para avaliar se o algoritmo preserva eventos de pico de demanda, que podem ser usados para cobrança de demanda contratada ou para detecção de partidas de equipamentos. A Figura 2.5 ilustra a proeminência e o cálculo do PeakRecall.

Figura 2.5 — Fidelidade de picos. Acima: a proeminência mede o quanto cada pico se destaca do contexto local, e um pico só é contado se ela supera o limiar θ_p (o bump raso é descartado). Abaixo: o PeakRecall é a fração dos picos do original recuperados na reconstrução dentro da tolerância de posição; aqui a reconstrução achata o pico central, resultando em PeakRecall de $2/3$.



Fonte: elaborada pelo autor.

O *PeakAmplitudeError* complementa o PeakRecall ao quantificar, entre os picos preservados em posição, o erro médio de amplitude:

$$\text{PeakAmplitudeError} = \frac{100}{|\mathcal{C}|} \sum_{p \in \mathcal{C}} \frac{|x_p - \hat{x}_{\hat{p}(p)}|}{|x_p|} \tag{2.12}$$

onde $\mathcal{C}$ é o conjunto de picos do original que encontraram correspondência na série reconstruída e $\hat{p}(p)$ é o pico correspondente mais próximo. Enquanto o PeakRecall responde à pergunta “os picos foram detectados?”, o PeakAmplitudeError responde a “com qual fidelidade de amplitude?”.

Capítulo 3

Algoritmos de Compressão Avaliados

3.1 Visão Geral

Este capítulo apresenta os algoritmos de compressão avaliados neste trabalho. A seleção contempla métodos baseados em simplificação geométrica de curvas (SDT e RDP) e métodos baseados em transformadas espectrais (DWT e DCT). Para cada algoritmo são descritos o fundamento teórico e o processo de implementação adotado, incluindo as decisões de projeto relativas ao custo de representação dos dados comprimidos. O ARC-SDT, proposta original deste trabalho derivada do SDT, é avaliado com o mesmo protocolo experimental, mas por tratar-se de contribuição própria é descrito em capítulo dedicado (Capítulo 4).

Antes de detalhar cada algoritmo, é necessário fixar o modelo de custo em bytes utilizado de forma consistente ao longo deste capítulo. O Capítulo 2 (Seção 2.2.3) apresentou a carga útil completa de uma leitura do dispositivo de monitoramento, com as cerca de 30 grandezas elétricas medidas a cada minuto: é essa carga útil, com todas as grandezas, que se deseja efetivamente transmitir em campo e que motiva a necessidade de compressão. Os algoritmos avaliados neste capítulo, no entanto, são testados isoladamente sobre a série temporal de potência ativa, a grandeza de maior disponibilidade nas coleções de dados utilizadas e também a mais relevante para o objetivo de monitoramento de consumo energético, conforme já justificado na Seção 5.2.1. Cada ponto dessa série é modelado como um par (*timestamp*, valor): o instante da medição e a potência ativa medida naquele instante, ambos codificados em 4 bytes (`float32` para o valor, `int32` para o *timestamp*), totalizando 8 bytes por ponto bruto, sem compressão. Essa convenção é coerente com o *timestamp* de leitura já contabilizado no exemplo de carga útil do Capítulo 2, e é aplicada de forma homogênea às quatro abordagens ao longo deste capítulo, de modo que a taxa de compressão relatada para cada um seja diretamente comparável em bytes, e não

apenas em número de pontos ou coeficientes.

3.2 Swinging Door Trending (SDT)

O *Swinging Door Trending* (SDT) é um algoritmo de compressão amostra a amostra para séries temporais, projetado para processar cada valor à medida que ele chega, sem necessidade de acesso retroativo ao histórico [15; 14]. A ideia central é manter um corredor de aproximação linear a partir do último ponto salvo: enquanto o novo ponto for compatível com uma reta que passe dentro desse corredor, ele é descartado; quando a compatibilidade é violada, o último ponto ainda válido é registrado como ponto-chave (*keypoint*) e um novo corredor é iniciado. O SDT já foi comparado diretamente ao RDP em avaliações de compressão de dados em tempo real [16], comparação também reproduzida neste trabalho.

Apesar de datar da década de 1990, o SDT segue sendo ativamente customizado na literatura recente voltada a dispositivos IoT. Arias Correa et al. propuseram uma etapa de calibração automática do parâmetro de tolerância a partir de um conjunto de treinamento [7], e o mesmo grupo, em trabalho posterior, incorporou essa calibração diretamente ao algoritmo, batizando a variante de *Training Swinging Door Trending* (TSDT) [6]. Essas customizações confirmam que a comunidade reconhece limitações práticas no SDT original e busca resolvê-las.

3.2.1 Estrutura do Corredor

O corredor é definido por duas portas imaginárias que giram (*swing*) em torno de dois pivôs fixos. Dado o ponto âncora $\texttt{ancora} = (t_0, x_0)$ e o parâmetro de tolerância `tolerancia`, os pivôs são:

$$P^+ = (t_0,\ x_0 + \texttt{tolerancia}) \qquad \text{(pivô superior)} \tag{3.1}$$

$$P^- = (t_0,\ x_0 - \texttt{tolerancia}) \qquad \text{(pivô inferior)} \tag{3.2}$$

Para cada novo ponto $p = (t, x)$ recebido, calculam-se as inclinações das retas que ligam cada pivô a p:

$$s^+ = \frac{x - (x_0 + \texttt{tolerancia})}{t - t_0} \qquad \text{(inclinação do pivô superior para } p) \tag{3.3}$$

$$s^- = \frac{x - (x_0 - \texttt{tolerancia})}{t - t_0} \qquad \text{(inclinação do pivô inferior para } p) \tag{3.4}$$

3.2.2 Condição de Validade do Corredor

Ao longo de um segmento, o algoritmo acumula os valores mais restritivos observados (denominados `max_incl_sup` e `min_incl_inf` no pseudocódigo):

$$S^+ = \max_j \; s_j^+ \qquad \text{(máxima inclinação pelo pivô superior)} \tag{3.5}$$

$$S^- = \min_j \; s_j^- \qquad \text{(mínima inclinação pelo pivô inferior)} \tag{3.6}$$

Para que exista uma reta com inclinação s tal que todos os pontos do segmento estejam dentro do corredor de tolerância $\pm$`tolerancia`, é necessário e suficiente que:

$$S^+ \leq s \leq S^-$$

O corredor é **válido** enquanto $S^+ \leq S^-$. Quando $S^+ > S^-$, as duas portas se cruzaram e não existe mais nenhuma reta que aproxime todos os pontos dentro da tolerância: o corredor foi **violado**.

3.2.3 Comportamento na Violação

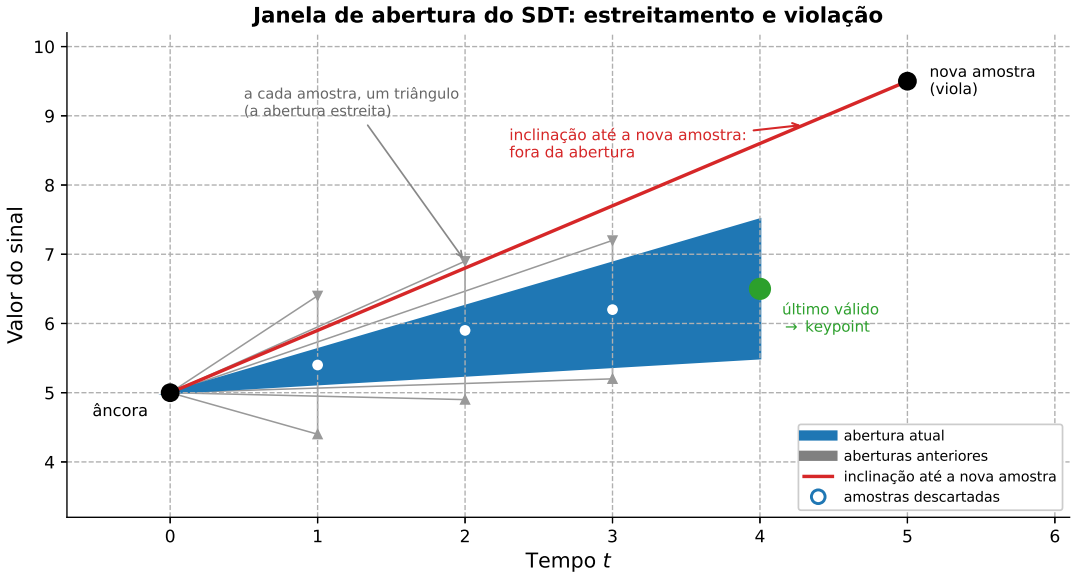
Quando a violação é detectada ao processar o ponto p:

1. O *keypoint* gerado é `ultimo_valido`, ou seja, o último ponto que havia sido processado sem violar o corredor;
2. Os pivôs são reposicionados em torno desse *keypoint*;
3. S^+ e S^- são reiniciados para $-\infty$ e $+\infty$, respectivamente;
4. O ponto p que causou a violação é reprocessado no novo corredor.

O algoritmo sempre preserva o primeiro e o último ponto da série. Para cada segmento, o *keypoint* de saída é sempre `ultimo_valido`, não o ponto violador.

A Figura 3.1 ilustra o estreitamento da abertura a cada amostra até a violação, em representação baseada na documentação do CygNet Swinging Door Compression [49].

Figura 3.1 — Janela de abertura do SDT. A cada nova amostra, o algoritmo traça as inclinações da âncora até a faixa de tolerância do ponto, formando um triângulo (a abertura daquele passo); as aberturas anteriores (em cinza) vão se estreitando, e a abertura atual é destacada em azul. Quando a inclinação até a nova amostra cai fora da abertura, o último ponto válido é emitido como *keypoint*.



Fonte: adaptado de [49].

3.2.4 Pseudocódigo

Código Fonte 3.1 — Pseudocódigo do SDT

```
funcao SDT(serie, tolerancia):
    ancora <- serie[0]
    max_incl_sup <- -infinito
    min_incl_inf <- +infinito
    ultimo_valido <- serie[0]
    keypoints <- [serie[0]]

    para cada (t, x) em serie[1:]:
        s_pos <- (x - (ancora.x + tolerancia)) / (t - ancora.t)
        s_neg <- (x - (ancora.x - tolerancia)) / (t - ancora.t)

        max_incl_sup <- max(max_incl_sup, s_pos)
        min_incl_inf <- min(min_incl_inf, s_neg)

        se max_incl_sup <= min_incl_inf:     // corredor valido
```

```
16              ultimo_valido <- (t, x)
17          senao:                                  // corredor violado
18              keypoints.adicionar(ultimo_valido)
19              ancora <- ultimo_valido
20              // reprocessa o ponto atual no novo corredor
21              s_pos <- (x - (ancora.x + tolerancia)) / (t - ancora.t)
22              s_neg <- (x - (ancora.x - tolerancia)) / (t - ancora.t)
23              max_incl_sup <- s_pos
24              min_incl_inf <- s_neg
25              ultimo_valido <- (t, x)
26
27      keypoints.adicionar(serie[-1])
28      retorna keypoints
```

Vale observar que o reprocessamento do ponto violador no novo corredor nunca gera uma segunda violação imediata: para qualquer ponto único relativo a uma âncora, $s^+ - s^- = -2\,\texttt{tolerancia}/(t - t_0) \leq 0$, ou seja, $S^+ \leq S^-$ é sempre satisfeito após o reset.

3.2.5 Reconstrução

Os pontos descartados entre dois *keypoints* consecutivos (t_0, x_0) e (t_1, x_1) são recuperados por interpolação linear:

$$\hat{x}(t) = x_0 + \frac{t - t_0}{t_1 - t_0}\,(x_1 - x_0), \quad t \in [t_0, t_1] \tag{3.7}$$

3.2.6 Custo de Representação

Cada *keypoint* preservado pelo SDT é um par (*timestamp*, valor): como os *keypoints* ficam irregularmente espaçados no tempo (o corredor pode acumular um número variável de pontos descartados entre dois *keypoints* consecutivos), o receptor não tem como inferir o instante de cada um a partir de uma taxa de amostragem fixa, ao contrário do sinal bruto. É necessário, portanto, transmitir o *timestamp* de cada *keypoint* junto com o valor, ao custo de 8 bytes por ponto mantido, exatamente a mesma estrutura de 8 bytes do dado bruto (Seção 3.1). Sendo N_c o número de *keypoints* retidos e N o número de pontos da janela original, o custo transmitido é $N_c \times 8$ bytes contra $N \times 8$ bytes do dado bruto, e a taxa de compressão real em bytes é:

$$\texttt{CR} = 100 \times \left(1 - \frac{N_c \times 8}{N \times 8}\right) = 100 \times \left(1 - \frac{N_c}{N}\right) \tag{3.8}$$

O fator de 8 bytes cancela-se entre numerador e denominador porque *keypoint* e ponto bruto compartilham a mesma estrutura de representação (valor e *timestamp*). Por esse motivo, a razão N_c/N já é, sem qualquer ajuste adicional, uma medida de compressão em bytes tão precisa quanto a de qualquer algoritmo que calcule seu custo diretamente em bytes.

3.2.7 Parâmetros

A Tabela 3.1 resume o único parâmetro de configuração do SDT e seu efeito sobre a taxa de compressão resultante.

Tabela 3.1 — Parâmetros do SDT

Parâmetro	**Efeito**
`tolerancia` (ε)	Tolerância vertical em unidades do sinal. Maior `tolerancia` $\rightarrow$ corredor mais largo $\rightarrow$ menos *keypoints* $\rightarrow$ maior taxa de compressão

Fonte: elaborada pelo autor.

3.3 Ramer-Douglas-Peucker (RDP)

O algoritmo Ramer-Douglas-Peucker (RDP) foi proposto independentemente por Ramer [43] e por Douglas e Peucker [21]. Ao contrário do SDT, é um algoritmo em lote: opera recursivamente sobre toda a série disponível para identificar os pontos geometricamente mais relevantes. Apesar de datar do início da década de 1970, o RDP segue sendo muito usado na literatura e aplicado em domínios variados, de compressão de trajetórias GPS e de embarcações a classificação de séries temporais [30], o que evidencia sua robustez e generalidade mesmo décadas após sua proposição original. No domínio específico de compressão de dados em tempo real, o RDP já foi comparado diretamente ao SDT [16], o que reforça sua relevância como termo de comparação para a avaliação realizada neste trabalho.

3.3.1 Distância Perpendicular

O critério de relevância do RDP é a **distância perpendicular** de um ponto à reta formada pelos extremos do segmento atual. Dado um ponto $P = (x, y)$ e uma reta definida por $A = (x_1, y_1)$ e $B = (x_2, y_2)$, a distância perpendicular é:

$$d(P, AB) = \frac{\left|(y_2 - y_1)\, x - (x_2 - x_1)\, y + x_2 y_1 - y_2 x_1\right|}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} \tag{3.9}$$

Quando $A = B$ (segmento degenerado), adota-se $d = |y - y_1|$.

3.3.2 Algoritmo Recursivo

O RDP recebe uma sequência de pontos $P = [p_0, p_1, \ldots, p_{N-1}]$ e um limiar $\xi > 0$ (na implementação: parâmetro `epsilon`):

Código Fonte 3.2 — Pseudocódigo do RDP

```
funcao RDP(pts):
    se len(pts) < 3:
        retorna pts

    a <- pts[0]; b <- pts[-1]

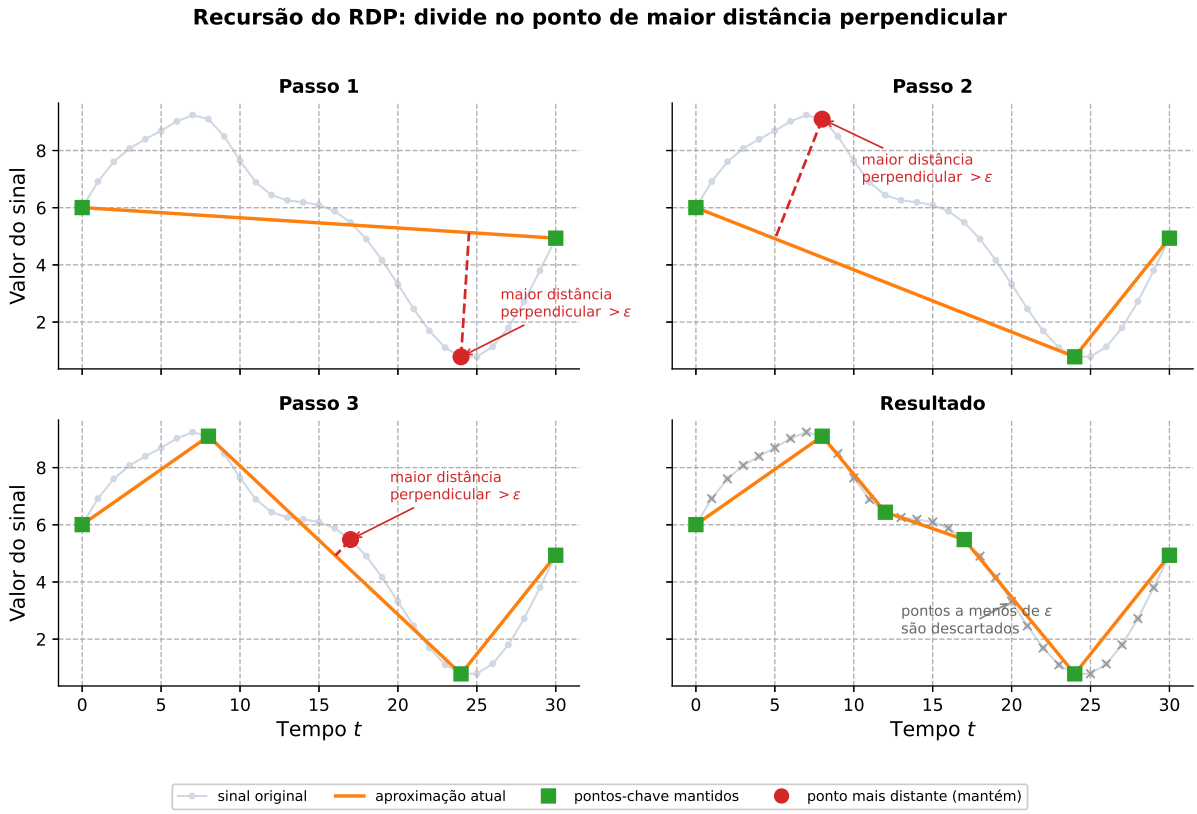
    max_d <- -1; idx <- -1
    para i de 1 ate len(pts)-2:
        d <- distancia_perpendicular(pts[i], a, b)
        se d > max_d: max_d <- d; idx <- i

    se max_d > epsilon:
        esq <- RDP(pts[:idx+1])
        dir <- RDP(pts[idx:])
        retorna esq[:-1] + dir

    retorna [a, b]
```

Os pontos inicial e final são sempre preservados. O ponto de maior distância perpendicular divide o segmento recursivamente; quando nenhum ponto supera ξ, o segmento inteiro é representado apenas pelos seus extremos. A Figura 3.2 ilustra essa recursão passo a passo.

Figura 3.2 — Recursão do RDP. A cada passo, entre os *keypoints* atuais busca-se o ponto de maior distância perpendicular à reta do segmento (tracejado vermelho); se essa distância supera o limiar ξ, o ponto é mantido e o segmento é dividido. No resultado, os pontos que ficam a menos de ξ de alguma reta mantida são descartados.



Fonte: elaborada pelo autor.

3.3.3 Propriedades

Garantia de erro: para qualquer ponto descartado, sua distância perpendicular à reta entre os *keypoints* vizinhos é $\leq \xi$. A garantia é geométrica (distância perpendicular), não em desvio vertical puro como no SDT.

Complexidade:

- Caso médio: $\mathcal{O}(N \log N)$, quando o ponto de maior distância divide o segmento de forma equilibrada;
- Pior caso: $\mathcal{O}(N^2)$, quando o ponto escolhido está sistematicamente próximo a uma extremidade.

Há uma consequência direta do critério de seleção do RDP que é particularmente relevante para o caso de uso desta pesquisa: como o ponto escolhido em cada divisão recursiva é sempre aquele de maior distância perpendicular à reta do segmento, o algoritmo tende a preservar exatamente os pontos que mais

se destacam localmente, ou seja, os picos e vales do sinal. Essa característica não é por acaso; um estudo comparativo entre algoritmos de simplificação de linha mostrou que o Douglas-Peucker obteve o menor erro de reconstrução entre os métodos avaliados [33], resultado consistente com sua tendência a manter os pontos de maior desvio local em vez de descartá-los. Para o monitoramento de grandezas elétricas, essa propriedade é desejável: picos de demanda, sobretensões e transitórios são justamente os eventos que métricas de fidelidade de picos buscam preservar, o que torna o RDP um candidato natural para essa dimensão de avaliação.

3.3.4 Reconstrução

A reconstrução é feita por interpolação linear entre os *keypoints*, da mesma forma que no SDT.

3.3.5 Processo de Implementação

O RDP opera sobre a série completa. Como a recursão pode atingir profundidade da ordem de N em casos degenerados, o limite de recursão do Python é ajustado para $\max(\text{limite atual}, 2N)$ antes da execução. Após a compressão, a reconstrução por interpolação linear é realizada fora do bloco monitorado.

3.3.6 Custo de Representação

O RDP produz *keypoints* da mesma natureza que o SDT: pares (*timestamp*, valor) irregularmente espaçados no tempo, cuja reconstrução também depende de interpolação linear entre pontos vizinhos. Aplica-se, portanto, exatamente o mesmo modelo de custo descrito na Seção 3.2.6: cada *keypoint* custa 8 bytes (valor mais *timestamp*), o mesmo custo por ponto do dado bruto, de modo que $\texttt{CR} = 100 \times (1 - N_c/N)$ já expressa a taxa de compressão real em bytes, sem necessidade de fator de correção adicional.

3.3.7 Parâmetros

A Tabela 3.2 resume o único parâmetro de configuração do RDP e seu efeito sobre a taxa de compressão resultante.

Tabela 3.2 — Parâmetros do RDP

Parâmetro	Efeito
`epsilon` (ξ)	Distância perpendicular máxima tolerada. Menor ξ $\rightarrow$ mais pontos mantidos $\rightarrow$ menor taxa de compressão

Fonte: elaborada pelo autor.

3.4 Transformada Wavelet Discreta (DWT)

3.4.1 Análise Multirresolução

A Transformada Wavelet Discreta (DWT) decompõe um sinal em camadas de resolução diferentes, separando componentes lentas de variações rápidas. A ideia central é a análise de multirresolução (MRA), introduzida por Mallat [37]. Em uma série de consumo elétrico, a escala grossa captura a tendência ao longo do período; as escalas finas capturam transitórios e picos localizados. Coeficientes de detalhe com magnitude pequena correspondem a variações de baixa relevância e podem ser descartados sem comprometer a reconstrução.

3.4.2 Banco de Filtros

Na prática, a DWT é implementada por um banco de filtros. Dado um sinal discreto $x[n]$ de comprimento N, um passo aplica dois filtros FIR (*Finite Impulse Response*) em paralelo:

$$x[n] \longrightarrow \begin{cases} h[n] \text{ (passa-baixa)} \longrightarrow \downarrow 2 \longrightarrow cA \quad \text{(aproximação)} \\ g[n] \text{ (passa-alta)} \longrightarrow \downarrow 2 \longrightarrow cD \quad \text{(detalhe)} \end{cases}$$

Os filtros satisfazem a relação de conjugado em quadratura, garantindo cobertura completa do espectro e a possibilidade de reconstrução perfeita. A decomposição é aplicada recursivamente à sub-banda de aproximação até o nível L configurado, produzindo o conjunto $\{cA_L,\, cD_L,\, cD_{L-1},\, \ldots,\, cD_1\}$.

3.4.3 Famílias de Wavelets

Konecny, Borova e Prauzek [31] propuseram um sistema de compressão wavelet para transmissão de dados de estações de monitoramento ambiental com interface IoT de baixa largura de banda, cenário próximo ao caso de uso considerado nesta dissertação; o mesmo grupo de pesquisa também avaliou compressão wavelet para monitoramento ambiental via LoRaWAN [13]. Com base nesses trabalhos, este experimento adota as mesmas três famílias de wavelets: `haar` e `sym2`, originadas da construção de Daubechies, e a família biortogonal `bior5.5`, introduzida por Cohen, Daubechies e Feauveau. Adotar uma configuração já usada em trabalhos correlatos, em vez de otimizar a escolha para o caso de uso específico deste trabalho, permite avaliar o comportamento dessas famílias tal como empregadas na literatura de sistemas embarcados de baixa largura de banda. A Tabela 3.3 compara as propriedades das três famílias adotadas.

Tabela 3.3 — Famílias de wavelets utilizadas no experimento

Propriedade	`haar`	`sym2`	`bior5.5`
Ortogonal	Sim	Sim	Não
Momentos nulos	1	2	5 (cada filtro)
Suporte (amostras)	2	4	12 (análise e síntese)
Simetria	Assimétrica	Quase simétrica	Simétrica
Fase	Não linear	Quase linear	Exatamente linear
Filtro decomp. = reconstr.	Sim	Sim	Não
Preserva energia (Parseval)	Sim	Sim	Não diretamente

Fonte: elaborada pelo autor.

Momentos nulos de ordem M indicam que a wavelet é ortogonal a polinômios de grau até $M - 1$: regiões do sinal com variação polinomial produzem coeficientes de detalhe próximos de zero, maximizando o número de coeficientes descartáveis. Como as três famílias adotadas têm poucos momentos nulos (entre 1 e 5) e suporte curto quando comparadas a famílias de ordem mais alta, sua capacidade de concentrar energia em poucos coeficientes é limitada; essa limitação se reflete diretamente nos resultados de fidelidade apresentados no Capítulo 6.

O nível de decomposição, por sua vez, foi fixado em 3, valor que corresponde ao maior nível avaliado

por Konecny, Borova e Prauzek [31], que testaram níveis 1, 2 e 3 para cada uma das três famílias adotadas. Sob o modelo de custo em bytes adotado neste trabalho (Seção 3.4.5), o nível de decomposição, a família da wavelet e o tamanho da janela de processamento são parâmetros combinados a priori entre emissor e receptor, e portanto não são transmitidos por mensagem; o overhead fixo de metadados não depende do nível de decomposição. A escolha de nível 3 decorre, assim, exclusivamente do alinhamento com a configuração de referência da literatura correlata, sem uma restrição concorrente de orçamento de bytes.

É importante avaliar como esse modelo de custo se comporta na menor janela avaliada neste trabalho (15 pontos), caso em que o orçamento de bytes é mais apertado. Para $N = 15$, o tamanho original do dado bruto é $15 \times 8 = 120$ bytes e o overhead fixo é 12 bytes, valores que não dependem do nível de decomposição nem da família adotada. Para taxas de compressão alvo de até 70%, o orçamento disponível ainda comporta mais de um coeficiente e a taxa de compressão real acompanha de perto o alvo: por exemplo, um alvo de 70% resulta em $K = 3$ coeficientes mantidos e CR real de exatamente 70%. A partir de 80%, o orçamento remanescente após o overhead cai abaixo de 8 bytes, e o piso $K \geq 1$ do algoritmo (a compressão nunca descarta a totalidade dos coeficientes) satura o CR real em $83{,}3\%$ tanto para o alvo de 80% quanto para o de 90%: o algoritmo passa a superestimar levemente o primeiro e a não alcançar o segundo, mas em nenhum caso a taxa de compressão real se torna nula ou negativa nessa janela.

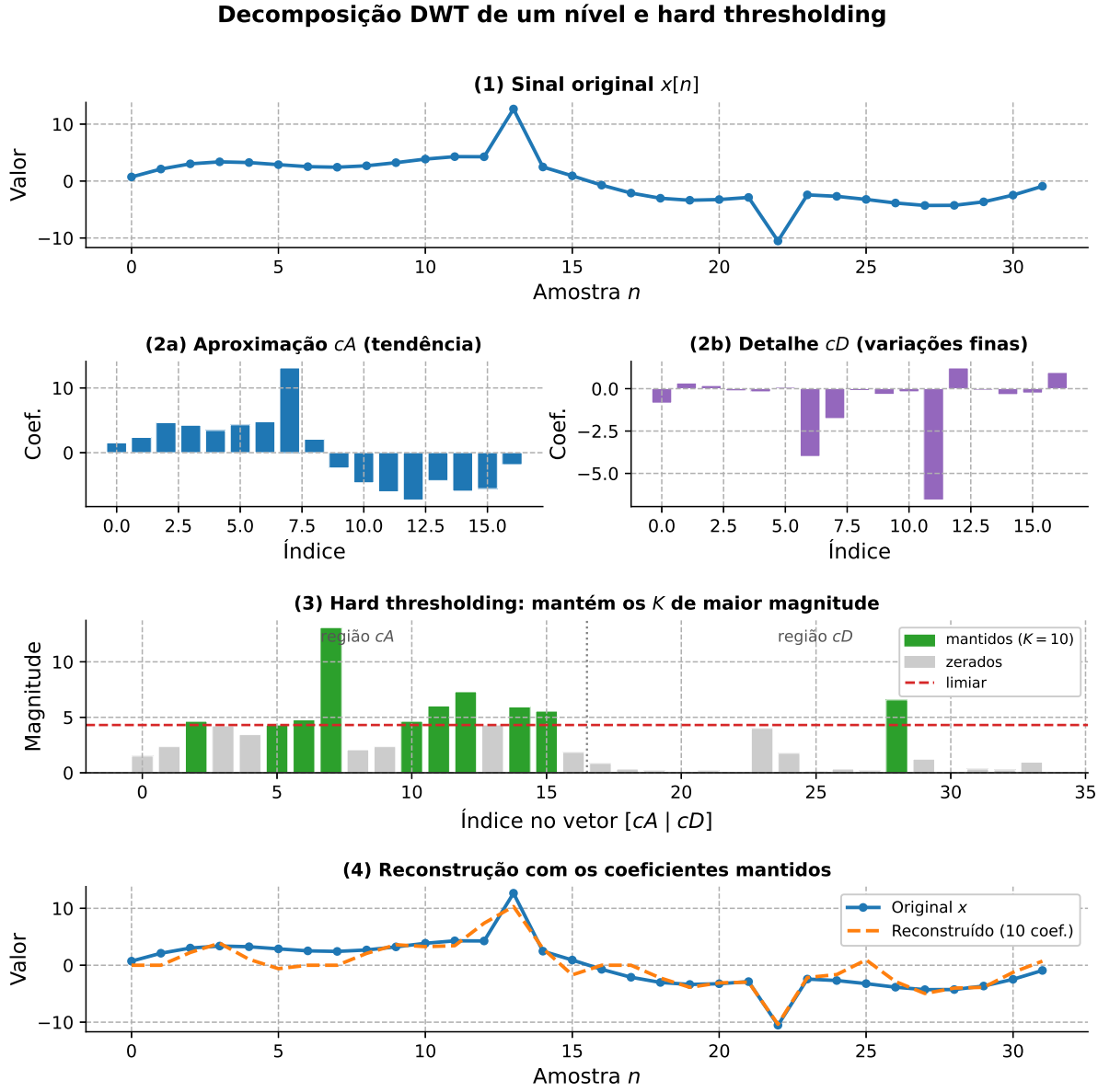
3.4.4 Compressão por Hard Thresholding

Após a decomposição, todos os coeficientes são organizados em um único vetor (*flat array*). O método de compressão adotado é o *hard thresholding*: mantém-se exatamente os K coeficientes de maior magnitude, zerando os demais:

$$\hat{c}[i] = \begin{cases} c[i], & \text{se } |c[i]| \text{ está entre os } K \text{ maiores} \\ 0, & \text{caso contrário} \end{cases} \tag{3.10}$$

Para as famílias ortogonais (haar e sym2), a preservação de energia da DWT (teorema de Parseval) garante que o erro de reconstrução seja exatamente a energia dos coeficientes descartados. Para a família biortogonal bior5.5, os filtros de análise e síntese são diferentes, de modo que a relação de Parseval não se aplica diretamente; ainda assim, o *hard thresholding* por magnitude permanece uma estratégia válida para concentrar a energia nos coeficientes mantidos. A Figura 3.3 ilustra a decomposição de um nível, o *hard thresholding* e a reconstrução resultante.

Figura 3.3 — Decomposição DWT de um nível e *hard thresholding*. O sinal (1) é decomposto em aproximação cA (2a), que retém a tendência, e detalhe cD (2b), que retém as variações finas e destaca os dois eventos abruptos. O *hard thresholding* (3) mantém os K coeficientes de maior magnitude do vetor $[cA \mid cD]$ (verde) e zera os demais (cinza). A reconstrução (4), a partir apenas dos coeficientes mantidos, preserva a tendência e os eventos significativos, descartando os detalhes pequenos.



Fonte: elaborada pelo autor.

3.4.5 Processo de Implementação

O processo adotado opera da seguinte forma:

1. A série é convertida para ponto flutuante de 32 bits e normalizada para o intervalo $[0, 1]$, armazenando $x_{\min}$ e $x_{\max}$;

2. Aplica-se a decomposição em L níveis com a família de wavelet e o nível configurados;

3. Os coeficientes são linearizados em um único vetor, acompanhado da estrutura que mapeia esse vetor de volta aos *sub-arrays* de cada nível. Essa estrutura não precisa ser transmitida: é inteiramente determinística a partir de N, do nível e da família, de modo que o receptor pode recalculá-la localmente sem receber nenhum dado adicional;

4. Calcula-se o orçamento de bytes disponível para os coeficientes: o tamanho original do dado bruto é $N \times 8$ bytes (par *timestamp* mais valor por ponto, Seção 3.1); o tamanho alvo é $(1 - \texttt{cr}/100)$ vezes o original; subtrai-se um overhead fixo de $3 \times 4 = 12$ bytes, cobrindo apenas os metadados $x_{\min}$, $x_{\max}$ e o *timestamp* inicial da janela. N, o nível de decomposição e a família da wavelet são parâmetros combinados a priori entre emissor e receptor (Seção 3.2.6 discute o mesmo princípio para o SDT) e, portanto, não entram nesse orçamento;

5. Como os coeficientes mantidos são esparsos (posições arbitrárias no vetor, selecionadas por magnitude), cada um exige o envio do valor **e** do índice original, com custo de 8 bytes por coeficiente. O número de coeficientes mantidos é $K = \max\big(1,\ \lfloor(\text{orçamento disponível})/8\rfloor\big)$;

6. Os K coeficientes de maior magnitude são selecionados por ordenação, e os demais são zerados;

7. A reconstrução recalcula a estrutura de mapeamento localmente e aplica a transformada inversa sobre os coeficientes com os demais zerados, seguida da reversão da normalização e truncamento para N amostras.

A taxa de compressão real é calculada após a execução: $\texttt{CR} = 100 \times (1 - (K \times 8 + 12)/(N \times 8))$.

3.4.6 Parâmetros

A Tabela 3.4 reúne os parâmetros de configuração do compressor baseado em DWT.

Tabela 3.4 — Parâmetros do compressor Wavelet

Parâmetro	Efeito
`wavelet`	Família wavelet: `haar`, `sym2` ou `bior5.5`
`level`	Profundidade da decomposição, fixada em 3 neste trabalho (ver Seção 3.4.3). Mais níveis → maior separação de escalas temporais; não afeta o overhead de metadados, que é fixo e independente do nível
`cr`	Taxa de compressão alvo em %. Controla o número de coeficientes mantidos via orçamento de bytes

Fonte: elaborada pelo autor.

3.5 Transformada Discreta do Cosseno (DCT)

3.5.1 Decomposição em Ondas de Cosseno

A Transformada Discreta do Cosseno (DCT-II), proposta por Ahmed, Natarajan e Rao [4], expressa um sinal como soma ponderada de ondas cosseno de diferentes frequências. O coeficiente de frequência zero corresponde ao valor médio do sinal (componente DC); coeficientes de frequências mais altas descrevem oscilações progressivamente mais rápidas. Em sinais suaves, como curvas de consumo ao longo de um dia, a energia tende a se concentrar nos primeiros coeficientes, o que motiva historicamente o descarte das componentes de frequência mais alta: é o mesmo princípio que fundamenta a compressão JPEG em imagens e MP3 em áudio. A DCT também é aplicada fora do domínio de imagem, áudio e vídeo: Yao, Zhang e Sun a utilizaram na compressão de sinais de comunicação digital em processamento de correlação de sinais, alcançando razões de compressão de pelo menos 20 vezes sem comprometer a acurácia da correlação [50].

3.5.2 Definição Formal

Para um sinal $x[n]$, $n = 0, \ldots, N-1$, com normalização ortogonal:

$$X[k] = \alpha[k] \sum_{n=0}^{N-1} x[n] \cos\left(\frac{\pi k(2n+1)}{2N}\right), \quad \alpha[k] = \begin{cases} \sqrt{1/N}, & k = 0 \\ \sqrt{2/N}, & k \geq 1 \end{cases} \tag{3.11}$$

A transformada inversa (IDCT) tem a mesma forma. A DCT ortogonal preserva a energia do sinal (Parseval): $\sum_n x[n]^2 = \sum_k X[k]^2$.

3.5.3 Compressão por Hard Thresholding

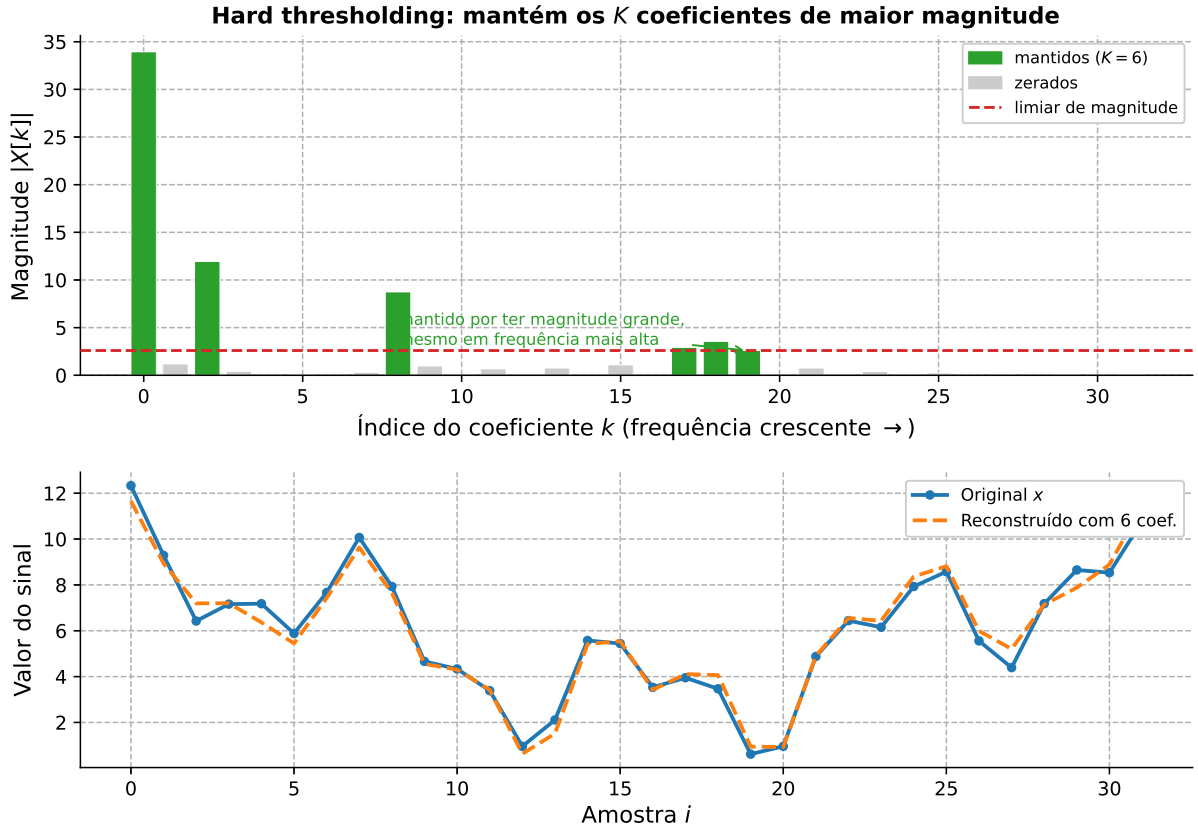
Apesar de a energia tender a se concentrar nas componentes de frequência mais baixa em sinais suaves, nada garante que essa tendência se verifique em toda janela: sinais com transientes ou picos de demanda abruptos podem concentrar energia relevante em componentes de frequência mais alta. Como a DCT também é uma transformada ortogonal (a normalização de $\alpha[k]$ garante $\sum_n x[n]^2 = \sum_k X[k]^2$, por Parseval), vale para ela o mesmo resultado de melhor aproximação em K termos já discutido para a DWT (Seção 3.4.4): a escolha que minimiza o erro quadrático de reconstrução para um orçamento fixo de K coeficientes é sempre manter os de maior magnitude, não necessariamente os de frequência mais baixa. A estratégia de compressão adotada é, portanto, o mesmo *hard thresholding* usado na DWT, aplicado aos coeficientes da DCT:

$$\hat{X}[k] = \begin{cases} X[k], & \text{se } |X[k]| \text{ está entre os } K \text{ maiores} \\ 0, & \text{caso contrário} \end{cases} \tag{3.12}$$

O erro de reconstrução é a energia dos coeficientes descartados, que por construção é a menor possível para o orçamento de K coeficientes disponível. Como os coeficientes mantidos podem incluir componentes de frequência alta e excluir componentes de frequência baixa, o fenômeno de Gibbs (oscilações artificiais ao redor de transições abruptas) permanece possível, mas não é a consequência determinística que seria de um truncamento sequencial: um coeficiente de alta frequência que descreva um transiente relevante pode ser preservado se sua magnitude superar a de coeficientes de frequência mais baixa.

A Figura 3.4 ilustra essa seleção por magnitude.

Figura 3.4 — *Hard thresholding* por magnitude. No espectro de coeficientes (aqui da DCT), mantêm-se os K de maior magnitude (verde, acima do limiar) e zeram-se os demais (cinza). A seleção é pela magnitude, e não pela posição: um coeficiente de frequência mais alta pode ser mantido se sua magnitude for suficientemente grande. O painel inferior compara o sinal original com a reconstrução usando apenas os K coeficientes mantidos. O mesmo mecanismo se aplica à DWT.



Fonte: elaborada pelo autor.

Uma consequência prática desse critério é que o coeficiente $k = 0$, que codifica a média do sinal ($X[0] = \sqrt{1/N}\sum_{n=0}^{N-1} x[n]$), não tem preservação garantida: se sua magnitude não estiver entre as K maiores, ele é descartado como qualquer outro. Num truncamento sequencial, a média do sinal reconstruído seria necessariamente igual à média original e o EnergyError seria estruturalmente nulo para qualquer taxa de compressão; sob *hard thresholding*, essa garantia não existe: a fidelidade energética da DCT depende de quão bem a energia do sinal se concentra nos coeficientes de maior magnitude, do mesmo modo que ocorre para a DWT.

3.5.4 Processo de Implementação

O processo adotado é análogo ao da DWT, incluindo a estrutura dos coeficientes transmitidos:

1. A série é convertida para ponto flutuante de 32 bits e normalizada para $[0, 1]$, armazenando $x_{\min}$

e $x_{\max}$;

2. Aplica-se a DCT ortogonal com normalização ortonormal;

3. Calcula-se o orçamento de bytes disponível para os coeficientes: o tamanho original do dado bruto é $N \times 8$ bytes (par *timestamp* mais valor por ponto, Seção 3.1); o tamanho alvo é $(1 - \texttt{cr}/100)$ vezes o original; subtrai-se um overhead fixo de $3 \times 4 = 12$ bytes, cobrindo apenas os metadados $x_{\min}$, $x_{\max}$ e o *timestamp* inicial da janela. N é um parâmetro combinado a priori entre emissor e receptor e não entra nesse orçamento;

4. Como os coeficientes mantidos são os de maior magnitude (posições arbitrárias, não necessariamente as primeiras), cada um exige o envio do valor **e** do índice original, com custo de 8 bytes por coeficiente. O número de coeficientes mantidos é $K = \max\big(1, \lfloor(\text{orçamento disponível})/8\rfloor\big)$, selecionados por ordenação;

5. Zeram-se os demais coeficientes e aplica-se a IDCT;

6. Reverte-se a normalização e o resultado é truncado para N amostras.

A taxa de compressão real é calculada após a execução: $\texttt{CR} = 100 \times (1 - (K \times 8 + 12)/(N \times 8))$, a mesma fórmula usada para a DWT (Seção 3.4.5). Sob o critério de seleção por magnitude, DCT e DWT compartilham exatamente o mesmo modelo de custo por coeficiente; a diferença entre as duas não está no custo de representação, mas exclusivamente na base de transformação (cosseno de frequência global, no caso da DCT, contra multirresolução tempo-frequência, no caso da DWT) e nas consequências dessa base sobre a qualidade de reconstrução.

3.5.5 Parâmetros

A Tabela 3.5 reúne o parâmetro de configuração do compressor baseado em DCT.

Tabela 3.5 — Parâmetros do compressor DCT

Parâmetro	**Efeito**
`cr`	Taxa de compressão alvo em %. Controla K via orçamento de bytes

Fonte: elaborada pelo autor.

3.6 Considerações Comparativas

As quatro abordagens apresentadas diferem em três dimensões estruturais que determinam seu perfil de uso em dispositivos embarcados.

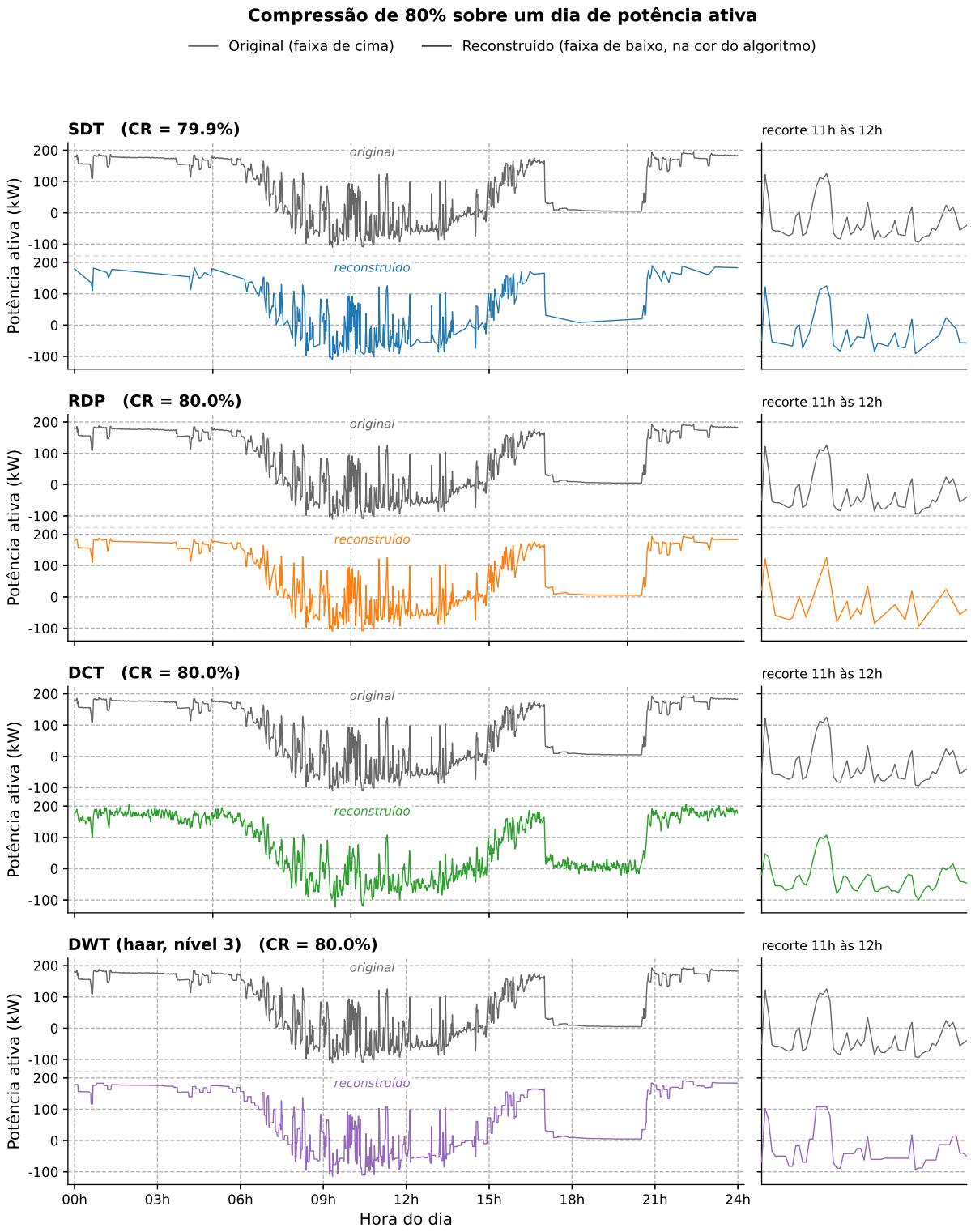
Modo de operação. O SDT processa o sinal amostra a amostra, sem acesso retroativo ao histórico, com custo de memória constante. RDP, DCT e DWT operam em lote: exigem que toda a janela esteja disponível em memória antes de iniciar a compressão.

Critério de descarte. SDT e RDP descartam amostras com base em desvio geométrico em relação a uma reta de aproximação, com reconstrução por interpolação linear. A diferença entre eles está na métrica de desvio: o SDT usa desvio vertical; o RDP usa distância perpendicular. DCT e DWT transformam o sinal para um domínio de coeficientes e descartam os de menor magnitude. A DWT decompõe o sinal em múltiplas escalas tempo-frequência; a DCT projeta em componentes de cosseno de frequência global.

Custo de representação. As quatro abordagens convergem para o mesmo custo unitário de 8 bytes por elemento mantido, ainda que por razões distintas: SDT e RDP transmitem *keypoints* irregulares no tempo, cujo instante precisa acompanhar o valor (Seção 3.2.6); DCT e DWT selecionam coeficientes por magnitude, esparsos dentro do vetor transformado, cujo índice precisa acompanhar o valor (Seção 3.4.5). A diferença entre os algoritmos está, portanto, em quantos elementos cada um precisa manter para atingir uma dada taxa de compressão, e não no custo unitário de cada elemento.

Para ilustrar essas diferenças de forma concreta, a Figura 3.5 aplica cada abordagem a um dia de potência ativa da coleção privada, todas ajustadas para 80% de compressão: o SDT e o RDP por busca binária na tolerância de erro, a DCT e a DWT pela definição direta do orçamento de coeficientes. Os algoritmos geométricos preservam a forma geral do sinal e acompanham bem as variações lentas, mas suavizam os picos abruptos ao interpolar entre os pontos mantidos; os espectrais representam o sinal pelas componentes de maior energia e atenuam os extremos. O impacto quantitativo dessas diferenças é avaliado experimentalmente no Capítulo 6.

Figura 3.5 — Resultado da compressão de 80% sobre série de potência ativa da coleção privada (21/08/2025) pelas quatro abordagens avaliadas. Em cada painel, o original ocupa a faixa superior, em cinza, e o reconstruído a inferior, na cor do algoritmo, ambas na mesma escala; à direita, recorte ampliado de 11h às 12h, a hora de maior erro do dia. A DWT é ilustrada pela `haar`, uma das três famílias avaliadas.



Fonte: elaborada pelo autor.

Capítulo 4

ARC-SDT: Controle Adaptativo da Taxa de Compressão

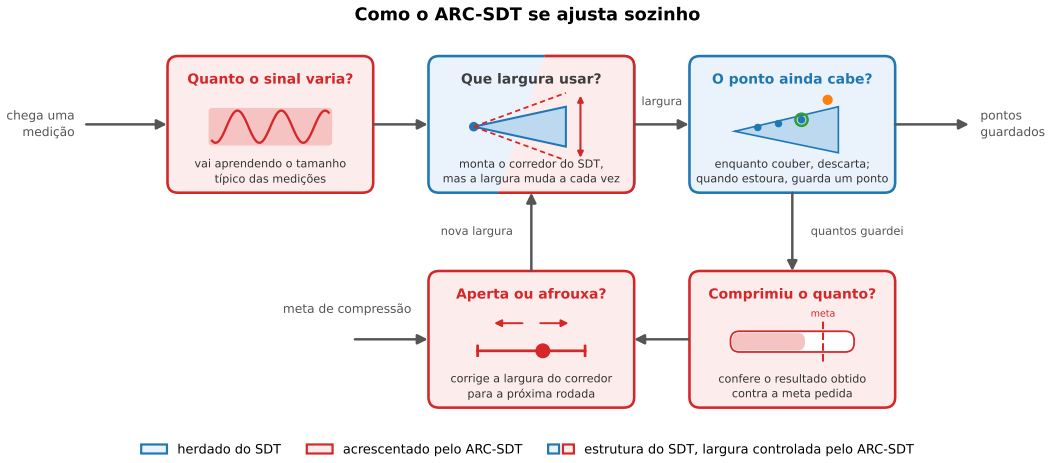
O SDT, descrito no Capítulo 3, reúne propriedades excepcionais para uso em dispositivos embarcados: processa o sinal amostra a amostra sem armazenar histórico, opera em tempo e memória constantes por ponto, garante que o erro máximo de reconstrução nunca exceda o parâmetro de tolerância configurado e, do ponto de vista legal, sua patente original [15] expirou em 2005, permitindo uso livre sem restrições de propriedade intelectual. No entanto, o SDT em sua formulação original não oferece controle sobre a taxa de compressão produzida. O único parâmetro de configuração, a tolerância absoluta, determina indiretamente a taxa de compressão por meio de uma relação que depende das características específicas do sinal: sua variabilidade, magnitude e estrutura temporal. Em sinais de consumo de energia elétrica, cujo perfil varia significativamente entre instalações, dias da semana e horários, a mesma tolerância pode produzir taxas de compressão radicalmente diferentes de um dia para outro ou de um medidor para outro. Isso torna inviável o uso do SDT em cenários com restrições estritas de carga útil, como transmissão por LoRaWAN, onde o volume de dados precisa ser previsível e controlado.

Diante desse diagnóstico, este capítulo apresenta o ARC-SDT (*Adaptive Real-time Controlled* SDT), uma adaptação do SDT que resolve essa limitação por meio de dois mecanismos combinados: a reformulação da tolerância como fração do RMS do sinal, calculada de forma incremental em uma variável chamada `ms`, e um controlador PID que ajusta essa fração (`tol_rel`) dinamicamente para que a taxa de compressão resultante se aproxime de um alvo configurável. A análise empírica dos ganhos, descrita na Seção 4.3, mostra que o termo derivativo não produz efeito mensurável neste sistema, simplificando o controlador para a forma PI na implementação final.

A Figura 4.1 resume esses dois mecanismos e como eles se encaixam no SDT original: as etapas

em azul são herdadas do algoritmo de base, as em vermelho são acrescentadas por esta proposta, e a montagem do corredor aparece dividida entre as duas cores, por manter a geometria do SDT sob uma largura que passa a ser recalculada a cada amostra.

Figura 4.1 — Visão geral do ARC-SDT como ciclo de ajuste. Em azul, a etapa herdada do SDT; em vermelho, as etapas acrescentadas por este trabalho; em cores divididas, a montagem do corredor, cuja geometria vem do SDT e cuja largura passa a ser controlada.



Fonte: elaborada pelo autor.

4.1 Invariância de Escala do SDT

O ponto de partida para a proposta é uma propriedade fundamental do SDT que, embora não seja explicitada na literatura original, tem implicação direta na forma como a tolerância deve ser definida: **o SDT é exatamente invariante à escala do sinal**.

4.1.1 Demonstração

Considere o SDT operando sobre uma série temporal $y = \{(t_i, x_i)\}$ com tolerância ε. Em um dado segmento com ponto âncora (t_0, x_0), a inclinação pelo pivô superior para um ponto (t, x) é:

$$s^+ = \frac{x - (x_0 + \varepsilon)}{t - t_0} \tag{4.1}$$

e a inclinação pelo pivô inferior é:

$$s^- = \frac{x - (x_0 - \varepsilon)}{t - t_0} \tag{4.2}$$

A violação do corredor ocorre quando $S^+ > S^-$, onde $S^+ = \max_j s_j^+$ e $S^- = \min_j s_j^-$ sobre todos os pontos processados no segmento atual.

Agora considere a série escalada $\tilde{y} = \{(t_i,\ kx_i)\}$, para algum fator $k > 0$, processada com tolerância $\tilde{\varepsilon} = k\varepsilon$. Para o ponto âncora (t_0, kx_0) e o ponto (t, kx):

$$\tilde{s}^+ = \frac{kx - (kx_0 + k\varepsilon)}{t - t_0} = k \cdot \frac{x - (x_0 + \varepsilon)}{t - t_0} = k \cdot s^+ \tag{4.3}$$

$$\tilde{s}^- = \frac{kx - (kx_0 - k\varepsilon)}{t - t_0} = k \cdot \frac{x - (x_0 - \varepsilon)}{t - t_0} = k \cdot s^- \tag{4.4}$$

Portanto, $\tilde{S}^+ = kS^+$ e $\tilde{S}^- = kS^-$. Como $k > 0$, a condição de violação é equivalente:

$$\tilde{S}^+ > \tilde{S}^- \iff kS^+ > kS^- \iff S^+ > S^- \tag{4.5}$$

A violação ocorre exatamente nos mesmos instantes em ambos os casos. O SDT aplicado a $k\,y$ com tolerância $k\,\varepsilon$ produz **exatamente os mesmos *keypoints*** que o SDT aplicado a y com tolerância ε. □

4.1.2 Consequência para a Parametrização

A invariância de escala implica que, para manter o mesmo comportamento de compressão quando a magnitude do sinal muda por um fator k, é necessário e suficiente escalar a tolerância pelo mesmo fator. Em outras palavras, o que determina o comportamento do SDT não é a tolerância absoluta ε (`tolerancia`), mas a razão percentual $\texttt{tol_rel} = 100 \cdot \varepsilon / \|y\|$, onde $\|\cdot\|$ é alguma medida de magnitude do sinal. Equivalentemente, a tolerância pode ser expressa como $\varepsilon = (\texttt{tol_rel}/100) \cdot \|y\|$, com `tol_rel` interpretado como uma porcentagem da magnitude do sinal.

Sendo assim, um valor fixo de `tol_rel` garante que dois sinais com perfil temporal idêntico, diferindo apenas na escala, sejam comprimidos com a mesma taxa. A parametrização original por `tolerancia` absoluta viola essa propriedade, pois um sinal de mesma forma mas magnitude dez vezes maior precisaria de uma tolerância dez vezes maior para produzir o mesmo resultado.

4.2 Caracterização da Magnitude pelo RMS Incremental

A invariância de escala exige uma medida de magnitude $\|y\|$ para expressar `tolerancia` como $\varepsilon = (\texttt{tol_rel}/100) \cdot \|y\|$. Em processamento amostra a amostra, essa medida deve ser computável de forma incremental, sem acesso ao sinal completo.

4.2.1 Por que RMS e não Média ou Máximo

Três candidatos naturais são a média, o máximo e o valor eficaz (RMS, raiz quadrada da média dos quadrados). A média aritmética é inadequada para o caso de uso deste trabalho. Em instalações com geração distribuída, como painéis solares ou geração própria conectada à rede, a potência ativa pode alternar entre valores positivos (consumo líquido) e negativos (injeção de energia na rede), fazendo com que a média temporal fique próxima de zero ao longo de um dia mesmo que o sinal tenha amplitude e variabilidade consideráveis. Usar a média como escala produziria uma `tolerancia` próxima de zero, tornando o algoritmo extremamente restritivo exatamente nos sinais onde a compressão mais importa. O máximo é sensível a picos esporádicos: um único evento transitório inflacionaria a tolerância e distorceria o comportamento do algoritmo nos períodos seguintes.

O RMS (*Root Mean Square*) é a medida padrão de magnitude em sinais elétricos e de processamento de sinais em geral. Para um sinal $x_1, \ldots, x_N$:

$$\mathrm{RMS}_N = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \tag{4.6}$$

O RMS é sempre não negativo, zero apenas quando o sinal é identicamente nulo, e escala linearmente com a amplitude: $\mathrm{RMS}(kx) = k\,\mathrm{RMS}(x)$ para $k > 0$. Essa última propriedade é precisamente o que a demonstração da seção anterior exige: $\tilde{\varepsilon} = (\texttt{tol_rel}/100) \cdot \mathrm{RMS}(kx) = (\texttt{tol_rel}/100) \cdot k \cdot \mathrm{RMS}(x) = k \cdot \varepsilon$, garantindo a invariância.

4.2.2 Atualização Incremental

Em processamento amostra a amostra, o RMS é mantido por meio da média quadrática acumulada, denominada `ms`, definida como:

$$\mathrm{ms}_N = \frac{1}{N}\sum_{i=1}^{N} x_i^2 \tag{4.7}$$

A partir dessa definição, é possível derivar uma forma de atualização incremental que não exige reter o histórico do sinal. Separando o último termo da soma:

$$\mathrm{ms}_N = \frac{1}{N}\left(\sum_{i=1}^{N-1} x_i^2 + x_N^2\right) \tag{4.8}$$

Como $\mathrm{ms}_{N-1} = \frac{1}{N-1}\sum_{i=1}^{N-1} x_i^2$, tem-se $\sum_{i=1}^{N-1} x_i^2 = (N-1)\,\mathrm{ms}_{N-1}$. Substituindo:

$$\mathrm{ms}_N = \frac{(N-1)\,\mathrm{ms}_{N-1} + x_N^2}{N} \tag{4.9}$$

o que resulta em:

$$\mathrm{ms}_N = \mathrm{ms}_{N-1} + \frac{x_N^2 - \mathrm{ms}_{N-1}}{N} \tag{4.10}$$

com $\mathrm{ms}_1 = x_1^2$. O RMS corrente é então $\mathrm{RMS}_N = \sqrt{\mathrm{ms}_N}$. Essa forma de atualização exige armazenar apenas dois escalares (`ms` e N) e realiza uma operação aritmética por amostra, sendo adequada para hardware embarcado com memória restrita.

4.2.3 Piso de Tolerância Absoluta

Quando o sinal permanece próximo de zero por um período prolongado (por exemplo, um circuito desligado registrando leituras nulas), `ms` tende a zero e `tolerancia` calculada como $(\texttt{tol_rel}/100) \cdot \sqrt{\mathrm{ms}}$ também tenderia a zero, tornando o algoritmo restritivo demais. Para evitar esse comportamento, `tolerancia` incorpora um piso mínimo absoluto $\varepsilon_{\min}$:

$$\varepsilon_N = \max\left(\frac{\texttt{tol_rel}}{100} \cdot \sqrt{\mathrm{ms}_N},\ \varepsilon_{\min}\right) \tag{4.11}$$

Nos experimentos deste trabalho, adotou-se $\varepsilon_{\min} = 0{,}001\,\mathrm{W}$. O piso atua apenas quando o sinal é essencialmente nulo e tem efeito desprezível em condições normais de operação. É ele que permite ao algoritmo comprimir sinais constantes ou próximos de zero, como um dia inteiro de leituras nulas (equipamento desligado) ou a frequência da rede elétrica em regime estável: sem o piso, ε tenderia a zero e cada amostra seria emitida como *keypoint*, anulando qualquer ganho de compressão.

4.3 Controle da Taxa de Compressão por Controlador PI

A parametrização via RMS garante que o comportamento do SDT seja comparável entre sinais de escalas diferentes, mas não determina qual valor de `tol_rel` produzirá uma taxa de compressão específica. A relação entre `tol_rel` e a taxa de compressão resultante $\mathrm{CR}(\texttt{tol_rel})$ depende da complexidade do sinal (o número e a amplitude das variações, a distribuição dos segmentos) e não tem forma analítica fechada. Não é possível calcular diretamente $\texttt{tol_rel} = \mathrm{CR}^{-1}(\mathrm{CR}_{\mathrm{alvo}})$.

A solução adotada no ARC-SDT é tratar o problema como controle em malha fechada: medir a taxa de compressão atual, compará-la com o alvo e ajustar `tol_rel` de forma iterativa. O SDT com

$\texttt{tolerancia} = (\texttt{tol_rel}/100) \cdot \sqrt{\text{ms}}$ é a *planta* do sistema; o controlador observa a saída (CR atual) e atua sobre a entrada (`tol_rel`).

4.3.1 Formulação do Controlador PID

O PID (*Proportional-Integral-Derivative*) é a estrutura de controle em malha fechada mais adotada na prática industrial por combinar três mecanismos complementares: o termo proporcional (P) gera ação imediata proporcional ao desvio atual; o termo integral (I) acumula o histórico de desvios e elimina o erro em regime permanente; o termo derivativo (D) responde à taxa de variação do desvio, amortecendo oscilações causadas por mudanças rápidas. A saída do controlador (`saida_pi`) define o próximo valor de `tol_rel`:

$$\texttt{tol_rel}_{n+1} = K_p \cdot \delta_n + K_i \sum_{j=0}^{n} \delta_j + K_d \cdot (\delta_n - \delta_{n-1}) \tag{4.12}$$

onde $\delta_n = \text{CR}_{\text{alvo}} - \text{CR}_n$ é o `desvio` no instante n, δ_j é o `desvio` no instante j (termo genérico da soma acumulada) e $\delta_n - \delta_{n-1}$ é sua variação entre passos consecutivos. O termo integral é o responsável por eliminar o erro residual que o controle proporcional puro não consegue anular: qualquer ponto de equilíbrio com desvio não-nulo exigiria ação de controle permanente, e o integrador resolve isso acumulando esse desvio até zerá-lo. A análise empírica dos três ganhos (apresentada na Seção 4.3.3) mostra que o termo derivativo não produz efeito mensurável neste sistema, confirmando que a implementação pode operar como PI puro ($K_d = 0$) sem perda de desempenho.

4.3.2 Mecanismos de Estabilidade

Dois mecanismos evitam que o controlador acumule integral indefinidamente (*integrator windup*) e oscile em torno do alvo.

Saturação com anti-*windup*: a técnica adotada é a suspensão condicional da integração, descrita por Åström e Hägglund [9] entre os esquemas clássicos de anti-*windup* para controladores PID. Aqui, `saida_pi` é limitada ao intervalo $[\texttt{tol_rel}_{\min}, \texttt{tol_rel}_{\max}] = [1, 10000]$ (em porcentagem). Quando `saida_pi` está fora desse intervalo, ela é truncada e o integral *não é atualizado* naquele passo: o acúmulo é suspenso enquanto o controlador estiver saturado. Isso evita que o integral cresça para valores que demandariam longo tempo de recuperação após a saturação.

Reinicialização do integral na inversão de sinal: quando o sinal de `desvio` muda de positivo para negativo ou vice-versa (indicando que o CR cruzou o alvo), o integral é zerado. Trata-se da estratégia proposta por Clegg [18], cujo integrador não linear zera a saída sempre que a entrada cruza o

zero, justamente para evitar o sobressinal prolongado que a inércia do integrador linear provoca. Isso impede a oscilação induzida por inércia integral: se o controlador ultrapassou o alvo (`desvio` mudou de sinal), reiniciar o integral faz com que a próxima `saida_pi` seja baseada apenas no novo `desvio` proporcional, sem o peso do histórico acumulado no sentido oposto.

4.3.3 Ajuste dos Parâmetros do Controlador

`tol_rel` é inicializado com o próprio `target_cr` (em porcentagem), mas a escolha do valor inicial é irrelevante na prática. Como `ms` parte de $\mathrm{ms}_1 = x_1^2$ (apenas o primeiro ponto), qualquer valor inicial de `tol_rel` produziria uma `tolerancia` dependente exclusivamente de $|x_1|$, que raramente representa a amplitude do sinal completo. Uma série que começa com o equipamento desligado ($x_1 \approx 0$) terá `tolerancia` próxima de zero independentemente do `tol_rel` escolhido; uma série que começa em pico de carga terá `tolerancia` excessivamente larga pelo mesmo motivo. Em ambos os casos, o controlador PI precisa corrigir o parâmetro desde os primeiros instantes. A consequência é um transiente inicial de duração variável: à medida que `ms` acumula mais amostras e passa a refletir a variabilidade real do sinal, o controlador converge para o regime adequado. É importante registrar, contudo, que a convergência observada é empírica: não foram estabelecidas garantias formais de estabilidade, o que está fora do escopo deste trabalho. É importante destacar que o desvio negativo é o único com implicação prática neste contexto: ele significa que o pacote gerado é maior do que o alvo de carga útil, podendo exceder o limite do canal. O desvio positivo, que pode ser expressivo em alguns sinais, é aceitável: indica que o sinal admite compressão naturalmente superior ao alvo, gerando um pacote ainda menor, o que não compromete os objetivos da transmissão. Esse comportamento está, aliás, fora do controle do SDT por ser consequência direta da estrutura do sinal: quando o sinal é regular o suficiente para admitir mais compressão do que o alvo, o algoritmo geométrico não tem mecanismo para reter pontos adicionais apenas para reduzir o CR.

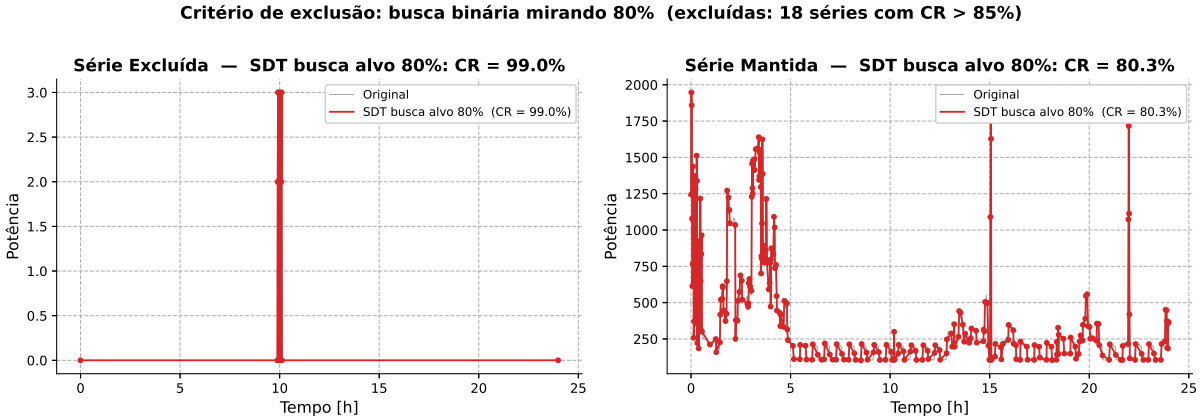
Cabe registrar como a janela usada nesta análise se relaciona com a operação real do algoritmo. O ARC-SDT é concebido como um algoritmo amostra a amostra: comprime cada valor à medida que chega, sem manter *buffer* do sinal completo, e o acumulador `ms` incorpora cada nova amostra sem descartar as anteriores. Em campo, portanto, não existe janela: o controlador opera de forma contínua enquanto houver medição, acumulando estatísticas sem nenhum ponto de corte imposto pelo algoritmo. A análise de ajuste, porém, precisa de trechos de tamanho definido, para que o que se observa em uma série possa ser comparado com o que se observa em outra. Adotou-se aqui a janela de 1.440 pontos, que corresponde a um dia de medição no intervalo de um minuto, o caso de uso natural do ARC-SDT. O dia é o recorte adequado porque o comportamento de uso de energia tende a se repetir de um dia para

o outro, acompanhando a rotina da instalação: as 24 horas cobrem o ciclo completo de operação, e um trecho maior repetiria em boa parte o mesmo ciclo.

Todo o ajuste foi conduzido com alvo de 80%, escolhido por dois motivos. Primeiro, é o menor valor da grade de taxas avaliadas (Seção 5.4.3) que atende à redução de pelo menos 75% exigida pelo SF10, o fator de espalhamento mais restritivo que ainda viabiliza a transmissão no cenário deste trabalho (Tabela 2.2). Segundo, é a taxa usada nas tabelas resumo em que os algoritmos são comparados, o que mantém o ajuste do controlador no mesmo ponto de operação em que o desempenho é confrontado. Os três ganhos do controlador PID (K_p, K_i e K_d) foram ajustados empiricamente em etapas sequenciais, avaliando a distribuição do CR final e a trajetória de convergência sobre as séries mantidas após a filtragem, com o controlador atuando a cada nova amostra processada (`update_interval` = 1). O processo partiu do ganho proporcional com os demais zerados, depois acrescentou o integral e, por último, avaliou o derivativo, para isolar o efeito de cada parâmetro. Essa sequência segue o efeito isolado de cada ganho descrito por Ang, Chong e Li [5] e a prática comum de ajuste manual em engenharia de controle, apresentada também em material técnico de fabricantes [41]. O ajuste foi conduzido exclusivamente sobre a Coleção Pública: a Coleção Privada é reservada como conjunto de validação genuinamente não visto durante o ajuste, avaliada apenas na comparação final do Capítulo 6. Ajustar e validar sobre o mesmo conjunto de dados inflaria artificialmente a qualidade de convergência reportada; separar as coleções permite verificar se os ganhos escolhidos generalizam para séries que não influenciaram nenhuma escolha de projeto. A escolha de atuar a cada amostra é revisitada e justificada empiricamente na Seção 4.3.4, que avalia intervalos de atualização maiores.

Para a análise das trajetórias de convergência, séries de alta compressibilidade foram excluídas previamente. Essas séries (caracterizadas por comportamento muito regular ou fortemente tendencioso) são comprimidas pelo ARC-SDT muito além do alvo independentemente dos ganhos configurados, pois a estrutura do sinal, e não o controlador, determina o CR resultante. Incluí-las distorceria a avaliação do comportamento de controle. A identificação foi feita por busca binária de tolerância sobre o **SDT puro**, mirando CR = 80%, o mesmo mecanismo de busca usado para SDT e RDP no protocolo experimental principal (Seção 5.4.4): se, mesmo com a busca ativamente tentando aproximar o CR do alvo, o resultado obtido ultrapassa 85% (uma margem de 5% acima do alvo), a série é considerada de alta compressibilidade e excluída, pois não existe tolerância que produza, para ela, um CR fino nas proximidades de 80%. Esse critério é direto e independente dos parâmetros do controlador PID: usa apenas o SDT base como medida de compressibilidade da série, sem viés circular. A Figura 4.2 ilustra o contraste entre uma série excluída e uma mantida. Das 167 séries da Coleção Pública, 18 foram descartadas por esse critério, restando 149 para a análise de ajuste que se segue.

Figura 4.2 — Exemplo de série excluída (esquerda) e mantida (direita) pelo critério de filtragem, ambas reconstruídas pelo SDT puro após a mesma busca binária mirando CR = 80%. A série excluída atinge CR = 99,0% mesmo com a busca tentando reduzi-lo até o alvo (mediana das 18 séries excluídas: 99,2%, variando de 86,5% a 99,9%); ela permanece praticamente nula durante todo o dia, com um único evento isolado por volta das 10h, e um sinal assim é descrito pelo SDT com pouquíssimos pontos qualquer que seja a tolerância. A série mantida atinge CR = 80,3% (mediana das 149 séries mantidas), com variabilidade real e contínua que permite à busca aproximar o CR do alvo com precisão fina.

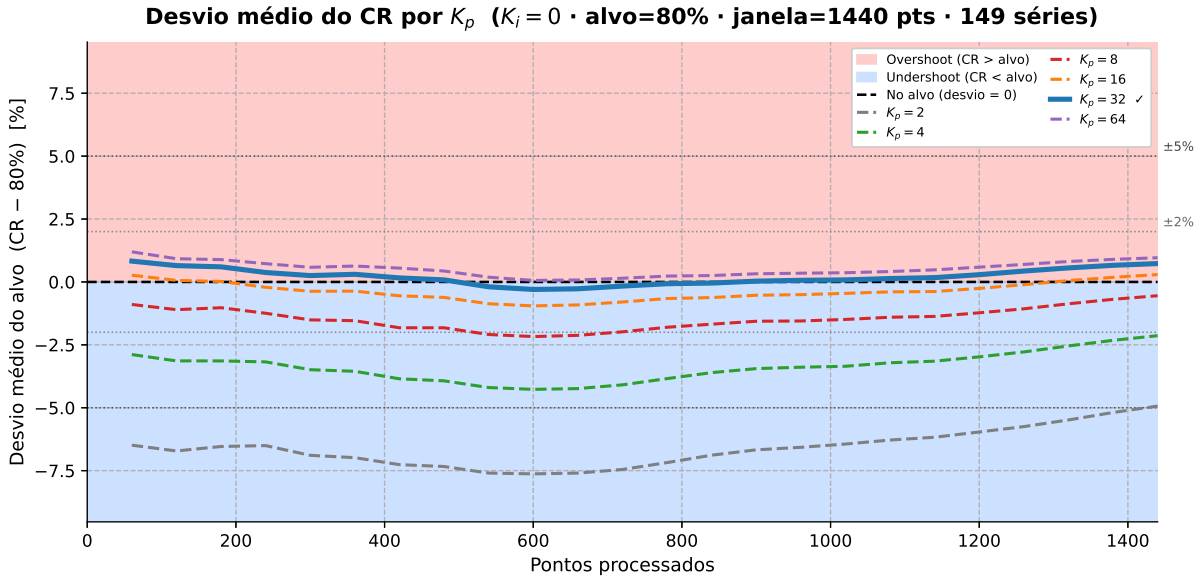


Fonte: elaborada pelo autor.

Ganho Proporcional (K_p)

Na primeira etapa, avaliou-se K_p com $K_i = 0$, de modo que a escolha não dependesse de um K_i ainda não definido. A medida usada para decidir K_p é a raiz do erro quadrático médio (RMSE) entre o desvio do alvo e zero, calculada sobre a trajetória média do CR ao longo da janela (a média ponto a ponto do CR entre as 149 séries da Coleção Pública mantidas após a filtragem): quanto menor o RMSE, mais próxima do alvo a trajetória permanece, em média, do início ao fim do processamento. Essa medida captura não apenas o ponto de chegada mas todo o caminho até lá, o que é relevante porque, como discutido adiante para o termo integral, um K_p que demora a convergir passa uma fração maior da janela longe do alvo, mesmo que termine perto dele. A Figura 4.3 mostra a trajetória média para cada valor testado. Com $K_p = 2$, a trajetória média termina a $-4{,}93\%$ do alvo e nunca o ultrapassa (RMSE = 6,65%). A convergência melhora progressivamente: $K_p = 8$ termina a $-0{,}55\%$ (RMSE = 1,51%) e $K_p = 16$ termina a $+0{,}29\%$, já com um desvio positivo residual mínimo no fim da trajetória (RMSE = 0,51%). Com $K_p = 32$, o desvio final é de $+0{,}73\%$ e o pico de desvio positivo ao longo da trajetória é de $+0{,}83\%$, mas por convergir mais rápido que os valores menores, o RMSE cai para 0,39%, o menor entre todos os valores testados. Com $K_p = 64$, o desvio final sobe para $+0{,}96\%$ e o pico para $+1{,}20\%$, e o RMSE piora para 0,61%, sinal de que o ganho excessivo já introduz mais oscilação do que ganho de velocidade. **Escolhido:** $K_p = 32$, menor RMSE de trajetória entre os valores testados.

Figura 4.3 — Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_p ($K_i = 0$, alvo = 80%, 149 séries da Coleção Pública após exclusão das séries de alta compressibilidade). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%.



Fonte: elaborada pelo autor.

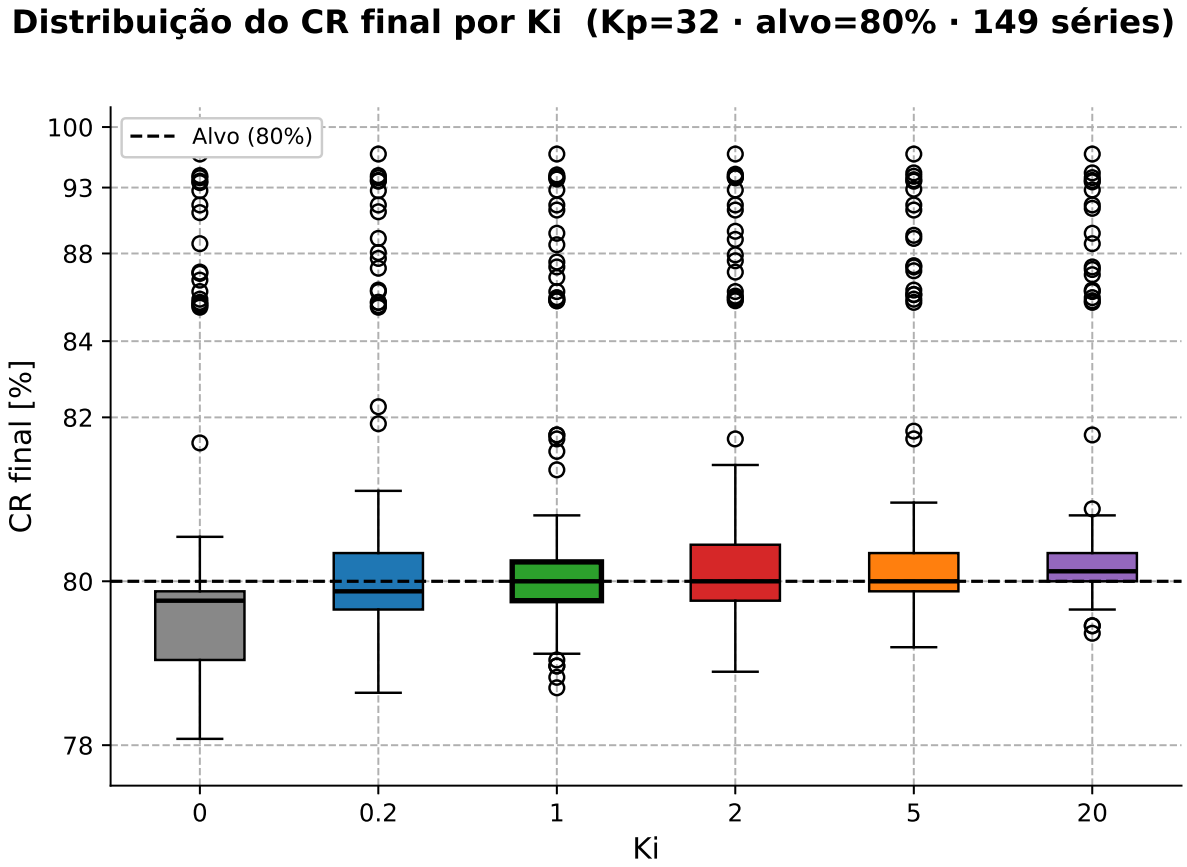
Ganho Integral (K_i)

Com $K_p = 32$ fixado, cabe agora perguntar: o proporcional puro já é suficiente, ou o CR final ainda fica abaixo do alvo, algo que o integral deveria corrigir? Para responder, é preciso mudar a medida usada até aqui, porque os dois ganhos fazem coisas diferentes. O proporcional governa o caminho até o alvo: define a rapidez da reação e o quanto o controlador passa do ponto antes de estabilizar, o que só aparece ao longo da janela e por isso foi medido pelo RMSE de trajetória. O integral existe para eliminar o erro que sobra quando só o proporcional atua, e isso aparece no valor final, não no caminho. Manter o RMSE aqui levaria à decisão errada, já que ele é mínimo em $K_i = 0$ e só piora à medida que o ganho cresce, o que descartaria o integral antes de ver o que ele melhora. O que interessa agora é o **CR entregue ao final da janela**, ou seja, o do pacote que seria de fato transmitido, resumido em cada uma das 149 séries pela mediana e pelo IQR (intervalo interquartil, a diferença entre o terceiro e o primeiro quartil): quanto mais próxima do alvo a mediana e mais estreito o IQR, mais parecido é o resultado entre séries de perfis diferentes. Com $K_p = 32$ e $K_i = 0$, a mediana é 79,86%, com Q1 = 79,31% e Q3 = 79,93% (IQR de 0,62%): a maioria das séries fecha um pouco abaixo do alvo. A Figura 4.4 mostra como essa distribuição se desloca e se estreita à medida que K_i cresce.

Os dois modos de desvio final têm naturezas distintas. Terminar abaixo do alvo gera uma carga útil maior do que o necessário, podendo exceder o limite do canal. Terminar acima do alvo significa que o

controlador elevou `tol_rel` além do necessário: com o corredor do SDT mais largo do que o requerido, pontos que seriam relevantes (picos de demanda, transientes ou variações rápidas) são absorvidos pelo corredor e descartados, comprometendo a fidelidade da reconstrução sem nenhum ganho adicional em capacidade de transmissão. No contexto de monitoramento de energia, onde eventos de pico têm significado operacional, esse descarte desnecessário é indesejável.

Figura 4.4 — Distribuição do CR final por valor de K_i com $K_p = 32$ fixo, alvo = 80% e janela de 1.440 pontos. Cada *boxplot* resume 149 séries da Coleção Pública, após exclusão das séries de alta compressibilidade. O eixo y aplica compressão logarítmica centrada no alvo ($\text{sign}(\delta) \cdot \ln(1 + |\delta|)$, com $\delta = \text{CR} - 80\%$), preservando a escala linear na vizinhança do alvo e comprimindo os extremos para melhor visualização.



Fonte: elaborada pelo autor.

A Tabela 4.1 reúne as duas medidas lado a lado para cada valor de K_i testado: o RMSE da trajetória média (a mesma métrica usada para escolher K_p) e a mediana com o IQR do CR final entre as 149 séries (a métrica usada para decidir o termo integral). As duas pioram e melhoram em direções opostas à medida que K_i cresce a partir de zero. O RMSE é menor exatamente em $K_i = 0$ (0,39%) e cresce à medida que K_i aumenta, chegando a 1,00% em $K_i = 20$: como o integral empurra a trajetória para

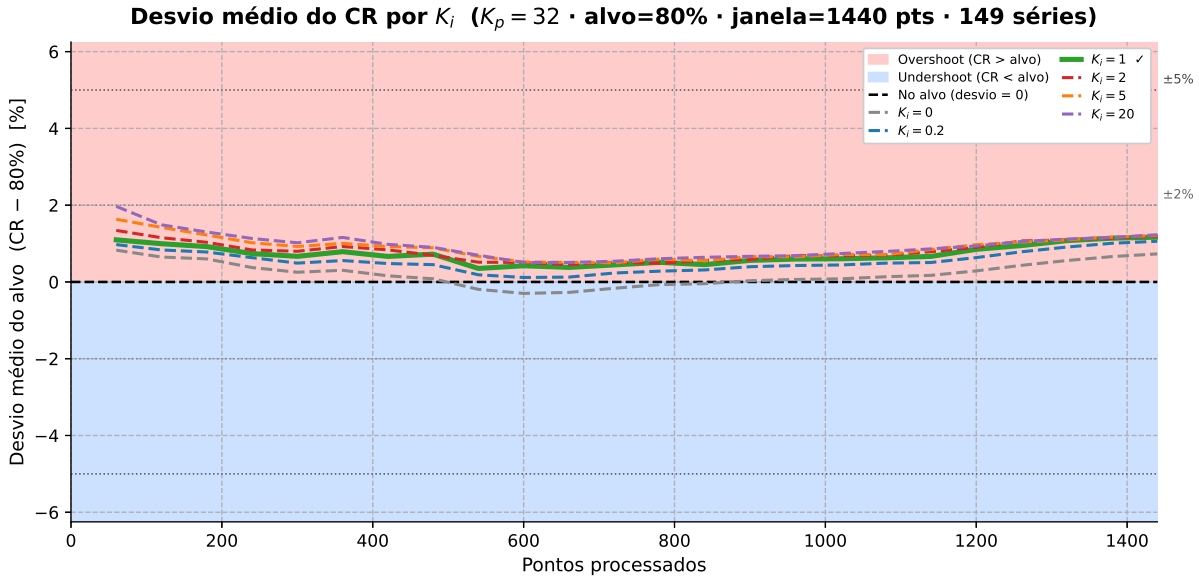
além do alvo ao longo de todo o caminho (não só ao final), ele piora o erro médio de trajetória. A mediana do CR final, por outro lado, só atinge exatamente o alvo (80,00%) a partir de $K_i = 1$, e o IQR se estreita de 0,62% em $K_i = 0$ para 0,28%. É importante destacar que o IQR continua encolhendo para além disso, chegando a 0,21% em $K_i = 20$, mas esse ganho vem acompanhado de um deslocamento da mediana para fora do alvo (80,07%), o que anula justamente o motivo de introduzir o termo integral. Esse é o comportamento discutido na Figura 4.5: o desvio final passa de $+0{,}73\%$ ($K_i = 0$) para $+1{,}17\%$ ($K_i = 1$), e o pico de desvio positivo ao longo da trajetória cresce de $+0{,}83\%$ para $+1{,}17\%$ (uma piora real na trajetória, aceita em troca de um resultado final mais consistente entre séries de perfis diferentes). **Escolhido:** $K_i = 1$: é o menor valor que centra exatamente a mediana do CR final no alvo e, entre os três que a centram ($K_i = 1$, 2 e 5), é o que combina o IQR mais estreito (0,28%, empatado com $K_i = 5$) com o menor RMSE de trajetória (0,76% contra 0,94%). Para o resultado efetivamente transmitido, esse fechamento no alvo importa mais do que o comportamento durante a trajetória.

Tabela 4.1 — Métricas de convergência para diferentes valores de K_i ($K_p = 32$, alvo = 80%, janela de 1.440 pontos, 149 séries da Coleção Pública). Desvio final, pico e vale referem-se à trajetória média do CR ao longo da janela; RMSE é o erro quadrático médio dessa mesma trajetória em relação ao alvo; mediana e IQR referem-se à distribuição do CR final medido individualmente em cada série.

K_i	Desvio final	Pico	Vale	RMSE	Mediana final	IQR
0	$+0{,}73\%$	$+0{,}83\%$	$-0{,}30\%$	0,39%	79,86%	0,62
0,2	$+1{,}06\%$	$+1{,}06\%$	$+0{,}12\%$	0,61%	79,93%	0,42
1	**+1,17%**	**+1,17%**	**+0,35%**	**0,76%**	**80,00%**	**0,28**
2	$+1{,}19\%$	$+1{,}34\%$	$+0{,}45\%$	0,85%	80,00%	0,42
5	$+1{,}21\%$	$+1{,}63\%$	$+0{,}50\%$	0,94%	80,00%	0,28
20	$+1{,}23\%$	$+1{,}97\%$	$+0{,}51\%$	1,00%	80,07%	0,21

Fonte: elaborada pelo autor.

Figura 4.5 — Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_i ($K_p = 32$, alvo = 80%, 149 séries da Coleção Pública após exclusão das séries de alta compressibilidade). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%.

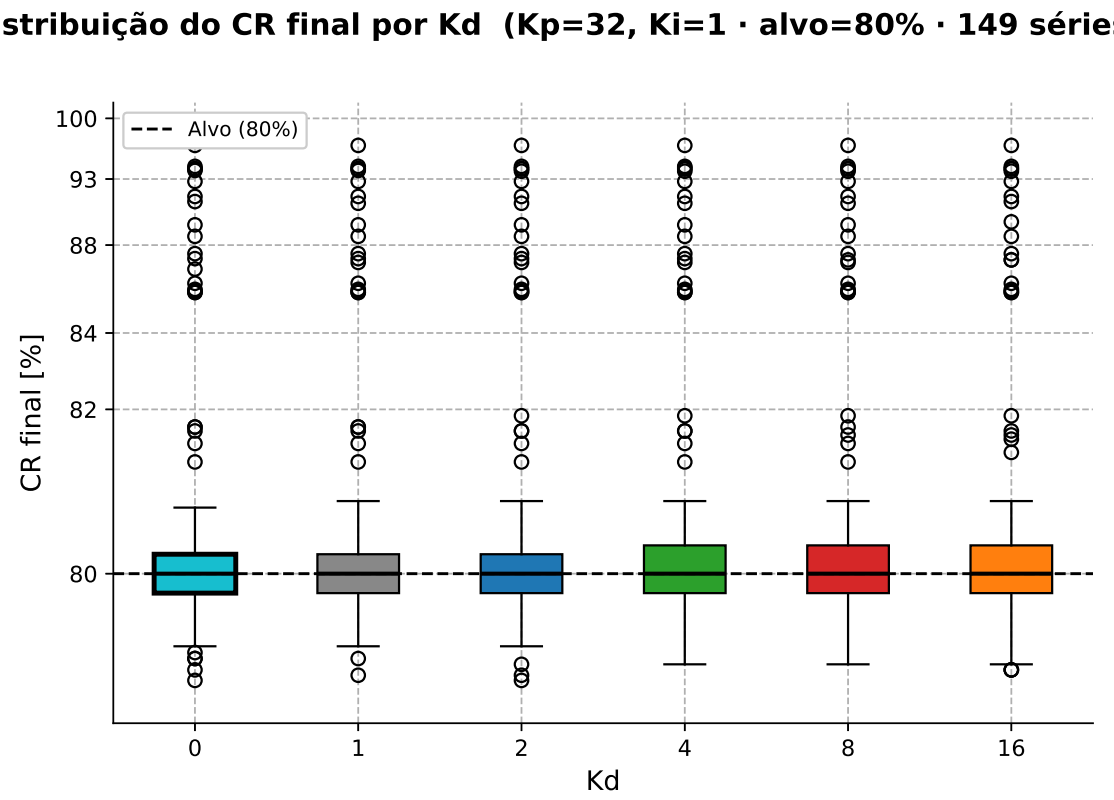


Fonte: elaborada pelo autor.

Ganho Derivativo (K_d)

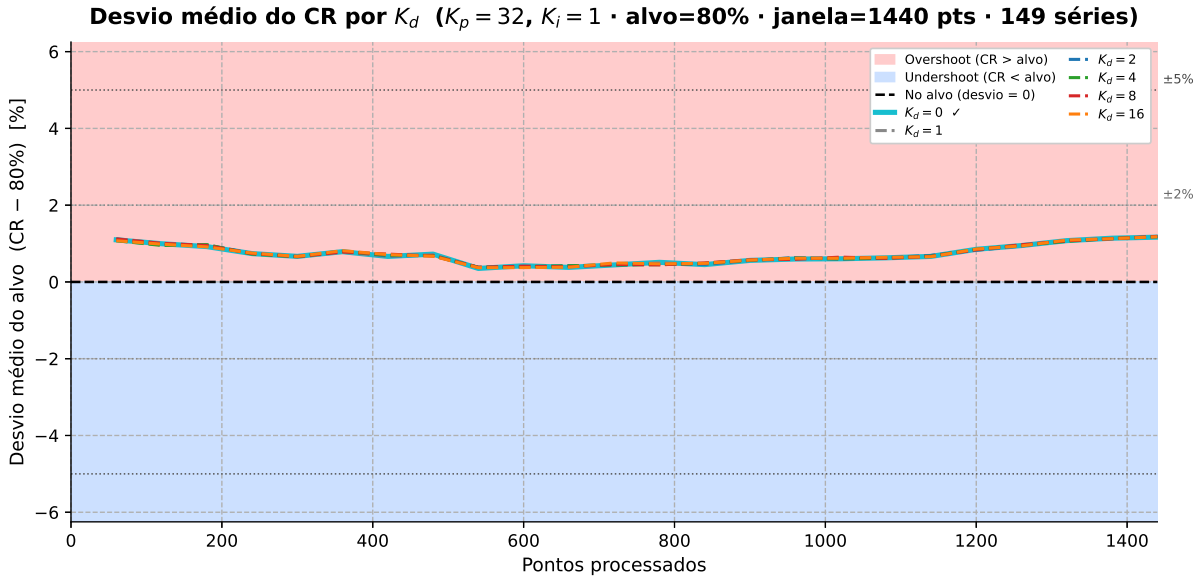
Com $K_p = 32$ e $K_i = 1$ definidos, o controlador PI já apresenta comportamento satisfatório. Resta avaliar se o termo derivativo (responsável por amortecer oscilações freiando a ação de controle quando o erro está convergindo rapidamente) traz algum benefício adicional neste sistema específico, ou se pode ser eliminado sem custo. O termo derivativo responde à taxa de variação do erro: quando o CR está convergindo rapidamente para o alvo, o derivativo freia a saída antes que o cruzamento ocorra, reduzindo potencialmente o desvio positivo. Para avaliar se esse mecanismo oferece algum benefício neste sistema, avaliou-se $K_d \in \{0, 1, 2, 4, 8, 16\}$ com $K_p = 32$ e $K_i = 1$ fixos, usando um filtro de primeira ordem com $\alpha = 0{,}9$ sobre a derivada bruta do erro, consistente com a implementação do ARC-SDT.

Figura 4.6 — Distribuição do CR final por valor de K_d com $K_p = 32$ e $K_i = 1$ fixos, alvo = 80% e janela de 1.440 pontos. Cada *boxplot* resume 149 séries da Coleção Pública, após exclusão das séries de alta compressibilidade. Eixo y com compressão logarítmica centrada no alvo (ver Figura 4.4).



Fonte: elaborada pelo autor.

Figura 4.7 — Desvio médio do CR em relação ao alvo ao longo da janela de 1.440 pontos para diferentes valores de K_d ($K_p = 32$, $K_i = 1$, alvo = 80%, 149 séries da Coleção Pública). Região vermelha = desvio positivo; região azul = desvio negativo. Linhas pontilhadas indicam ±2% e ±5%.



Fonte: elaborada pelo autor.

Como mostra a Tabela 4.2, todos os valores de K_d resultam em RMSE de trajetória praticamente idêntico entre si (entre 0,76% e 0,77%) e ao de $K_d = 0$ (0,76%, correspondente ao $K_i = 1$ da Tabela 4.1), sem tendência de melhora ou piora conforme K_d cresce. A ausência de efeito é estrutural: o CR é uma média acumulada que varia tipicamente pouco por passo, de forma lenta e sem degraus abruptos. O derivativo filtrado de um sinal com essa característica resulta em contribuição próxima de zero independentemente do valor de K_d.

Essa ausência de efeito é sobre a trajetória agregada; olhando para as séries individualmente, um efeito residual pequeno aparece especificamente na cauda de pior desvio negativo. Com $K_d = 0$, 8 das 149 séries de ajuste ficam abaixo da cerca inferior do *boxplot* (o limiar padrão de 1,5 vez o IQR abaixo do primeiro quartil, usado para identificar valores atípicos), com o pior caso individual em 78,96% (desvio de $-1{,}04\%$ em relação ao alvo). Com $K_d = 4$, esse número cai para 5 séries abaixo da cerca, e o pior caso individual sobe para 79,17% (desvio de $-0{,}83\%$); das 8 séries originalmente na cauda, 6 melhoram de fato (não apenas mudam de classificação estatística) e 2 permanecem inalteradas. Ainda assim, essa melhora não justifica adotar o termo derivativo: afeta apenas 3% a 5% das séries, não desloca o RMSE nem a mediana da distribuição geral, e o pior caso já em $K_d = 0$ ($-1{,}04\%$) está dentro da margem considerada aceitável (Seção 4.3.3, discussão de **Parâmetros Selecionados**). Diante de um ganho tão restrito, o custo de manter o filtro derivativo e mais um parâmetro para calibrar em um controlador embarcado

não se justifica. **O termo derivativo foi desconsiderado** ($K_d = 0$), simplificando o controlador para PI puro.

Tabela 4.2 — Métricas de convergência da trajetória média para diferentes valores de K_d ($K_p = 32$, $K_i = 1$, alvo = 80%, janela de 1.440 pontos, 149 séries da Coleção Pública).

K_d	Desvio final	Pico	Vale	RMSE
0	**+1,17%**	**+1,17%**	**+0,35%**	**0,76%**
1	$+1{,}19\%$	$+1{,}19\%$	$+0{,}37\%$	$0{,}77\%$
2	$+1{,}18\%$	$+1{,}18\%$	$+0{,}38\%$	$0{,}77\%$
4	$+1{,}18\%$	$+1{,}18\%$	$+0{,}38\%$	$0{,}76\%$
8	$+1{,}19\%$	$+1{,}19\%$	$+0{,}38\%$	$0{,}77\%$
16	$+1{,}18\%$	$+1{,}18\%$	$+0{,}37\%$	$0{,}77\%$

Fonte: elaborada pelo autor.

Parâmetros Selecionados

As três etapas de ajuste produziram os seguintes parâmetros para o controlador do ARC-SDT:

- $K_p = 32$: menor RMSE de trajetória entre os valores testados no controle proporcional puro;
- $K_i = 1$: menor valor que centra a mediana do CR final exatamente no alvo e, entre os que a centram, o que reúne o IQR mais estreito e o menor RMSE de trajetória; esse RMSE é, ainda assim, maior do que o de $K_i = 0$;
- $K_d = 0$: sem efeito mensurável em nenhuma das combinações de K_p e K_i avaliadas; controlador PID reduz-se a PI puro.

Com $K_p = 32$ e $K_i = 1$, o CR final medido individualmente nas 149 séries de ajuste tem mediana de 80,00%: 47% das séries terminam acima do alvo, 42% abaixo e as demais atingem o alvo exatamente. O pior caso de desvio negativo entre todas as séries é de $-1{,}04\%$, e o pior caso de desvio positivo chega a $+16{,}53\%$ (uma série cuja estrutura permite compressão muito além do alvo, discutido a seguir). Como já estabelecido, apenas o desvio negativo tem implicação prática de estouro de carga útil, e o pior caso observado ($-1{,}04\%$) é pequeno o suficiente para ser absorvido por uma margem de segurança modesta no dimensionamento do pacote.

Sendo assim, percebemos que o sistema de controle do ARC-SDT é naturalmente compatível com um controlador PI, sem necessidade do termo derivativo. Isso não é uma simplificação arbitrária: é uma

consequência direta da natureza do sinal controlado, o CR acumulado, que varia lentamente e de forma contínua, não apresentando os degraus abruptos que motivariam o uso do derivativo. A ausência de efeito do termo derivativo simplifica a implementação: o ARC-SDT opera com um controlador PI, reduzindo a complexidade de otimização de três parâmetros para dois e eliminando o custo computacional do filtro derivativo. A equação efetiva, substituta da Equação 4.12, é:

$$\texttt{tol_rel}_{n+1} = K_p \cdot \delta_n + K_i \sum_{j=0}^{n} \delta_j \tag{4.13}$$

4.3.4 Intervalo de Atualização

O controlador pode ser atualizado a cada ponto processado ou a intervalos regulares de `update_interval` pontos. Avaliou-se `update_interval` $\in \{1, 10, 50, 100\}$ com $K_p = 32$ e $K_i = 1$ fixos: intervalos maiores introduzem atraso entre a mudança de `tol_rel` e seu efeito observável no CR, gerando oscilações amplas logo no início da janela. Com `update_interval` = 1, a trajetória média nunca se afasta do alvo em mais de $+1{,}17\%$ (RMSE de trajetória de 0,76%, a mesma trajetória de $K_i = 1$ já discutida); com `update_interval` = 100, a mesma trajetória chega a um pico de $+16{,}30\%$ acima do alvo por volta do ponto 60 antes de se estabilizar (RMSE de 4,70%), embora o desvio final acabe próximo do alvo em todos os valores testados. Nos experimentos deste trabalho, utilizou-se `update_interval` = 1, que oferece a trajetória mais estável sem penalidade computacional relevante para sinais de 1.440 pontos.

4.4 Descrição do Algoritmo

O ARC-SDT mantém, a cada ponto processado, o seguinte conjunto de variáveis de estado: `ancora` (último *keypoint* emitido, de onde o corredor parte), `ultimo_valido` (último ponto que coube dentro do corredor), `ms` (média quadrática acumulada), `tolerancia` (meia-largura atual do corredor), `max_incl_sup` e `min_incl_inf` (inclinações máxima e mínima acumuladas desde `ancora`), além das variáveis do controlador PI (Equação 4.13): `tol_rel`, `integral` e `desvio_prev`. A Tabela 4.3 apresenta os parâmetros configuráveis pelo usuário; `kd` é exposto pela interface mas fixado em zero, conforme a análise empírica da Seção 4.3.3.

Tabela 4.3 — Parâmetros do ARC-SDT

Parâmetro	**Efeito**
`target_cr`	Taxa de compressão alvo (%), entre 0 e 100
`kp`	Ganho proporcional do controlador PI (padrão: 32)
`ki`	Ganho integral do controlador PI (padrão: 1)
`kd`	Ganho derivativo do controlador PI (padrão: 0; sem efeito mensurável neste sistema, conforme Seção 4.3.3)
`update_interval`	Intervalo entre atualizações do controlador (em número de pontos)
`min_absolute_error`	Piso mínimo da tolerância absoluta $\varepsilon_{\min}$, ativo quando o RMS é muito pequeno (padrão: 0,001)

Fonte: elaborada pelo autor.

O fluxo de processamento para cada nova amostra (t_n, x_n) é:

1. Atualizar `ms`: $\mathrm{ms}_n \leftarrow \mathrm{ms}_{n-1} + (x_n^2 - \mathrm{ms}_{n-1})/n$
2. Recalcular `tolerancia`: $\varepsilon_n \leftarrow \max\left(\frac{\texttt{tol_rel}}{100} \cdot \sqrt{\mathrm{ms}_n},\ \varepsilon_{\min}\right)$; reposicionar pivôs em `ancora` $\pm\, \varepsilon_n$
3. Calcular as inclinações do ponto atual até cada pivô; atualizar `max_incl_sup` e `min_incl_inf`
4. Se `max_incl_sup` $\leq$ `min_incl_inf`: corredor válido; atualizar `ultimo_valido`
5. Senão: corredor violado; emitir `ultimo_valido` como *keypoint*, avançar `ancora`, resetar inclinações, reprocessar o ponto atual
6. A cada `update_interval` pontos: calcular `desvio` $= \mathrm{CR}_{\text{alvo}} - \mathrm{CR}_{\text{atual}}$, computar `saida_pi`, aplicar anti-*windup* e reinicialização de integral, atualizar `tol_rel`

4.4.1 Pseudocódigo

Código Fonte 4.1 — Pseudocódigo do ARC-SDT

```
1 funcao ARC_SDT(serie, target_cr, Kp, Ki,
```

```
2                 update_interval, e_min, tau_min=1, tau_max=10000):
3
4       // Inicializacao
5       tol_rel <- target_cr        // tolerancia relativa inicial (em %)
6       ms <- serie[0].valor^2; N <- 1
7       tolerancia <- max(sqrt(ms) * tol_rel / 100, e_min)
8       ancora <- serie[0]; ultimo_valido <- serie[0]
9       max_incl_sup <- -inf; min_incl_inf <- +inf
10      keypoints <- [serie[0]]
11      integral <- 0; desvio_prev <- nulo
12
13      para i de 1 ate len(serie)-1:
14          (t, x) <- serie[i]
15
16          // 1. Atualizar ms e recalcular tolerancia
17          N <- N + 1
18          ms <- ms + (x^2 - ms) / N
19          tolerancia <- max(sqrt(ms) * tol_rel / 100, e_min)
20
21          // 2. Reposicionar pivos no ponto ancora com a nova tolerancia
22          pivo_sup <- (ancora.t, ancora.x + tolerancia)
23          pivo_inf <- (ancora.t, ancora.x - tolerancia)
24
25          // 3. Calcular inclinacoes e atualizar leque
26          s_pos <- (x - pivo_sup.x) / (t - pivo_sup.t)
27          s_neg <- (x - pivo_inf.x) / (t - pivo_inf.t)
28          max_incl_sup <- max(max_incl_sup, s_pos)
29          min_incl_inf <- min(min_incl_inf, s_neg)
30
31          se max_incl_sup <= min_incl_inf:    // corredor valido
32              ultimo_valido <- (t, x)
33          senao:                               // corredor violado
34              keypoints.adicionar(ultimo_valido)
35              ancora <- ultimo_valido
36              max_incl_sup <- -inf; min_incl_inf <- +inf
37              // reprocessa ponto atual sem atualizar ms
38              s_pos <- (x-(ancora.x+tolerancia))/(t-ancora.t)
39              s_neg <- (x-(ancora.x-tolerancia))/(t-ancora.t)
```

```
40              max_incl_sup <- s_pos; min_incl_inf <- s_neg
41              ultimo_valido <- (t, x)
42
43          // 4. Atualizar controlador PI a cada update_interval pontos
44          se (i + 1) mod update_interval == 0 ou i == len(serie)-1:
45              CR_atual <- 100 * (1 - len(keypoints) / (i + 1))
46              desvio <- target_cr - CR_atual
47              se desvio_prev != nulo e sinal(desvio) != sinal(desvio_prev):
48                  integral <- 0       // reinicializar na inversao de sinal
49              saida_pi <- Kp * desvio + Ki * (integral + desvio)
50              se tau_min <= saida_pi <= tau_max:
51                  integral <- integral + desvio   // anti-windup
52              tol_rel <- clamp(saida_pi, tau_min, tau_max)
53              desvio_prev <- desvio
54
55      se keypoints[-1] != serie[-1]:
56          keypoints.adicionar(serie[-1])
57      retorna keypoints
```

4.5 Propriedades para Dispositivos Embarcados

O ARC-SDT preserva as propriedades do SDT que o tornam atraente para hardware embarcado, com custo adicional mínimo.

Memória constante: além das variáveis já mantidas pelo SDT (`max_incl_sup`, `min_incl_inf`, pivôs, `ultimo_valido`), o ARC-SDT armazena apenas `ms`, N, `tol_rel`, `integral` e a contagem de pontos desde a última atualização, todos escalares. O uso de memória é $O(1)$ independentemente do comprimento da série.

Processamento amostra a amostra: cada amostra é processada individualmente à medida que chega, o que traz duas vantagens práticas independentes. A primeira é a ausência de *buffering*: o dispositivo não precisa armazenar o sinal completo para iniciar a compressão, tornando o algoritmo viável em hardware com memória mínima, independentemente do tempo de coleta. A segunda é a latência de notificação: um evento (pico de demanda, transitório ou anomalia de consumo) gera um *keypoint* assim que `max_incl_sup` $>$ `min_incl_inf`, ou seja, quando a tendência associada ao evento se encerra. O dispositivo pode transmitir esse dado imediatamente, sem aguardar o fim de uma janela de coleta. Em algoritmos em lote, o mesmo evento só seria comunicado após o processamento completo da janela con-

figurada, que pode corresponder a horas ou um dia inteiro de coleta quando janelas maiores são usadas para maximizar o desempenho de compressão. Cabe destacar que implementações adicionais poderiam reduzir essa diferença nos algoritmos em lote (por exemplo, transmitindo batches parciais a intervalos regulares); este trabalho considera apenas o processo de compressão, sem otimizações adicionais de transmissão.

Custo computacional: cada ponto exige um número constante de operações aritméticas. A única raiz quadrada adicional em relação ao SDT original é a do RMS, calculada a cada ponto para derivar `tolerancia` de `ms`. O custo total é $O(n)$ em tempo e $O(1)$ em memória, idêntico ao SDT.

Capítulo 5

Metodologia de Avaliação

5.1 Visão Geral

A metodologia deste trabalho foi conduzida em duas fases complementares. Na primeira, realizou-se uma avaliação comparativa de quatro abordagens de compressão (*Swinging Door Trending*, SDT; Ramer-Douglas-Peucker, RDP; Transformada Discreta do Cosseno, DCT; e Transformada Wavelet Discreta, DWT), totalizando seis variantes de algoritmo, pois a DWT foi avaliada com três famílias de wavelets distintas (haar, sym2 e bior5.5). Esses algoritmos foram aplicados sobre um conjunto amplo de séries temporais de consumo de energia elétrica, cobrindo diferentes combinações de taxa de compressão alvo e tamanho de janela de processamento, com coleta sistemática de métricas de qualidade de reconstrução.

Na segunda fase, motivada tanto pelas limitações identificadas no SDT durante a avaliação comparativa quanto pelo seu potencial para uso em dispositivos IoT com restrições de memória e processamento, propôs-se o algoritmo ARC-SDT (*Adaptive Real-time Controlled* SDT), descrito em capítulo próprio. O SDT, por ser um algoritmo amostra a amostra com custo computacional e de memória muito reduzidos, é um candidato natural para execução em microcontroladores e dispositivos embarcados de baixo custo; a proposta do ARC-SDT busca potencializar esse perfil adicionando controle adaptativo da taxa de compressão. O ARC-SDT foi então incluído no mesmo protocolo experimental para permitir comparação direta com os demais algoritmos, utilizando as mesmas séries, janelas, taxas-alvo e métricas.

É importante destacar a convenção de contagem adotada daqui em diante, já que três números diferentes aparecem ao longo do texto para descrever o mesmo conjunto: são quatro as **abordagens** de compressão avaliadas (SDT, RDP, DCT e DWT); são seis as **variantes** da primeira fase, pois a DWT entra com três famílias de wavelets; e são sete os **algoritmos** comparados no total, somando o ARC-SDT proposto na segunda fase. Sempre que o texto mencionar os sete algoritmos, trata-se do conjunto

completo apresentado nas figuras e tabelas do Capítulo 6.

5.2 Coleções

5.2.1 Fontes e Origem

Os experimentos foram realizados sobre duas coleções de séries temporais de consumo de energia elétrica, resumidas na Tabela 5.1:

Tabela 5.1 — Coleções de dados utilizadas na avaliação comparativa

Coleção	Arquivos	Origem
Pública	167	Fontes acadêmicas: RAE, EcoData, GreenD, Indian Smart Meter, COMBED, Dryad, IDEAL, UKDALE, Smart*
Privada	89	Instalações brasileiras (dados coletados localmente)

Fonte: elaborada pelo autor.

As coleções públicas foram obtidas de repositórios acadêmicos disponibilizados por iniciativas de pesquisa em monitoramento de energia. Os dados privados foram coletados a partir de medidores instalados em instalações brasileiras, que exportam registros no formato CSV com colunas de data/hora e potência ativa total. A combinação das duas coleções garante diversidade tanto de perfis de consumo quanto de contextos geográficos e tipos de instalação.

As duas coleções são mantidas separadas ao longo de toda a análise e os resultados são reportados individualmente para cada uma. A motivação e o método dessa separação são descritos na Seção 5.7.2.

Embora o caso de uso considerado neste trabalho monitore cerca de 30 grandezas elétricas por leitura, a análise comparativa dos algoritmos de compressão foca exclusivamente na potência ativa. Essa escolha decorre de um conjunto de razões convergentes. Do ponto de vista dos dados disponíveis, as coleções públicas utilizadas disponibilizam predominantemente séries de potência ativa (a maioria das demais grandezas não está presente nesses conjuntos em resolução e duração compatíveis com os experimentos), e o trabalho apresentado na qualificação já adotava essa grandeza como objeto central da análise. Do ponto de vista do comportamento do sinal, incluir as demais grandezas acrescentaria pouco à comparação entre algoritmos: a tensão tende a se manter próxima dos valores nominais, com variação

pequena, e é comprimida com folga por praticamente qualquer técnica, o que faz os algoritmos se diferenciarem pouco quando aplicados a ela; a corrente, por sua vez, tende a acompanhar a potência ativa em instalações com fator de potência relativamente estável, de modo que a análise de uma delas implica comportamento similar na outra. Do ponto de vista da relevância prática, a potência ativa é a grandeza de maior interesse para o monitoramento de consumo, conforme discutido na introdução. Por fim, do ponto de vista metodológico, avaliar os algoritmos sobre uma única grandeza evita confundir o desempenho de compressão com diferenças de dinâmica entre variáveis: uma avaliação comparativa que misturasse tensão, corrente e potência estaria avaliando ao mesmo tempo os algoritmos e as características de cada sinal. Deste modo, a potência ativa representa o caso mais desafiador e mais informativo para a avaliação comparativa: combina variações lentas de consumo de base com picos abruptos de carga, exigindo dos algoritmos tanto fidelidade global quanto capacidade de preservar eventos pontuais.

Todas as coleções utilizadas têm resolução de 1 minuto por leitura. O intervalo de 15 minutos é o padrão consolidado para medição de demanda em sistemas de faturamento, regulamentado pela ANEEL para apuração da demanda contratada em medidores inteligentes; entretanto, medições integradas nesse intervalo são insuficientes para capturar variações de consumo e eventos de curta duração. A resolução de 1 minuto permite identificar padrões de demanda intrahorária e detectar eventos recorrentes associados a ciclos de produção e hábitos de consumo, sendo o menor intervalo amplamente disponível nas coleções públicas de monitoramento energético.

5.2.2 Descrição das Coleções Públicas

Os 167 dias da coleção pública foram extraídos de nove fontes acadêmicas, selecionadas por disponibilizarem séries de potência ativa com resolução de ao menos um minuto e formato adequado para extração de dias completos de 1.440 pontos.

UKDALE (*UK Domestic Appliance-Level Electricity*): coleção residencial britânica com medições de potência ativa total e por aparelho para cinco residências ao longo de meses a anos, com resolução de 6 segundos para a medição total e 1/16 Hz para os aparelhos individuais [29]. Para este trabalho, utilizou-se a série de potência ativa total, agregada para 1 minuto.

Smart*: conjunto de dados de três residências instrumentadas no campus da Universidade de Massachusetts Amherst, com medições de potência ativa total e por circuito coletadas em 2012 com frequência de 1 Hz [11]. A série de potência total foi integrada em intervalos de 1 minuto.

IDEAL: coleção residencial britânica coletada em aproximadamente 255 domicílios na Escócia e norte da Inglaterra, com dados de energia elétrica, gás, temperatura e condições ambientais [40]. A potência ativa total foi extraída com granularidade de 1 minuto.

COMBED (*Commercial Building Energy Dataset*): medições de potência ativa em prédios comerciais do Instituto de Tecnologia de Delhi (IIT-D), com resolução de 1 minuto [12]. O contexto comercial/universitário resulta em perfis de consumo distintos dos residenciais, com ocupação horária previsível.

RAE (*Rainforest Automation Energy*): dados de residências canadenses coletados por equipamentos da empresa Rainforest Automation, disponibilizados publicamente [36]. Contém séries de potência ativa total com resolução de 1 minuto.

EcoData: coleção de consumo residencial europeia com medições de potência ativa e reativa em granularidade de 1 Hz, coletadas em residências suíças [23]. Para este trabalho, a série de potência ativa foi agregada para 1 minuto.

GreenD: coleção de consumo residencial alemã com medições de potência ativa e reativa em múltiplas frequências [39]. As séries de potência ativa foram agregadas para 1 minuto.

Indian Smart Meter: medições de consumo elétrico em instalações residenciais e comerciais indianas, com granularidade sub-minuto ou de 1 minuto [26]. As séries com resolução inferior a 1 minuto foram agregadas por média ao intervalo de 1 minuto antes da inclusão no experimento; nenhuma interpolação foi realizada, de modo que todos os pontos da série agregada correspondem a médias de medições reais.

Dryad: repositório de dados científicos de acesso aberto. Diversas séries temporais de consumo elétrico com resolução de 1 minuto foram obtidas de depósitos individuais hospedados nesse repositório, provenientes de estudos publicados em diferentes contextos geográficos [22].

5.2.3 Descrição dos Dados Privados

Os 89 dias da coleção privada foram coletados a partir de medidores instalados em instalações de uma empresa parceira do Laboratório de Sistemas Distribuídos (LSD/UFCG) em projetos de pesquisa e desenvolvimento (P&D). Os dados foram coletados pelo autor e colaboradores do grupo de pesquisa, utilizando medidores de energia que exportam registros em formato CSV com colunas de data/hora e potência ativa total, com resolução de 1 minuto. Por questões de proteção dos dados da empresa parceira, os locais de coleta não são identificados e os dados não são disponibilizados publicamente. Os perfis de consumo refletem ambientes industriais e comerciais brasileiros, com padrões distintos das coleções residenciais predominantes nas coleções públicas.

5.2.4 Formato dos Arquivos

Cada arquivo representa exatamente um dia completo de medição de uma instalação. O formato adotado é um *array* JSON de 1.440 valores de potência em Watts (`float`), com leituras espaçadas de um minuto. O nome de cada arquivo segue o padrão:

`[Nome]-[DataBase]-[DiaMedido].json`

onde `Nome` identifica a instalação ou fonte de dados, `DataBase` é a data de referência do arquivo de origem e `DiaMedido` é a data do dia extraído. Esse padrão permite que múltiplos dias de uma mesma instalação sejam armazenados como arquivos independentes, cada um com exatamente 1.440 pontos.

5.2.5 Processo de Aquisição dos Dados Privados

Os arquivos CSV brutos dos dados privados possuem separador ponto e vírgula, decimal por vírgula e codificação Latin-1, padrão comum em equipamentos de medição comercializados no Brasil. O processo de tratamento envolve duas etapas:

1. **Validação:** para cada arquivo CSV, os registros são agrupados por dia e contados após remoção de entradas nulas. Apenas dias com exatamente 1.440 leituras válidas são considerados elegíveis. Dias com qualquer lacuna de medição são descartados integralmente;
2. **Conversão:** os dias elegíveis são extraídos, os valores de potência são ordenados cronologicamente e salvos como *arrays* JSON, um arquivo por dia.

O critério de 1.440 leituras exatas assegura que todos os algoritmos sejam avaliados sobre exatamente a mesma grade temporal. Padronizar dias completos mantém a comparação restrita às diferenças entre os algoritmos, sem introduzir o tratamento de lacunas como variável adicional do experimento.

5.2.6 Representação Interna das Séries

Antes da execução dos algoritmos, cada série JSON é convertida para uma lista de pares `(timestamp, valor)`, onde o *timestamp* é um número inteiro representando segundos desde a época Unix (UTC), calculado a partir da data indicada no nome do arquivo com fuso horário UTC-3. A primeira leitura recebe o *timestamp* correspondente à meia-noite do dia medido, e cada leitura subsequente é deslocada em 60 segundos em relação à anterior.

Essa representação é necessária porque os algoritmos geométricos (SDT, RDP, ARC-SDT) operam no espaço tempo-valor e utilizam os *timestamps* diretamente como eixo x para o cálculo de inclinações.

Os algoritmos espectrais (DCT e DWT) utilizam apenas os valores de potência, ignorando os *timestamps*, mas a estrutura uniforme de pares permite que todos os compressores compartilhem a mesma interface de entrada.

5.3 Ambiente e Implementação

5.3.1 Linguagem e Ambiente de Execução

Todos os algoritmos foram implementados em Python 3.10, utilizando um ambiente virtual isolado (`venv`) para controle preciso das dependências. A escolha do Python se justifica por três razões principais. Primeira, a disponibilidade de bibliotecas científicas maduras e amplamente validadas para as operações necessárias na avaliação comparativa, como transformadas espectrais e álgebra linear. Segunda, o Python é a linguagem de referência em pesquisa de aprendizado de máquina e processamento de sinais, o que facilita a comparação e reprodução dos resultados. Terceira, o objetivo do experimento é comparar os algoritmos em condições controladas e igualitárias; a linguagem não é o alvo da otimização, e o Python oferece o ambiente mais neutro para essa finalidade. Em cenários de produção, algoritmos destinados a dispositivos embarcados seriam reimplementados em linguagens de baixo nível como C ou C++, mas esse não é o escopo deste trabalho.

5.3.2 Bibliotecas Utilizadas

As principais bibliotecas externas utilizadas nas implementações são descritas a seguir:

NumPy [25], na versão 2.2.6, é a biblioteca fundamental de computação numérica para Python, fornecendo *arrays* multidimensionais com operações vetorizadas executadas em código C. É utilizada na DCT e na DWT para manipulação dos *arrays* de coeficientes, normalização dos sinais e seleção dos K maiores coeficientes.

SciPy [47] (*Scientific Python*), na versão 1.15.3, é uma biblioteca de computação científica que estende o NumPy com algoritmos para integração, otimização, processamento de sinais e transformadas. Na avaliação comparativa, é utilizada especificamente para a DCT e sua inversa via `scipy.fftpack.dct` e `idct`, que implementam a DCT-II com normalização ortogonal (`norm='ortho'`) usando algoritmos FFT (*Fast Fourier Transform*) eficientes. A SciPy também é empregada na detecção de picos via `scipy.signal.find_peaks`, usada no cálculo das métricas de pico.

PyWavelets [32] (`pywt`), na versão 1.8.0, é a biblioteca Python de referência para transformadas

wavelet, amplamente utilizada na comunidade de processamento de sinais. Fornece implementações das famílias wavelet mais comuns com suporte a decomposição multinível (`wavedec`) e reconstrução (`waverec`). É o componente central da implementação do compressor DWT.

scikit-image [46], na versão 0.25.2, é uma biblioteca de processamento de imagens para Python que disponibiliza, entre outros algoritmos, a implementação da métrica SSIM (*Structural Similarity Index*), utilizada no cálculo das métricas de qualidade de reconstrução.

pandas [38], na versão 2.3.3, é uma biblioteca para manipulação de dados tabulares e séries temporais em Python, utilizada internamente no módulo de métricas para representar as séries como objetos `Series`, simplificando as operações de agregação e comparação elemento a elemento.

5.3.3 Interface Uniforme dos Compressores

Para garantir que o experimento possa iterar sobre os algoritmos de forma genérica e sistemática, todos os compressores implementam a mesma interface:

- **Entrada:** lista de pares `(timestamp: int, valor: float)`;
- **Saída:** lista de pares reconstruídos no mesmo formato;
- **Atributos expostos após a compressão:** `compression_ratio` (CR real atingido em %), `execution_time` (tempo em segundos) e `metrics` (dicionário com todas as métricas de qualidade).

Essa uniformidade permite que o laço principal do experimento chame qualquer compressor da mesma forma, sem ramificações específicas por algoritmo além da instanciação e do mecanismo de controle do CR.

5.3.4 Implementação dos Algoritmos

SDT e RDP. Os algoritmos de reamostragem geométrica foram implementados durante o desenvolvimento desta pesquisa, em Python puro, sem uso de bibliotecas externas para o núcleo dos algoritmos. O SDT segue fielmente a descrição original do algoritmo. A lógica central é composta por uma classe interna que mantém os dois pivôs, as inclinações acumuladas e `ultimo_valido`, atualizando o estado a cada nova amostra em tempo $\mathcal{O}(1)$ e com uso de memória $\mathcal{O}(1)$, já que todo o estado se resume a esses escalares. O RDP segue diretamente o algoritmo descrito por Ramer [43] e por Douglas e Peucker [21], implementado de forma recursiva.

DCT. A DCT utiliza a implementação da biblioteca SciPy (`scipy.fftpack.dct`), que aplica a DCT-II com normalização ortogonal via algoritmos FFT. O sinal é convertido para `float32` e normalizado para o intervalo $[0, 1]$ antes da transformada, com os valores mínimo e máximo armazenados como metadados para a reversão posterior. O número de coeficientes a manter (K) é calculado a partir de um modelo de custo de transmissão detalhado na Seção 5.4.4.

DWT. A DWT utiliza a biblioteca PyWavelets (`pywt`), aplicando decomposição multinível com `pywt.wavedec` e reconstrução com `pywt.waverec`. O mesmo processo de normalização da DCT é aplicado. Os coeficientes de todos os níveis são linearizados em um único vetor com `pywt.coeffs_to_array`, que também retorna a estrutura de *slices* necessária para a reconstrução.

5.3.5 Complexidade Computacional

A Tabela 5.2 resume a complexidade de tempo e de memória de cada algoritmo. A notação $\mathcal{O}(f(N))$ indica como o custo cresce em função do número de pontos N da janela: $\mathcal{O}(1)$ significa custo constante independente de N; $\mathcal{O}(N)$ significa crescimento proporcional; $\mathcal{O}(N \log N)$ significa crescimento ligeiramente acima do proporcional.

Tabela 5.2 — Complexidade algorítmica dos algoritmos avaliados

Algoritmo	**Tempo**	**Memória extra**	**Modo**
SDT	$\mathcal{O}(N)$	$\mathcal{O}(1)$	Amostra a amostra
ARC-SDT	$\mathcal{O}(N)$	$\mathcal{O}(1)$	Amostra a amostra
RDP	$\mathcal{O}(N \log N)$ médio	$\mathcal{O}(N)$	Em lote
DCT	$\mathcal{O}(N \log N)$	$\mathcal{O}(N)$	Em lote
DWT	$\mathcal{O}(N)$	$\mathcal{O}(N)$	Em lote

Fonte: elaborada pelo autor.

SDT e ARC-SDT processam cada nova amostra no momento em que ela chega, mantendo apenas um estado mínimo: os dois pivôs do corredor, as inclinações acumuladas e, no caso do ARC-SDT, as duas variáveis do controlador PI. Esse estado não cresce com o número de pontos processados: comprimir uma série de 1.440 pontos ou de 1.440.000 pontos exige a mesma quantidade de memória. Não há *buffer* de histórico.

Os demais algoritmos operam em lote: precisam que toda a janela esteja disponível em memória antes de começar. O RDP percorre a série inteira em busca do ponto mais distante, divide o problema recursivamente e acumula uma pilha de chamadas proporcional ao número de pontos. A DCT transforma o sinal inteiro via FFT antes de selecionar coeficientes. A DWT aplica filtros em cascata sobre a janela completa. Em todos esses casos, o tamanho da janela de processamento determina diretamente o requisito mínimo de memória do algoritmo (o que, em microcontroladores com memória da ordem de kilobytes, é uma restrição prática significativa). Além disso, algoritmos em lote impõem uma latência de notificação mínima igual ao tamanho da janela: um evento que ocorre no início da janela só é detectado e pode ser transmitido após o processamento de todos os pontos seguintes. Algoritmos amostra a amostra não têm essa limitação, pois cada novo ponto é avaliado à medida que chega, permitindo notificação imediata quando uma tendência muda.

5.4 Protocolo Experimental

5.4.1 Algoritmos Avaliados

As seis variantes da primeira fase cobrem duas famílias de abordagem distintas quanto ao domínio de representação e ao modo de operação:

- **Baseados em geometria de curvas:** SDT e RDP preservam pontos com base no desvio em relação a uma aproximação linear entre *keypoints*, com reconstrução por interpolação linear. O SDT processa amostra a amostra (um ponto por vez); o RDP opera em lote (requer a série completa antes de começar);
- **Baseados em transformadas espectrais:** DCT e DWT transformam o sinal para um domínio de coeficientes e descartam os de menor magnitude. Ambos operam em lote, sobre blocos de tamanho fixo.

A DWT é avaliada com as famílias haar, sym2 e bior5.5, resultando em seis variantes de algoritmo no total da primeira fase. O ARC-SDT, acrescentado na segunda fase, integra o mesmo conjunto de comparações com protocolo idêntico.

5.4.2 Janelamento

Cada série de 1.440 pontos é dividida em janelas não sobrepostas de tamanho fixo antes de ser processada. A série é particionada em blocos consecutivos; a última janela é descartada se seu comprimento for

inferior ao tamanho configurado. Isso garante que todos os algoritmos recebam janelas de comprimento idêntico, evitando que efeitos de borda em janelas incompletas contaminem as métricas. A Tabela 5.3 apresenta os tamanhos avaliados e o horizonte temporal que cada um representa.

Tabela 5.3 — Tamanhos de janela avaliados e interpretação temporal

Janela (pontos)	**Janelas por série**	**Duração**
15	96	15 minutos
60	24	1 hora
360	4	6 horas
720	2	12 horas
1440	1	1 dia completo

Fonte: elaborada pelo autor.

É importante destacar que o janelamento não implica trabalhar com conjuntos de dados diferentes: o dado de referência é sempre o dia completo de 1.440 pontos. O que varia é a forma como esse dia é apresentado ao algoritmo. Com janela de 1.440 pontos, o dia inteiro é processado de uma vez; com janela de 15 pontos, o mesmo dia é processado em 96 fatias consecutivas. Para permitir comparação justa entre configurações, os resultados são sempre consolidados ao nível do dia completo: a taxa de compressão é a média das taxas obtidas em cada fatia.

A escolha de múltiplos tamanhos de janela serve a três propósitos complementares.

O primeiro é avaliar o efeito da granularidade temporal no desempenho dos algoritmos: janelas menores isolam trechos mais curtos e potencialmente mais homogêneos do sinal, enquanto janelas maiores capturam variações ao longo de períodos mais longos do dia, expondo padrões de consumo como o período noturno de baixo consumo e picos matinais.

O segundo propósito é explorar a relação entre tamanho de bloco e qualidade dos algoritmos espectrais: a DCT e a DWT dependem da disponibilidade de coeficientes para representar o sinal, e janelas muito curtas (15 pontos) limitam essa capacidade, tornando difícil atingir altas taxas de compressão com boa qualidade.

O terceiro propósito, diretamente relevante para o contexto de dispositivos embarcados, é determinar quanto dado precisa ser armazenado em memória para que cada algoritmo opere de forma satisfatória. Algoritmos em lote como RDP, DCT e DWT exigem que toda a janela esteja disponível em memória antes que a compressão possa iniciar. Algoritmos amostra a amostra como SDT e ARC-SDT, por ou-

tro lado, processam cada amostra à medida que ela chega e não requerem nenhum *buffer* de histórico, podendo operar com uso de memória praticamente constante independentemente do tempo de coleta. O janelamento é, portanto, também um parâmetro que define o requisito mínimo de memória de cada abordagem, informação essencial para avaliar a viabilidade de cada algoritmo em hardware embarcado com memória restrita.

5.4.3 Taxas de Compressão Alvo

Cada algoritmo é avaliado em nove taxas de compressão alvo, variando de 10% a 90% em incrementos de 10%. Essa faixa cobre desde compressões conservadoras, onde a maioria das informações é preservada, até compressões agressivas, onde apenas uma fração pequena do dado original é mantida. A cobertura uniforme com passos iguais permite identificar de forma sistemática em que ponto cada algoritmo começa a degradar a qualidade de forma significativa e como cada um se comporta nos extremos da faixa.

5.4.4 Controle da Taxa de Compressão

Os algoritmos diferem na forma como atingem a taxa de compressão alvo, o que exige estratégias distintas:

DCT e DWT. O número de coeficientes K a manter é calculado algebricamente a partir do CR alvo, com o mesmo modelo de custo para as duas transformadas (Seção 3.1 e Seção 3.4.5 do Capítulo 3). O dado original é modelado como um par (*timestamp*, valor) por ponto, ocupando $N \times 8$ bytes, e o dado comprimido deve ocupar $(1 - \texttt{CR}/100) \times N \times 8$ bytes. Desse orçamento subtrai-se um *overhead* fixo de $3 \times 4 = 12$ bytes, correspondente aos metadados obrigatórios para reconstrução (valor mínimo, valor máximo e *timestamp* inicial da janela); o tamanho da janela, o nível de decomposição e a família de wavelet são parâmetros combinados a priori entre emissor e receptor e não entram nesse orçamento. Em ambas as transformadas, os coeficientes mantidos são selecionados por magnitude, e não pelas primeiras posições da transformada: cada um exige o envio do valor e do índice original, ao custo de 8 bytes por coeficiente. O número de coeficientes mantidos é $K = \max(1, \lfloor(\text{orçamento disponível})/8\rfloor)$, e o CR real pode diferir ligeiramente do alvo em função do arredondamento de K para inteiro.

SDT e RDP. Esses algoritmos não expõem controle direto sobre a taxa de compressão: o parâmetro de entrada é a tolerância de erro (`error` para o SDT, `epsilon` para o RDP), e o CR resultante depende da

variabilidade local do sinal. Para permitir a comparação nas mesmas taxas-alvo dos demais algoritmos, aplica-se busca binária sobre o parâmetro de tolerância:

1. O intervalo de busca é inicializado com $[\texttt{lo}, \texttt{hi}] = [0{,}1,\ 10^7]$, cobrindo desde tolerâncias muito pequenas até tolerâncias muito grandes;
2. A cada iteração, calcula-se $\texttt{mid} = (\texttt{lo}+\texttt{hi})/2$, executa-se o compressor com esse valor e mede-se o CR real obtido;
3. Se $|\texttt{CR}_{\text{real}} - \texttt{CR}_{\text{alvo}}| \leq 2{,}0\%$, a busca é encerrada antecipadamente;
4. Caso contrário, o intervalo é ajustado: se $\texttt{CR}_{\text{real}} < \texttt{CR}_{\text{alvo}}$, a tolerância é muito conservadora e `lo` é atualizado para `mid`; se $\texttt{CR}_{\text{real}} > \texttt{CR}_{\text{alvo}}$, a tolerância é excessiva e `hi` é atualizado;
5. O processo é limitado a 30 iterações. O melhor resultado obtido ao longo das iterações é utilizado mesmo que a tolerância de 2% não tenha sido atingida, o que pode ocorrer em janelas de curta duração onde a granularidade do CR é limitada.

ARC-SDT. O controle é feito de forma adaptativa durante a própria execução do algoritmo, por um controlador PI que ajusta o parâmetro de tolerância a cada ponto processado. Não é necessária busca externa. Os detalhes do mecanismo de controle são descritos no capítulo dedicado ao ARC-SDT.

5.4.5 Ordem de Execução e Agregação dos Resultados

O experimento itera nas seguintes dimensões, nesta ordem: instalação (nome), data, taxa de compressão alvo, tamanho de janela, algoritmo. Para cada combinação, a série do dia é particionada em janelas e o algoritmo é executado individualmente sobre cada janela. Esse esquema garante que, para cada configuração (taxa-alvo, janela, algoritmo), o resultado reflita o comportamento médio ao longo do dia, e não apenas um trecho isolado.

Os resultados de cada janela são coletados e, ao final, agregados por combinação da seguinte forma:

- **CR real:** média aritmética dos CRs obtidos em cada janela;
- **Métricas de qualidade:** média aritmética sobre as janelas, desconsiderando valores `None`. Valores indefinidos ocorrem quando a métrica não se aplica a uma janela específica, como o MAPE em janelas com amostras de valor zero ou o PeakRecall em janelas sem picos detectáveis.

Cada combinação agrega seus resultados e os escreve imediatamente no arquivo CSV de saída antes de avançar para a próxima. Exceções em combinações individuais são registradas e reportadas ao final, sem interromper o experimento.

5.5 Métricas de Avaliação

As métricas utilizadas para avaliar a qualidade de reconstrução, cobrindo fidelidade de forma, fidelidade energética e preservação de picos, são definidas e discutidas na Seção 2.4 da fundamentação teórica. No contexto do experimento, cada métrica é calculada por janela após a reconstrução do sinal e, ao final, agrega-se a média aritmética sobre todas as janelas da série, desconsiderando valores indefinidos.

5.6 Combinações e Volume Experimental

O experimento cobre o produto cartesiano de algoritmos, taxas-alvo e tamanhos de janela, conforme a Tabela 5.4:

Tabela 5.4 — Dimensões do experimento e total de combinações

Dimensão	**Valores**	**Qtd.**
Algoritmos (fase 1)	SDT, RDP, DCT, DWT-haar, DWT-bior5.5, DWT-sym2	6
Algoritmos (fase 2)	ARC-SDT (adicionado)	+1
Taxas-alvo	10%, 20%, …, 90%	9
Tamanhos de janela	15, 60, 360, 720, 1.440 pontos	5

Fonte: elaborada pelo autor.

Com 7 algoritmos, cada série gera $7 \times 9 \times 5 = 315$ combinações de configuração. Sobre as 256 séries disponíveis (167 públicas + 89 privadas), o experimento completo totaliza aproximadamente 80.640 execuções individuais de compressão. Para o SDT e o RDP, cada uma dessas execuções pode envolver até 30 compressões durante a busca binária, elevando o número real de chamadas aos algoritmos para mais de 1,6 milhão no total do experimento.

5.7 Estratégia de Análise dos Resultados

5.7.1 Natureza Exploratória da Comparação

A comparação entre algoritmos conduzida neste trabalho tem caráter exploratório e descritivo. O objetivo não é estabelecer um ranqueamento estatisticamente significativo entre os sete algoritmos, mas identificar

tendências: quais abordagens tendem a se sair melhor em cada dimensão de fidelidade (forma, picos, energia e erro relativo ponto a ponto) e sob quais condições (taxa-alvo, tamanho de janela e tipo de coleção). Para esse propósito, a mediana de cada métrica sobre o conjunto de séries, acompanhada da dispersão observada nos *boxplots* e do comportamento das curvas ao longo das taxas de compressão, oferece uma leitura suficiente e interpretável do desempenho relativo de cada abordagem.

Deste modo, este trabalho não aplica testes de inferência estatística (como o teste de Wilcoxon pareado ou o de Friedman com pós-teste de Nemenyi) nem reporta tamanhos de efeito formais. Essa escolha é deliberada e coerente com o objetivo proposto: quando duas medianas ficam muito próximas, o texto as trata como equivalentes na prática, e não como vantagem de um algoritmo sobre o outro. As conclusões apoiam-se em diferenças consistentes e de magnitude expressiva, que se mantêm ao longo de múltiplas taxas de compressão e nas duas coleções, e não em desempates finos entre valores próximos. A verificação formal dessas tendências por meio de testes não paramétricos pareados, com o devido controle de comparações múltiplas, é um desdobramento natural deste trabalho e é registrada entre as direções futuras (Capítulo 7).

5.7.2 Análise Separada das Coleções Pública e Privada

Toda a análise comparativa é conduzida separadamente para a coleção pública e para a coleção privada, e os resultados são reportados de forma independente para cada uma. Essa separação decorre de uma preocupação metodológica fundamental: os dados privados foram coletados em instalações específicas às quais o pesquisador teve acesso, e não passaram por um processo de curadoria independente como os dados públicos. Existe, portanto, o risco de que os perfis de consumo dessas instalações sejam sistematicamente diferentes dos da coleção pública, por exemplo, por concentrarem ambientes industriais com consumo mais regular, ou por refletirem padrões de uso atípicos de uma região ou setor específico.

Se os resultados fossem agregados diretamente entre as duas coleções, qualquer viés nos dados privados se propagaria para as conclusões sem possibilidade de diagnóstico. Manter as análises separadas permite comparar o ranqueamento relativo dos algoritmos em cada coleção: se a ordem de desempenho for consistente entre pública e privada, isso indica que a coleção privada não introduz tendências que favoreçam artificialmente nenhum algoritmo. Se, por outro lado, diferenças significativas forem observadas, elas podem ser investigadas em termos das características dos perfis de consumo, o que em si já é um resultado de interesse sobre a sensibilidade dos algoritmos a diferentes tipos de sinal.

A análise da coleção pública é considerada o resultado primário da avaliação comparativa, por ser reproduzível por qualquer pesquisador a partir das fontes originais. A análise da coleção privada funciona como validação em dados do contexto de aplicação brasileiro e, quando consistente com a pública,

reforça a generalidade das conclusões.

5.7.3 Seleção das Métricas de Análise

O experimento coleta treze métricas de qualidade por execução. Analisar todas individualmente tornaria este trabalho excessivamente extenso sem acrescentar perspectivas novas, pois diversas métricas capturam aspectos sobrepostos da fidelidade do sinal. Quatro métricas foram selecionadas com base nas perguntas operacionais relevantes para o domínio de monitoramento de consumo de energia elétrica: fidelidade de forma (SSIM), preservação de picos de demanda (PeakRecall), fidelidade energética (EnergyError) e erro relativo ponto a ponto (MAPE). A justificativa detalhada para cada escolha é apresentada no início do Capítulo 6.

5.7.4 Tratamento de Outliers

Certos perfis de consumo elétrico apresentam longos períodos de potência constante ou quase constante, como instalações desocupadas, equipamentos em modo de espera ou medições noturnas. Para o SDT, esses sinais representam um cenário excepcionalmente favorável: a baixa variabilidade faz com que qualquer tolerância de erro não nula acomode longas sequências em um único segmento, produzindo taxas de compressão muito superiores ao alvo e erro de reconstrução nulo. Trata-se de uma vantagem real do algoritmo (e não de uma distorção do experimento) que vem da natureza geométrica do SDT: quanto mais regular a tendência do sinal, mais pontos são descartados sem comprometer a fidelidade da reconstrução linear.

Para efeitos de avaliação comparativa, no entanto, esse comportamento cria um desequilíbrio: os algoritmos espectrais e geométricos que operam em lote processam a janela completa e selecionam pontos ou coeficientes para atingir o orçamento configurado, independentemente da regularidade do sinal, sem o mesmo benefício estrutural. Incluir essas séries na comparação tornaria a análise menos representativa do desempenho de cada abordagem em condições equivalentes.

Para lidar com esse desequilíbrio, a análise é apresentada em duas versões. Na primeira, consideram-se todas as séries, permitindo observar o SDT (e o ARC-SDT) em seu melhor cenário de uso. Na segunda, removem-se as séries de alta compressibilidade natural, identificadas por busca binária de tolerância sobre o SDT puro mirando CR = 80% (o mesmo mecanismo de busca usado para o SDT e o RDP em toda a avaliação comparativa): se, mesmo com a busca ativamente tentando aproximar o CR do alvo, o resultado obtido ultrapassa 85% (uma margem de 5% acima do alvo), a série é considerada de alta compressibilidade natural e excluída do conjunto filtrado. O critério é direto e não circular, pois

usa apenas o SDT base como medida de compressibilidade da série, independente dos parâmetros de qualquer controlador. Essa separação evidencia o desempenho dos algoritmos tanto nas condições que favorecem o SDT quanto nas condições de variabilidade típica.

Além dessa filtragem, execuções com CR negativo são excluídas da análise antes de qualquer agregação, como proteção geral: CR negativo indicaria que o dado comprimido ocupa mais espaço do que o original (ou seja, o algoritmo falhou em comprimir). Com o modelo de custo em bytes da DCT e da DWT descrito na Seção 3.4.5, em que o *overhead* de metadados é fixo e não depende do tamanho da janela, esse caso não ocorre em nenhuma das execuções deste trabalho, nem mesmo na menor janela avaliada (15 pontos): o CR real acompanha o alvo configurado até 70%, saturando em torno de 83% para alvos de 80% e 90% em função do piso mínimo de 1 coeficiente mantido, mas nunca se torna nulo ou negativo.

5.7.5 Visualizações e Comparações

A análise comparativa foi estruturada em torno das seguintes visualizações:

Curvas CR real vs. métrica de qualidade. Para cada métrica selecionada, gera-se um gráfico de linhas com o CR real no eixo x e a métrica no eixo y, com uma linha por algoritmo. Cada ponto representa a mediana sobre todas as séries para aquele CR alvo. Esse gráfico é o principal instrumento de comparação: permite visualizar diretamente o custo-benefício entre compressão e qualidade para cada abordagem e identificar em que ponto cada algoritmo começa a degradar significativamente.

Efeito do tamanho de janela. Para cada algoritmo, gera-se um conjunto de curvas CR vs. SSIM separado por tamanho de janela. O objetivo é identificar se e em que medida o desempenho de cada algoritmo depende do volume de dados disponível por bloco.

Tabela resumo. Uma tabela com a mediana de cada métrica selecionada por algoritmo, fixando um CR alvo representativo (por exemplo, 80%), oferece uma visão consolidada para comparação direta e complementa os gráficos de curva.

Distribuição por *boxplot*. Para análises onde a dispersão é relevante, como na comparação antes e após a remoção de *outliers*, *boxplots* das métricas por algoritmo revelam não apenas a mediana, mas também a variabilidade e a presença de valores extremos.

5.7.6 Avaliação do ARC-SDT

O ARC-SDT participa da comparação principal com os demais algoritmos, usando o mesmo protocolo experimental e as mesmas métricas. Nesse contexto, é avaliado como mais um competidor nas curvas

CR vs. qualidade e nas tabelas resumo, permitindo verificar se a proposta entrega qualidade competitiva em relação às abordagens existentes.

No entanto, por ser uma proposta nova com comportamento dinâmico, o ARC-SDT também recebe uma avaliação isolada centrada em aspectos que os demais algoritmos não possuem.

Fidelidade de controle. A principal afirmação do ARC-SDT é que o controlador PI ajusta o parâmetro de erro do SDT para que a taxa de compressão real se aproxime da taxa alvo configurada. Essa afirmação é avaliada diretamente: para cada combinação de série e CR alvo, mede-se a diferença entre o CR real obtido e o CR alvo. Gráficos de dispersão e a distribuição dessa diferença revelam se o controlador é eficaz em diferentes condições e em que situações ele tende a errar mais.

Comparação direta com o SDT. O SDT é o algoritmo base do ARC-SDT, e os dois compartilham o mesmo mecanismo de compressão. A diferença está no controle da taxa. Isolar essa comparação permite quantificar o ganho específico trazido pelo controle adaptativo: o ARC-SDT produz CR mais próximo do alvo com qualidade equivalente? Ou paga algum custo em qualidade para ganhar controlabilidade? Essa comparação é apresentada separadamente das demais para tornar essa troca explícita.

5.8 Saída e Reprodutibilidade

Os resultados são gravados de forma incremental em arquivo CSV, com uma linha por combinação (instalação, data, algoritmo, taxa-alvo, tamanho de janela). A gravação ocorre imediatamente após cada combinação ser processada e seus resultados agregados, sem acumulação em memória. Essa estratégia garante que experimentos interrompidos não percam os resultados já obtidos, e possam ser retomados filtrando as combinações ainda não presentes no arquivo.

São gerados dois arquivos separados: um para a coleção pública e outro para a coleção privada. Cada linha contém os campos de identificação (instalação, data, método, algoritmo, taxa-alvo, tamanho de janela), o CR real atingido e os valores de todas as métricas de qualidade de reconstrução.

Quanto à reprodutibilidade, o experimento é inteiramente determinístico: nenhuma etapa recorre a sorteio ou inicialização estocástica, de modo que não há semente aleatória a registrar e a reexecução sobre os mesmos dados reproduz os mesmos valores. As versões das bibliotecas ficam registradas na Seção 5.3.2, o que importa porque a DCT e a DWT dependem de implementações de terceiros. Os dois arquivos CSV são o único insumo da análise do Capítulo 6: cada figura e cada tabela é gerada por um script que os lê sem repetir nenhuma compressão, de modo que os números relatados podem ser conferidos sem executar o experimento de novo. Os scripts de análise, os arquivos CSV de resultados e os dados da Coleção Privada podem ser requisitados ao autor.

Capítulo 6

Avaliação dos Algoritmos de Compressão

6.1 Seleção das Métricas de Análise

O experimento coleta treze métricas de qualidade por execução. Analisar todas individualmente tornaria este trabalho excessivamente extenso e dificultaria a interpretação dos resultados, pois diversas métricas capturam aspectos sobrepostos da fidelidade do sinal. Sendo assim, quatro métricas foram selecionadas com base nas perguntas operacionais que um sistema de monitoramento de consumo de energia elétrica precisaria responder após a descompressão:

- **SSIM**: fidelidade de forma do sinal: avalia padrões locais de consumo ao longo do tempo, comparando, em cada trecho, o nível médio, a amplitude das variações e a correlação entre os padrões de subida e descida do sinal original e do reconstruído. Para obter SSIM alto, não basta ter erro médio pequeno; é preciso que os picos e vales ocorram no momento certo e com amplitude proporcional ao contexto local;

- **PeakRecall**: preservação dos picos de demanda: informa diretamente se os eventos de pico de consumo foram mantidos após a compressão. Picos de demanda determinam o enquadramento tarifário em sistemas de cobrança por demanda contratada e são eventos de interesse primário para gestão de carga e proteção de instalações;

- **EnergyError**: fidelidade energética: mede o erro na energia acumulada ao longo do período. A energia integrada é a grandeza que entra diretamente no faturamento de consumo e na análise de eficiência energética; um algoritmo que distorça essa grandeza seria inaceitável em aplicações de medição para faturamento;

- **MAPE**: erro relativo ponto a ponto: captura o erro percentual médio em cada amostra, sendo

particularmente sensível a distorções em trechos de baixo consumo, onde os erros absolutos são pequenos mas percentualmente relevantes. Complementa o SSIM ao revelar distorções locais que podem não impactar a forma global mas afetam a leitura pontual do nível de consumo.

Cada uma dessas métricas responde a uma pergunta distinta, e nenhuma das quatro substitui as demais. As demais métricas (MSE, RMSE, NRMSE, PRD, SNR, PSNR e PeakAmplitudeError) não são analisadas individualmente neste trabalho; capturam dimensões parcialmente sobrepostas às quatro selecionadas e sua inclusão não acrescentaria perspectivas novas ao conjunto de conclusões.

6.2 Séries de Alta Compressibilidade

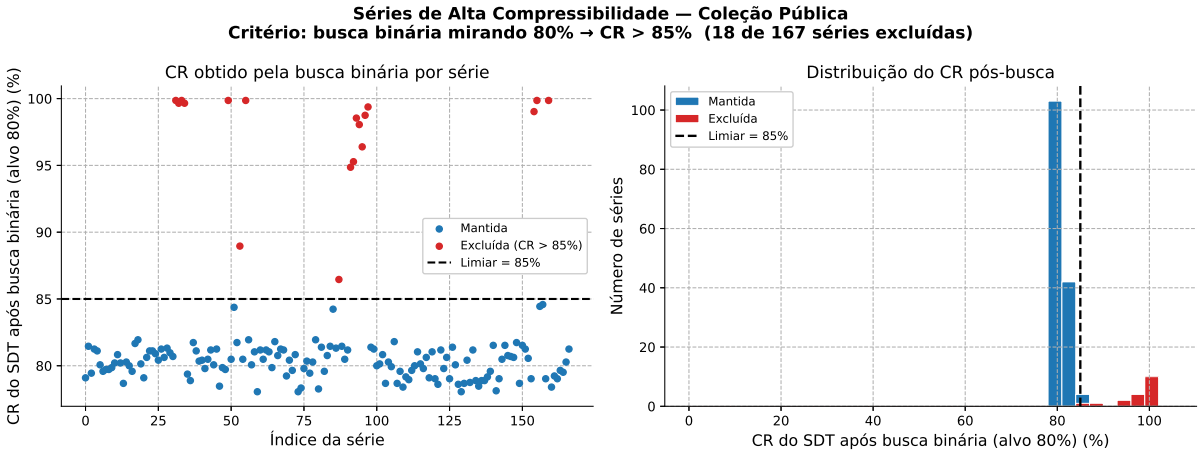
Conforme discutido na Seção 5.7.4, o SDT se beneficia estruturalmente em sinais de baixa variabilidade, comprimindo muito além do alvo com erro nulo. Para identificar essas séries, utiliza-se o critério definido na metodologia: aplica-se o SDT puro com busca binária de tolerância mirando CR = 80%, o mesmo mecanismo usado para o SDT e o RDP em toda a avaliação comparativa; se, mesmo com a busca ativamente tentando aproximar o CR do alvo, o resultado obtido ultrapassa 85% (uma margem de 5% acima do alvo), a série é classificada como de alta compressibilidade e excluída da análise comparativa.

O uso da busca binária como sonda de compressibilidade, em vez de uma tolerância fixa arbitrária, tem uma justificativa direta: é o mesmo mecanismo que efetivamente controla o SDT e o RDP ao longo de toda a avaliação, então usá-lo para identificar séries triviais garante que o critério meça exatamente a mesma noção de "dificuldade" que a busca binária enfrenta durante os experimentos. Se, mesmo tentando ativamente atingir 80%, a busca não consegue aproximar-se do alvo e ultrapassa-o por uma margem confortável, é porque a série é tão regular que não existe tolerância que produza um CR fino nas proximidades de 80% para ela: qualquer tolerância não nula já acomoda o sinal inteiro em poucos segmentos.

Na coleção pública, 18 das 167 séries (11%) apresentam alta compressibilidade natural sob esse critério, como mostra a Figura 6.1. Nessas séries, o SDT produz resultados excepcionais: a baixa variabilidade permite que qualquer tolerância não nula acomode o dia inteiro em pouquíssimos segmentos, com CR muito acima do alvo e erro de reconstrução nulo, exatamente o comportamento esperado de um algoritmo que preserva tendências lineares com garantia de erro. Na coleção privada, nenhuma série ultrapassa o limiar (0 das 89): mesmo as séries mais regulares da coleção privada ainda exercitam o corredor do SDT o suficiente para que a busca binária encontre uma tolerância que aproxime o CR do alvo de 80% com precisão fina, ao contrário do que se observa numa fração relevante da coleção pública. Essa diferença entre coleções é, em si, informativa: sugere que os perfis de consumo residenciais

e acadêmicos predominantes na coleção pública incluem mais séries excepcionalmente regulares (por exemplo, longos períodos de equipamento desligado) do que as instalações industriais e comerciais da coleção privada.

Figura 6.1 — Identificação de séries de alta compressibilidade na Coleção Pública. Esquerda: CR obtido pelo SDT após busca binária mirando 80% por série, destacando séries que ultrapassam 85% (linha tracejada). Direita: distribuição do CR pós-busca por série.



Fonte: elaborada pelo autor.

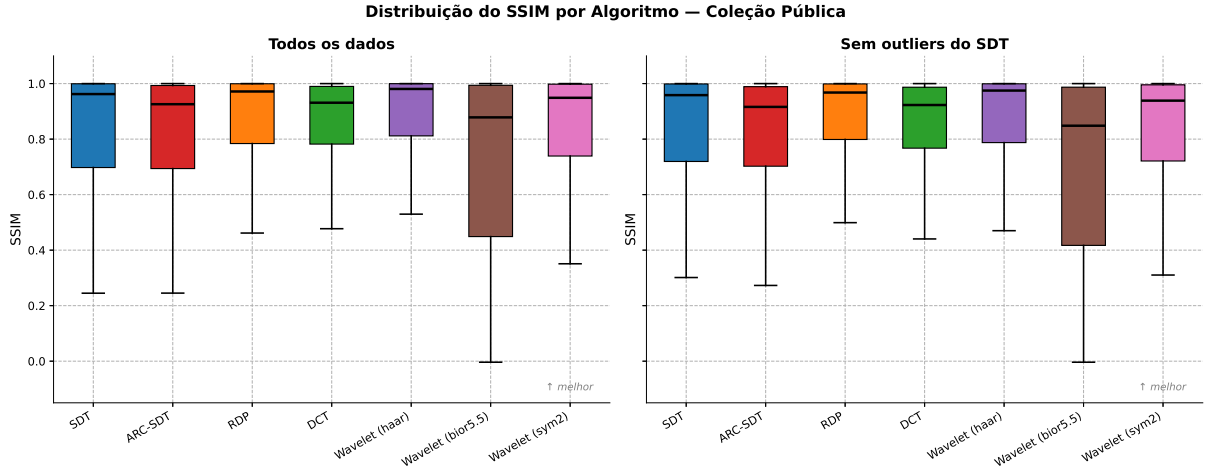
A análise comparativa da coleção pública é, por isso, apresentada em duas versões: com todas as séries, incluindo as de alta compressibilidade, e com essas séries removidas. A primeira mostra o desempenho real de cada algoritmo em toda a diversidade do conjunto; a segunda permite comparar os algoritmos em condições de variabilidade equivalentes, sem a vantagem de partida que favorece o SDT em sinais regulares. Para a coleção privada, onde nenhuma série é removida, as duas versões coincidem e apresenta-se apenas uma.

6.3 Qualidade de Compressão: Coleção Pública

6.3.1 Distribuição do SSIM por Algoritmo

A Figura 6.2 apresenta a distribuição do SSIM por algoritmo em todas as combinações de taxa-alvo e janela, com e sem as séries de alta compressibilidade.

Figura 6.2 — Distribuição do SSIM por algoritmo na Coleção Pública. Esquerda: todas as séries. Direita: excluindo séries de alta compressibilidade.



Fonte: elaborada pelo autor.

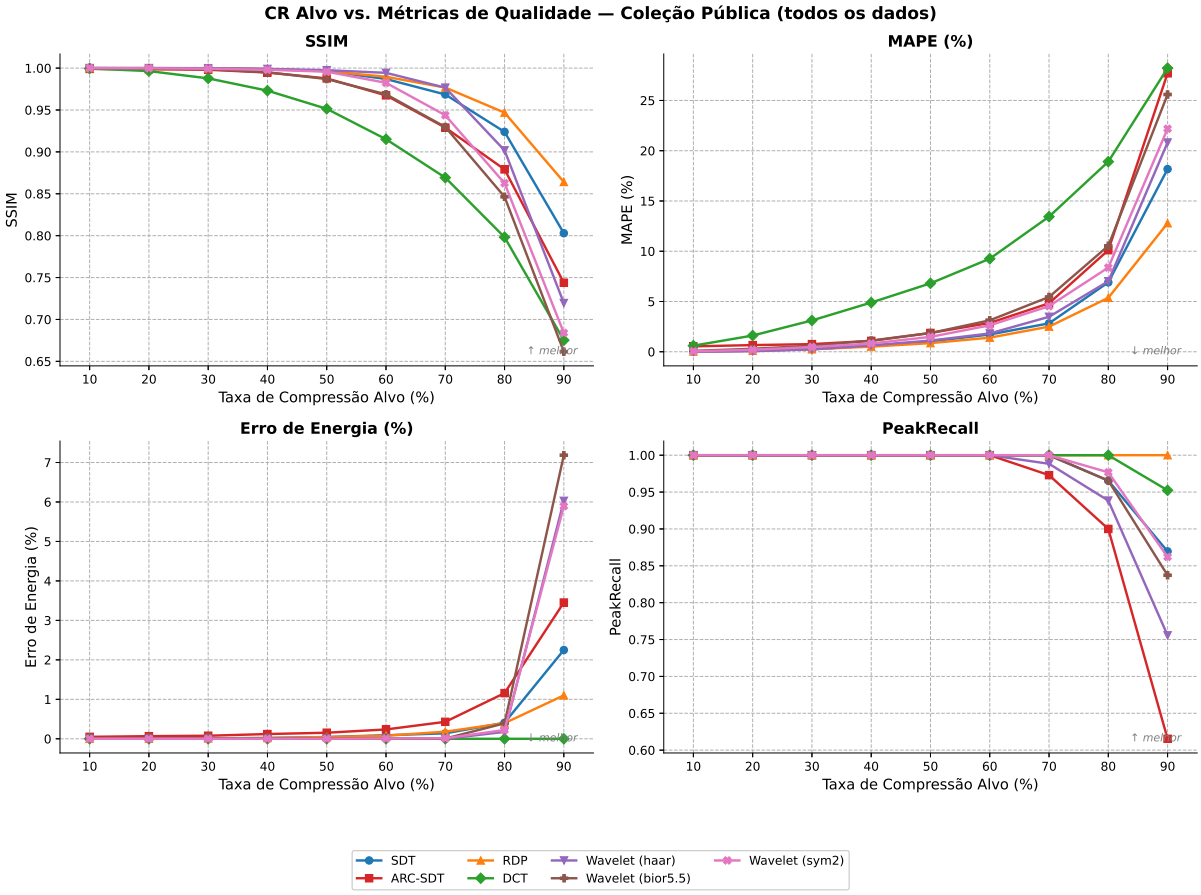
Com todas as séries, o SDT apresenta mediana de SSIM elevada (0,962), reflexo do desempenho excepcional nas séries de alta compressibilidade, onde a compressão além do alvo vem acompanhada de qualidade perfeita. Excluindo essas séries, a mediana cai para 0,958, aproximando-se do comportamento do algoritmo em condições de variabilidade típica. O RDP mantém uma mediana elevada e estável nas duas versões (0,972 completo, 0,968 filtrado), com dispersão pequena.

O resultado mais notável, no entanto, vem das Wavelets. Ao contrário do que se poderia supor de uma abordagem espectral concorrente da DCT, a wavelet `haar` alcança a maior mediana de SSIM entre os sete algoritmos avaliados, tanto no conjunto completo (0,981) quanto no filtrado (0,975), à frente do RDP (0,972 e 0,968), ainda que por uma margem estreita (cerca de 0,01 de SSIM). A `sym2` também se posiciona de forma competitiva (0,939 no conjunto filtrado), acima da DCT (0,923). Apenas a `bior5.5`, das três famílias de wavelet, permanece atrás das demais abordagens (0,848 no conjunto filtrado). Esse resultado é consequência direta do critério de seleção de coeficientes por magnitude discutido no Capítulo 3: ao manter os coeficientes de maior energia em vez de truncar sequencialmente as frequências mais baixas, tanto a DWT quanto a DCT exploram a garantia de melhor aproximação em K termos, da qual as famílias ortogonais (`haar`, `sym2`) se beneficiam integralmente; a `bior5.5`, biortogonal, não tem essa garantia e por isso continua atrás das demais.

6.3.2 Curvas CR Alvo vs. Métricas de Qualidade

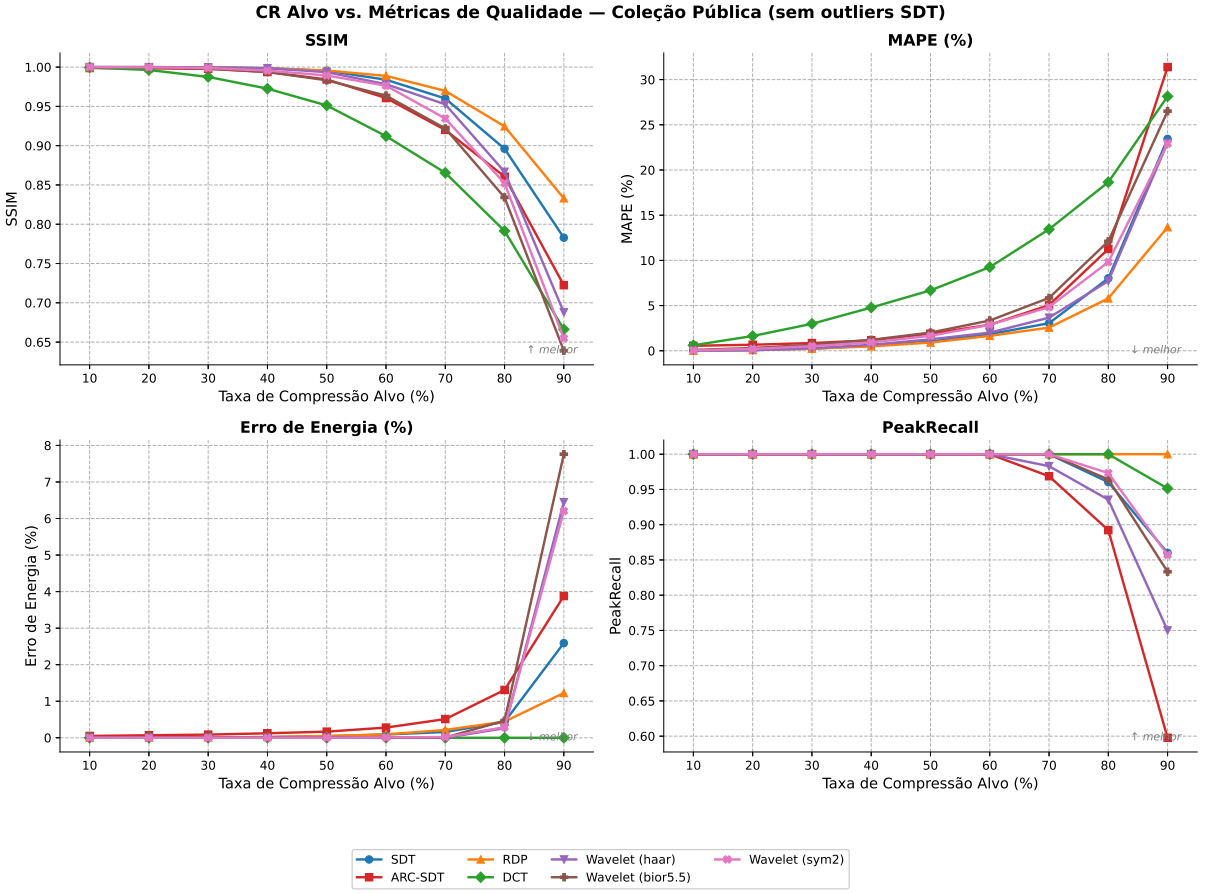
As Figuras 6.3 e 6.4 apresentam a mediana de cada métrica em função do CR alvo, com e sem as séries de alta compressibilidade, para a janela de 1440 pontos.

Figura 6.3 — Mediana das métricas de qualidade em função do CR alvo: Coleção Pública, janela de 1440 pontos (todos os dados).



Fonte: elaborada pelo autor.

Figura 6.4 — Mediana das métricas de qualidade em função do CR alvo: Coleção Pública, janela de 1440 pontos (excluindo séries de alta compressibilidade).



Fonte: elaborada pelo autor.

Quatro padrões se destacam no conjunto completo. Primeiro, o SDT apresenta SSIM e PeakRecall elevados nas taxas reais baixas, reflexo real do desempenho excepcional nas séries de alta compressibilidade, onde compressão além do alvo vem acompanhada de qualidade perfeita. Segundo, o RDP e a wavelet `haar` são praticamente indistinguíveis em SSIM até o CR alvo de 70%: nessa faixa inteira a diferença entre as duas medianas nunca passa de 0,005, e é a `haar`, e não o RDP, quem fica ligeiramente à frente na maior parte dela. É apenas nas duas taxas mais agressivas que o RDP abre vantagem de fato, e aí a diferença deixa de ser desprezível: 0,947 contra 0,902 em CR alvo de 80%, e 0,864 contra 0,720 em 90%. Em PeakRecall a separação segue o mesmo padrão e só se torna expressiva nessas mesmas taxas, onde o RDP é o único algoritmo que mantém PeakRecall de 1,000 mesmo em CR alvo de 90%, enquanto todos os demais (DCT e as três Wavelets inclusive) já apresentam alguma perda de picos nessa faixa. Terceiro, o EnergyError da DCT e das Wavelets ortogonais (`haar` e `sym2`) permanece próximo de zero na mediana em praticamente todas as taxas, mas essa proximidade não constitui uma garantia

estrutural sob o *hard thresholding* por magnitude (Capítulo 3): em CR alvo de 80% e 90%, uma fração pequena mas não desprezível das séries apresenta erro de energia severo, quando o coeficiente que codifica a média do sinal não está entre os de maior magnitude e acaba descartado. Quarto, ao contrário do que se poderia supor de métodos espectrais, as Wavelets ortogonais não têm desempenho uniformemente inferior às abordagens geométricas: a `haar` rivaliza com o RDP em SSIM e a `sym2` supera a DCT na maior parte das taxas; apenas a `bior5.5`, biortogonal, mantém-se atrás das demais em todas as métricas de forma.

Após a filtragem, a curva do SDT aproxima-se da do RDP sem, contudo, superá-la em nenhuma faixa de CR real. A vantagem do RDP em PeakRecall nas taxas mais agressivas mantém-se mesmo após a filtragem.

6.3.3 Tabela Resumo em CR Alvo de 80%

As tabelas a seguir servem como referência prática para seleção de algoritmo: ao configurar um CR alvo de 80%, os valores apresentados são as métricas de qualidade esperadas para cada algoritmo nessa configuração. Para DCT e DWT, o número de coeficientes é calculado algebricamente a partir do alvo e o CR real converge ao configurado dentro da margem de arredondamento. Para SDT, RDP e ARC-SDT, o usuário configura o alvo mas o CR real é um resultado que depende da estrutura do sinal, controlado indiretamente por busca binária no SDT e no RDP ou pelo controlador PI no ARC-SDT, podendo divergir do alvo em até 2%. As Tabelas 6.1 e 6.2 refletem esse valor esperado ao se configurar 80%.

Tabela 6.1 — Mediana das métricas de qualidade para CR alvo = 80% — Coleção Pública

Algoritmo	**SSIM**	**MAPE (%)**	**Erro de Energia (%)**	**PeakRecall**
SDT	0.6872	10.0562	1.4516	0.6931
ARC-SDT	0.6744	10.3712	2.5574	0.5498
RDP	**0.7842**	**7.0069**	1.2739	**0.8748**
DCT	0.7225	17.4568	**0.0000**	0.8036
Wavelet (haar)	0.6751	9.9782	0.6386	0.5804
Wavelet (bior5.5)	0.5323	17.8495	2.7798	0.6157
Wavelet (sym2)	0.6291	12.6195	0.8851	0.6579

Tabela 6.2 — Mediana das métricas de qualidade para CR alvo = 80% — Coleção Pública (sem outliers SDT)

Algoritmo	**SSIM**	**MAPE (%)**	**Erro de Energia (%)**	**PeakRecall**
SDT	0.6648	10.6843	1.7319	0.6990
ARC-SDT	0.6559	10.9522	2.8139	0.5515
RDP	**0.7660**	**7.7466**	1.4052	**0.8898**
DCT	0.7112	17.7446	**0.0000**	0.8100
Wavelet (haar)	0.6423	10.7900	0.7152	0.5776
Wavelet (bior5.5)	0.4695	19.5851	3.3258	0.6051
Wavelet (sym2)	0.6029	13.0308	0.9753	0.6533

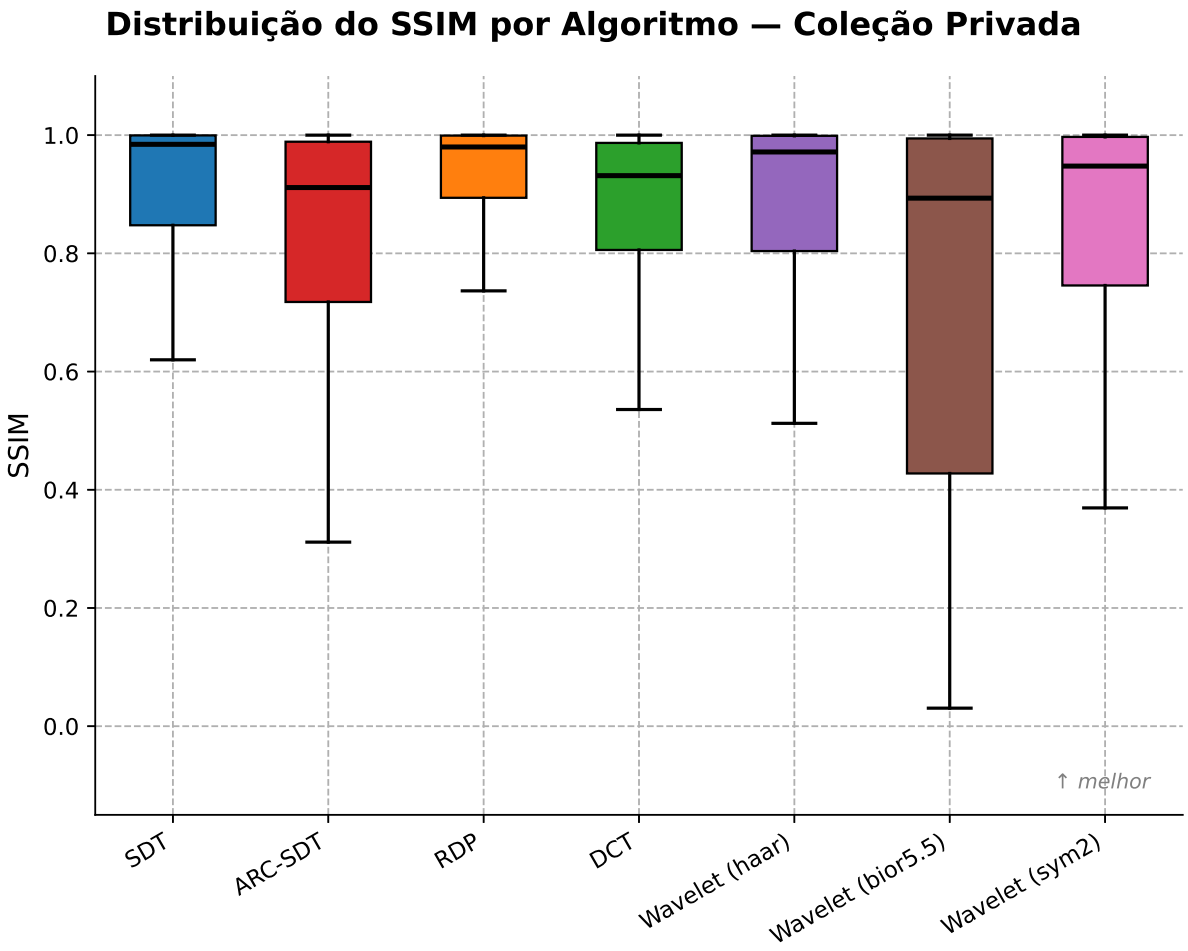
O RDP mantém, portanto, o melhor desempenho em SSIM e PeakRecall em ambas as versões da tabela. A DCT ainda alcança o menor erro de energia mediano do conjunto (essencialmente zero), mas essa proximidade não é uma garantia estrutural, como discutido na seção anterior: é o comportamento típico do *hard thresholding* por magnitude nessas séries, não uma propriedade garantida para toda execução. As Wavelets, nesta tabela que agrega os cinco tamanhos de janela avaliados (incluindo a janela de 15 pontos, onde o orçamento de bytes é mais apertado), ainda apresentam SSIM inferior ao RDP e à DCT; a `haar` se aproxima do SDT e do ARC-SDT (SSIM de 0,6423, na versão filtrada, contra 0,6648 e 0,6559, respectivamente), mas a `bior5.5` permanece distintamente atrás de todas as demais abordagens.

6.4 Qualidade de Compressão: Coleção Privada

6.4.1 Distribuição do SSIM por Algoritmo

A Figura 6.5 apresenta a distribuição do SSIM por algoritmo na coleção privada.

Figura 6.5 — Distribuição do SSIM por algoritmo na Coleção Privada.



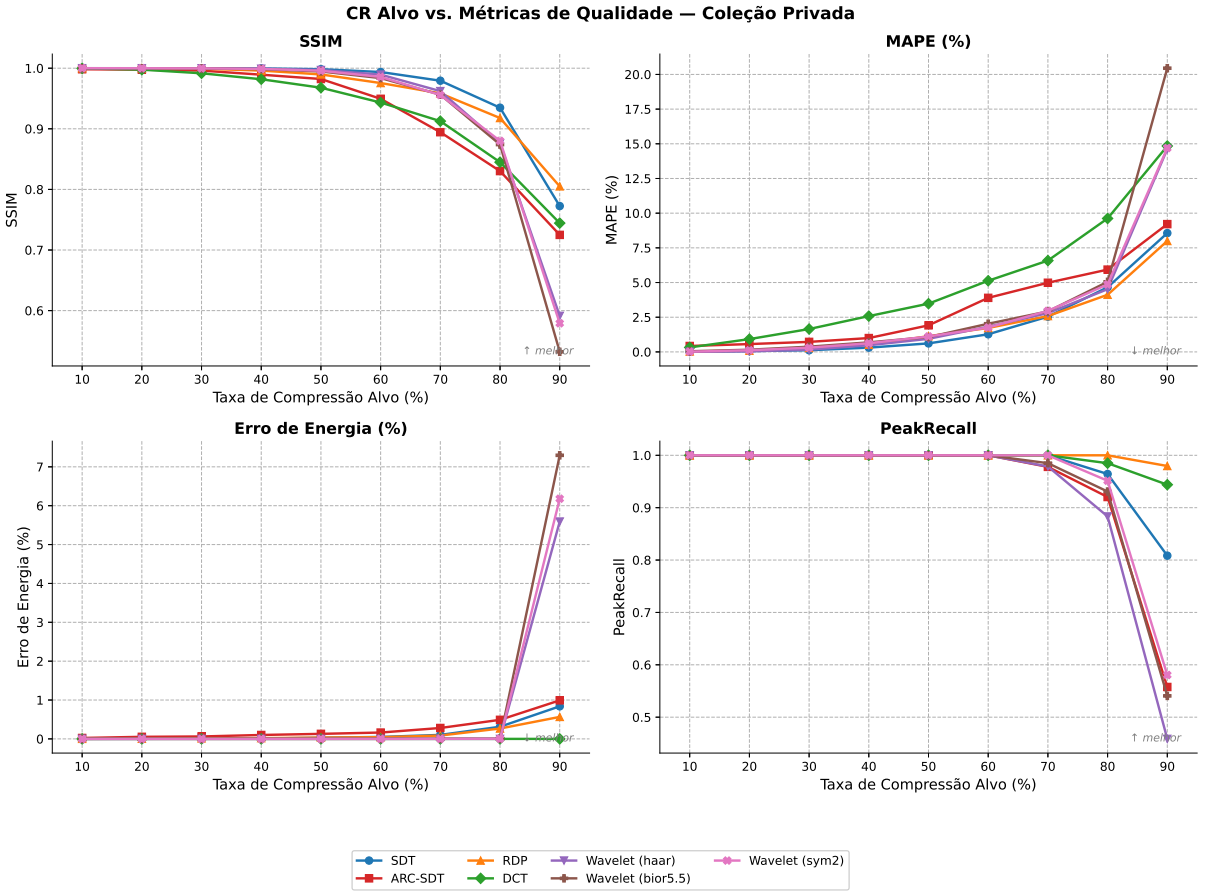
Fonte: elaborada pelo autor.

O padrão distribucional na coleção privada difere em um ponto notável do observado na pública: aqui é o SDT quem alcança a maior mediana de SSIM entre os sete algoritmos (0,984), à frente do RDP (0,980) por uma margem pequena, com a wavelet `haar` em terceiro lugar (0,972). Como na coleção pública, a `bior5.5` permanece a mais baixa do conjunto (0,893).

6.4.2 Curvas CR Alvo vs. Métricas de Qualidade

A Figura 6.6 apresenta as curvas de qualidade para a coleção privada.

Figura 6.6 — Mediana das métricas de qualidade em função do CR alvo: Coleção Privada, janela de 1440 pontos.



Fonte: elaborada pelo autor.

A ordenação dos algoritmos diverge, num ponto importante, da observada na coleção pública. Na janela de 1.440 pontos, o SDT e o RDP ficam praticamente empatados nas taxas conservadoras (10% a 40%), ambos com SSIM próximo de 1; a vantagem do SDT torna-se perceptível a partir das taxas intermediárias (50% a 80%), sendo ele superado pelo RDP apenas na taxa mais agressiva (90%). Essa tendência favorável ao SDT é bem mais ampla do que a observada na coleção pública, onde o RDP já superava o SDT a partir de 30%. A DCT mantém EnergyError mediano próximo de zero na maior parte das taxas, sob a mesma ressalva já discutida de que essa proximidade não é garantida para toda execução individual. As Wavelets ortogonais (`haar` e `sym2`) seguem competitivas com as abordagens geométricas até taxas moderadas, mas perdem PeakRecall de forma acentuada nas taxas mais agressivas: a `haar` cai de PeakRecall de 1,000 em CR alvo de 60% para 0,459 em 90%, queda mais acentuada do que a observada na coleção pública. Essas diferenças sugerem que os perfis de consumo industriais e comerciais da coleção privada, mais regulares que a média da coleção pública fora das séries de alta compressibilidade, favorecem o SDT de forma mais consistente ao longo de toda a faixa de taxas avaliada.

6.4.3 Tabela Resumo em CR Alvo de 80%

Na Tabela 6.3, que agrega o CR alvo de 80% sobre todos os tamanhos de janela, o padrão observado na tabela da coleção pública se repete: o RDP mantém o melhor desempenho em SSIM (0,7974) e PeakRecall (0,8821), e a DCT alcança o menor erro de energia mediano (essencialmente zero), sob a mesma ressalva de que essa proximidade não é garantida para toda execução. A wavelet `haar` (SSIM de 0,6076) fica, nesta tabela, atrás do SDT (0,6591) e próxima do ARC-SDT (0,6180) e da `sym2` (0,6164); a `bior5.5` permanece a mais baixa (0,5518). A consistência dessa ordenação agregada entre as duas coleções reforça que a vantagem do RDP em compressões agressivas não vem das características específicas dos dados privados, mas a comparação isolada por janela, discutida acima, mostra que a coleção privada tem uma particularidade real: na janela de 1.440 pontos, é o SDT quem tende a liderar na maior parte das taxas-alvo, diferença que a tabela agregada por si só não captura.

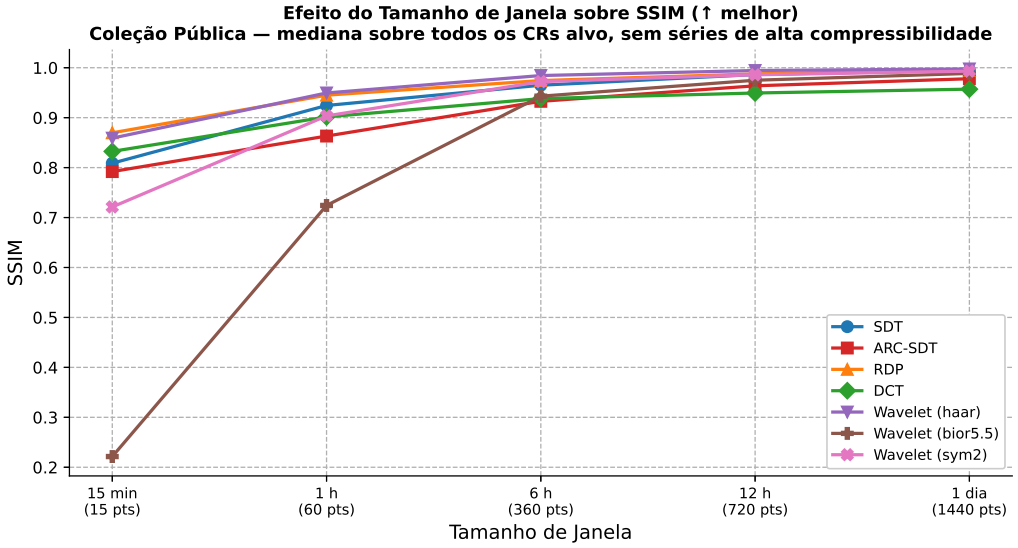
Tabela 6.3 — Mediana das métricas de qualidade para CR alvo = 80% — Coleção Privada

Algoritmo	**SSIM**	**MAPE (%)**	**Erro de Energia (%)**	**PeakRecall**
SDT	0.6591	5.5587	0.8017	0.6627
ARC-SDT	0.6180	6.2478	0.9915	0.5222
RDP	**0.7974**	**4.8813**	0.7724	**0.8821**
DCT	0.7546	9.1757	**0.0000**	0.7267
Wavelet (haar)	0.6076	6.2484	0.0427	0.4949
Wavelet (bior5.5)	0.5518	9.2570	0.1332	0.5230
Wavelet (sym2)	0.6164	6.9823	0.0893	0.5752

6.5 Efeito do Tamanho de Janela

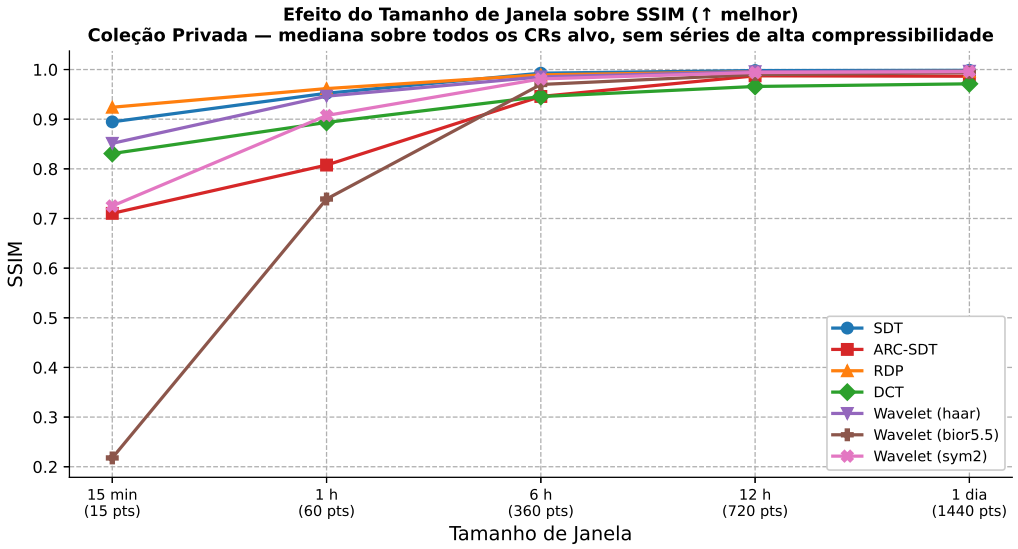
As Figuras 6.7 a 6.14 apresentam, para cada métrica, a mediana sobre todos os CRs alvo em função do tamanho de janela, com uma linha por algoritmo, nas duas coleções. O detalhamento por algoritmo com as curvas completas em função do CR alvo encontra-se no Apêndice B.

Figura 6.7 — Efeito do tamanho de janela sobre o SSIM: Coleção Pública. Cada linha é um algoritmo; o eixo y é a mediana sobre todos os CRs alvo, excluindo as séries de alta compressibilidade.



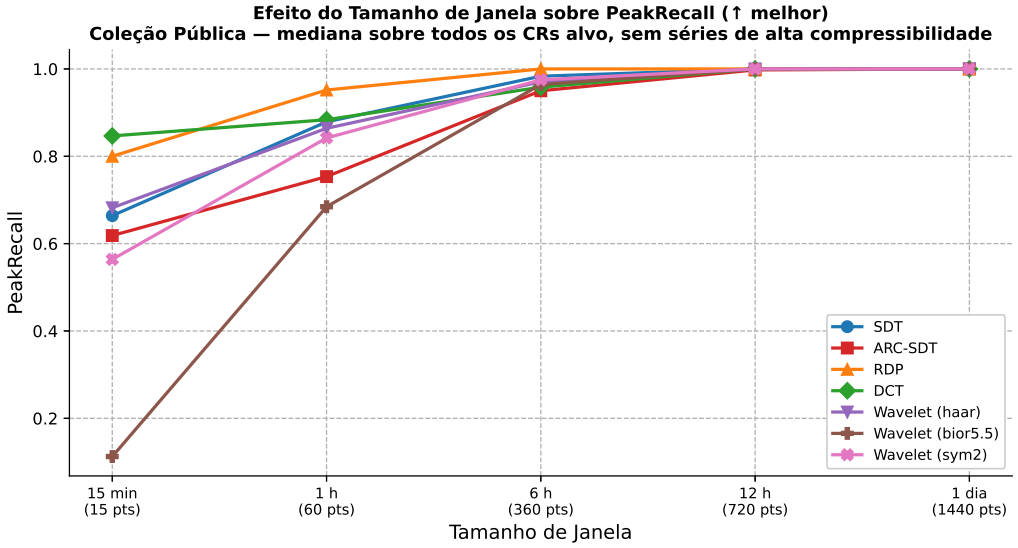
Fonte: elaborada pelo autor.

Figura 6.8 — Efeito do tamanho de janela sobre o SSIM: Coleção Privada.



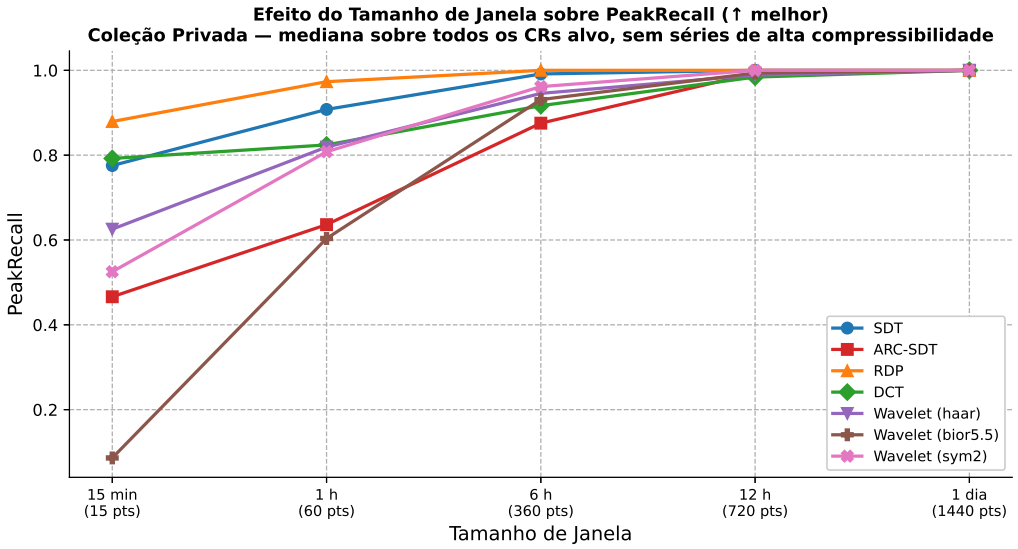
Fonte: elaborada pelo autor.

Figura 6.9 — Efeito do tamanho de janela sobre o PeakRecall: Coleção Pública (excluindo as séries de alta compressibilidade).



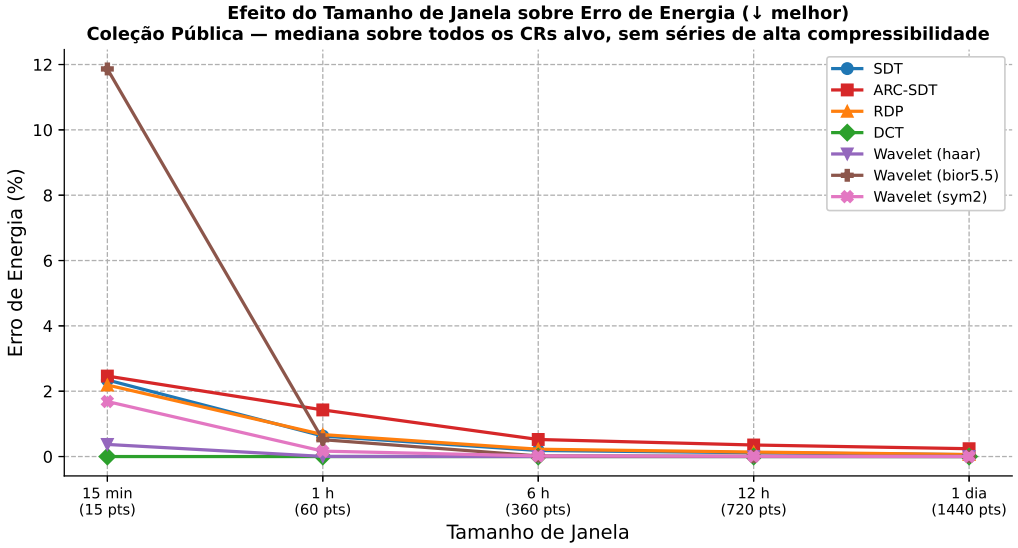
Fonte: elaborada pelo autor.

Figura 6.10 — Efeito do tamanho de janela sobre o PeakRecall: Coleção Privada.



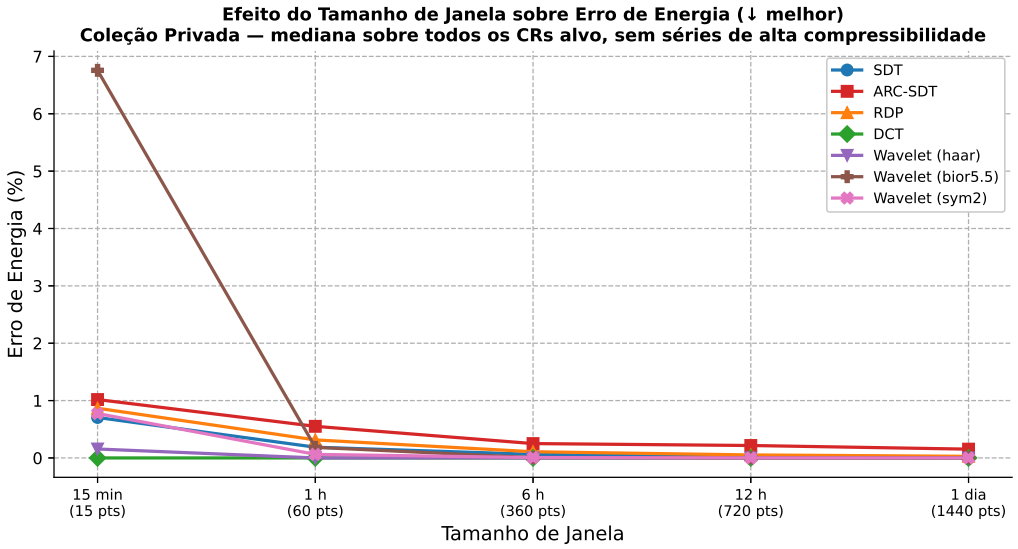
Fonte: elaborada pelo autor.

Figura 6.11 — Efeito do tamanho de janela sobre o Erro de Energia: Coleção Pública (excluindo as séries de alta compressibilidade).



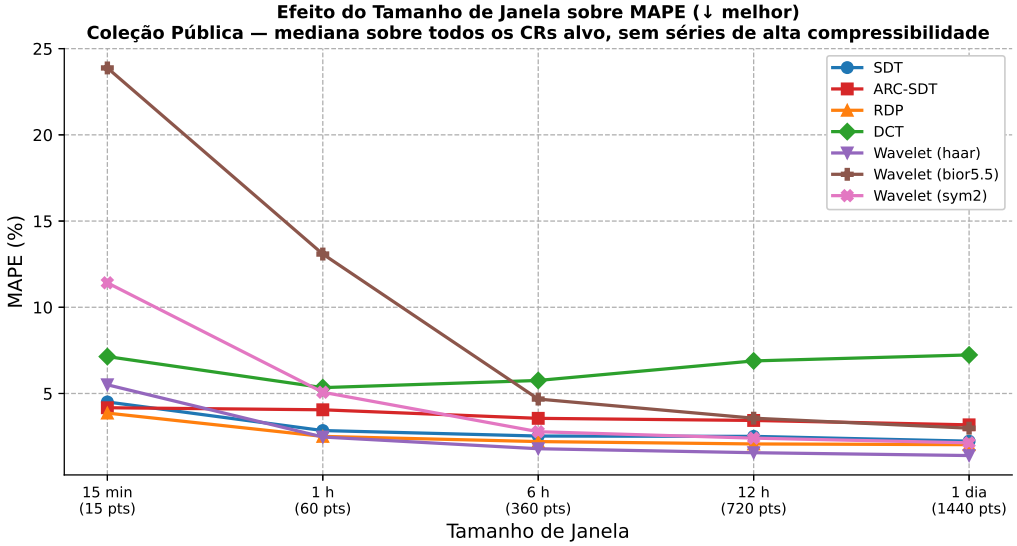
Fonte: elaborada pelo autor.

Figura 6.12 — Efeito do tamanho de janela sobre o Erro de Energia: Coleção Privada.



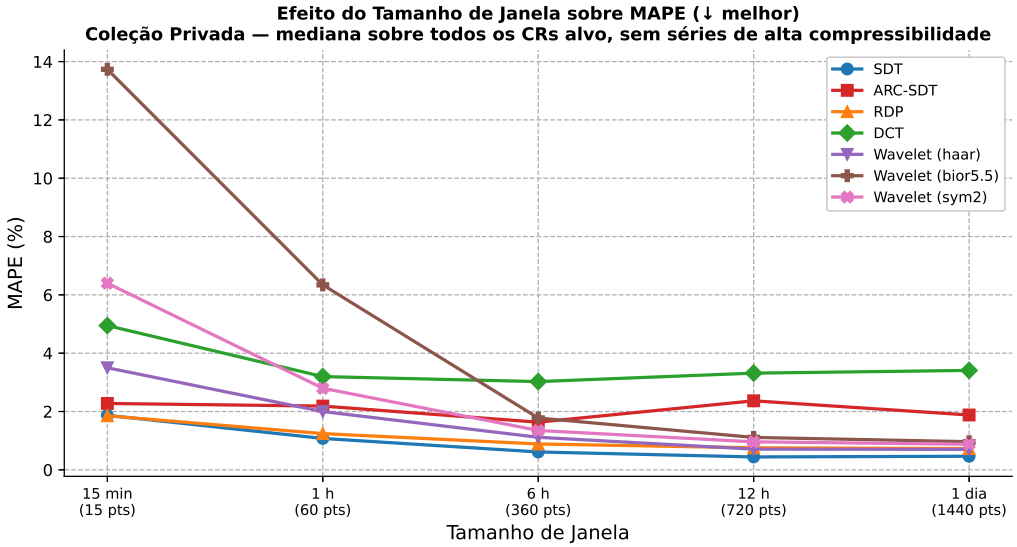
Fonte: elaborada pelo autor.

Figura 6.13 — Efeito do tamanho de janela sobre o MAPE: Coleção Pública (excluindo as séries de alta compressibilidade).



Fonte: elaborada pelo autor.

Figura 6.14 — Efeito do tamanho de janela sobre o MAPE: Coleção Privada.



Fonte: elaborada pelo autor.

O tamanho da janela tem impacto significativo sobre os algoritmos geométricos, e o padrão é consistente entre as duas coleções e entre as quatro métricas. Para o SDT, o SSIM mediano cai de 0,995 na janela de 1.440 pontos para 0,809 na janela de 15 pontos na coleção pública, evidenciando que o algoritmo depende de contexto temporal amplo para tomar boas decisões de descarte; o mesmo padrão

se observa no PeakRecall (de 1,000 para 0,664), no Erro de Energia (de 0,06% para 2,34%) e no MAPE (de 2,24% para 4,51%), que se degradam progressivamente à medida que a janela diminui. O RDP parte de uma base equivalente à do SDT (SSIM de 0,995 na janela de 1.440 pontos) mas se degrada menos: cai para 0,870 na janela de 15 pontos, contra 0,809 do SDT, e o mesmo padrão de vantagem relativa se repete no PeakRecall (0,800 contra 0,664) e no MAPE (3,86% contra 4,51%).

O comportamento da DCT e das Wavelets na janela de 15 pontos é bem mais favorável do que se poderia esperar de um orçamento de bytes tão apertado. Como discutido no Capítulo 3, o overhead de metadados desses algoritmos é fixo e não depende do tamanho da janela nem do nível de decomposição, o que evita qualquer colapso da taxa de compressão real: mesmo na janela de 15 pontos, o CR real acompanha o alvo configurado até 70%, saturando em torno de 83% para alvos de 80% e 90% (consequência do piso mínimo de 1 coeficiente mantido, discutido na Seção 3.4.5). O CR alvo permanece, portanto, um parâmetro efetivo mesmo nas janelas mais curtas.

Isso não significa, porém, que as quatro variantes espectrais se comportem de forma equivalente nessa janela. A DCT e as wavelets ortogonais (`haar` e `sym2`) mantêm qualidade de reconstrução razoável mesmo em 15 pontos: a DCT tem SSIM de 0,832, a `haar` de 0,859 (superior, inclusive, ao SDT nessa mesma janela) e a `sym2` de 0,721. A `bior5.5`, em contraste, colapsa: SSIM de apenas 0,221, EnergyError de 11,9% e PeakRecall de 0,112, ordens de magnitude piores que as demais famílias. A explicação está na estrutura do filtro: como discutido no Capítulo 3, o suporte da `bior5.5` é muito mais longo (12 amostras, contra 2 da `haar` e 4 da `sym2`), o que produz, por efeito de borda da decomposição, um vetor de coeficientes bem maior que a própria janela: cerca de 47 coeficientes para 15 pontos de entrada, contra 23 da `sym2` e apenas 16 da `haar`. Manter um único coeficiente entre 47 descarta uma fração muito maior da representação transformada do que manter 1 entre 16, o que penaliza desproporcionalmente a família de suporte mais longo justamente na configuração mais restrita de orçamento de bytes. Esse padrão se dissolve rapidamente com o crescimento da janela: já em 1.440 pontos, DCT e as três famílias de Wavelet atingem SSIM acima de 0,95, aproximando-se ou superando os algoritmos geométricos, como discutido na seção anterior. A restrição prática de janelas muito pequenas aplica-se, portanto, especificamente à `bior5.5`, e não às Wavelets como um todo.

Por fim, é importante contextualizar os resultados do SDT e do ARC-SDT nesta análise. Por processarem amostra a amostra, eles não dependem de uma janela fixa para operar: processam cada valor à medida que chega, sem armazenar o sinal completo, e podem funcionar indefinidamente com custo de memória constante. A avaliação com diferentes tamanhos de janela é útil para entender seu comportamento em contextos variados, mas a escolha da janela não tem nenhum embasamento forte para o ARC-SDT em particular: o custo computacional por amostra é exatamente o mesmo independentemente

do tamanho da janela, já que o algoritmo processa cada ponto com o mesmo conjunto de operações aritméticas e sem qualquer estrutura que dependa do horizonte temporal configurado. Janelas maiores são, pelo contrário, ativamente benéficas para o ARC-SDT: o RMS iterativo acumulado em `ms` torna-se progressivamente mais representativo da amplitude real do sinal à medida que mais amostras são incorporadas, melhorando a calibração da tolerância pelo controlador PI sem nenhum custo adicional de processamento ou memória. Diante disso, não há motivo para restringir o ARC-SDT a janelas pequenas. É importante destacar, ainda, que janelas grandes não implicam latência de monitoramento: cada vez que o SDT decide que um ponto não cabe no corredor atual e emite um *keypoint*, esse ponto pode ser transmitido imediatamente, comunicando ao receptor a tendência que acabou de fechar e marcando o início do próximo segmento. O monitoramento em tempo real é, portanto, compatível com janelas arbitrariamente grandes no SDT e no ARC-SDT.

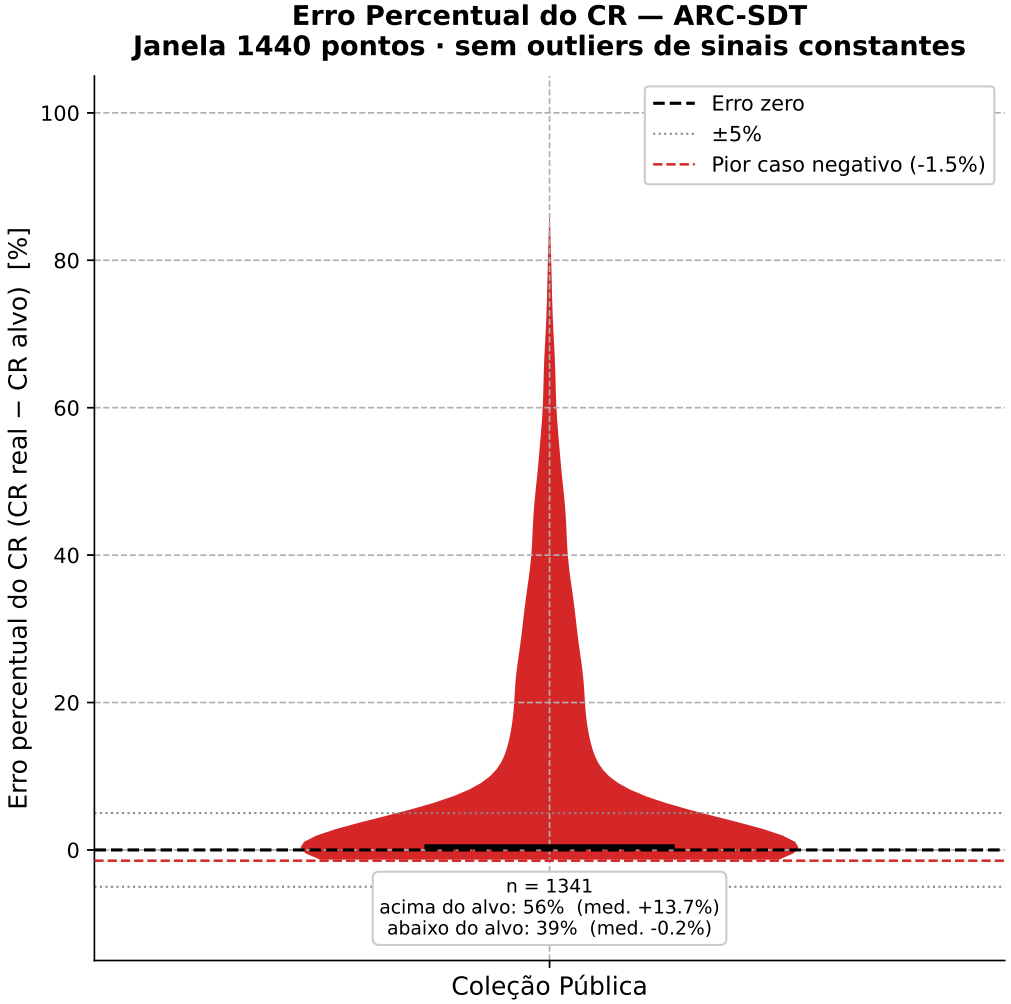
6.6 Avaliação do ARC-SDT

O ARC-SDT participa da comparação principal com os demais algoritmos, e suas curvas aparecem nas figuras desta seção. Esta seção complementa essa visão com a análise específica das características que distinguem o ARC-SDT do SDT base: a fidelidade do controle de taxa de compressão e o custo dessa controlabilidade em termos de qualidade.

6.6.1 Fidelidade de Controle da Taxa de Compressão

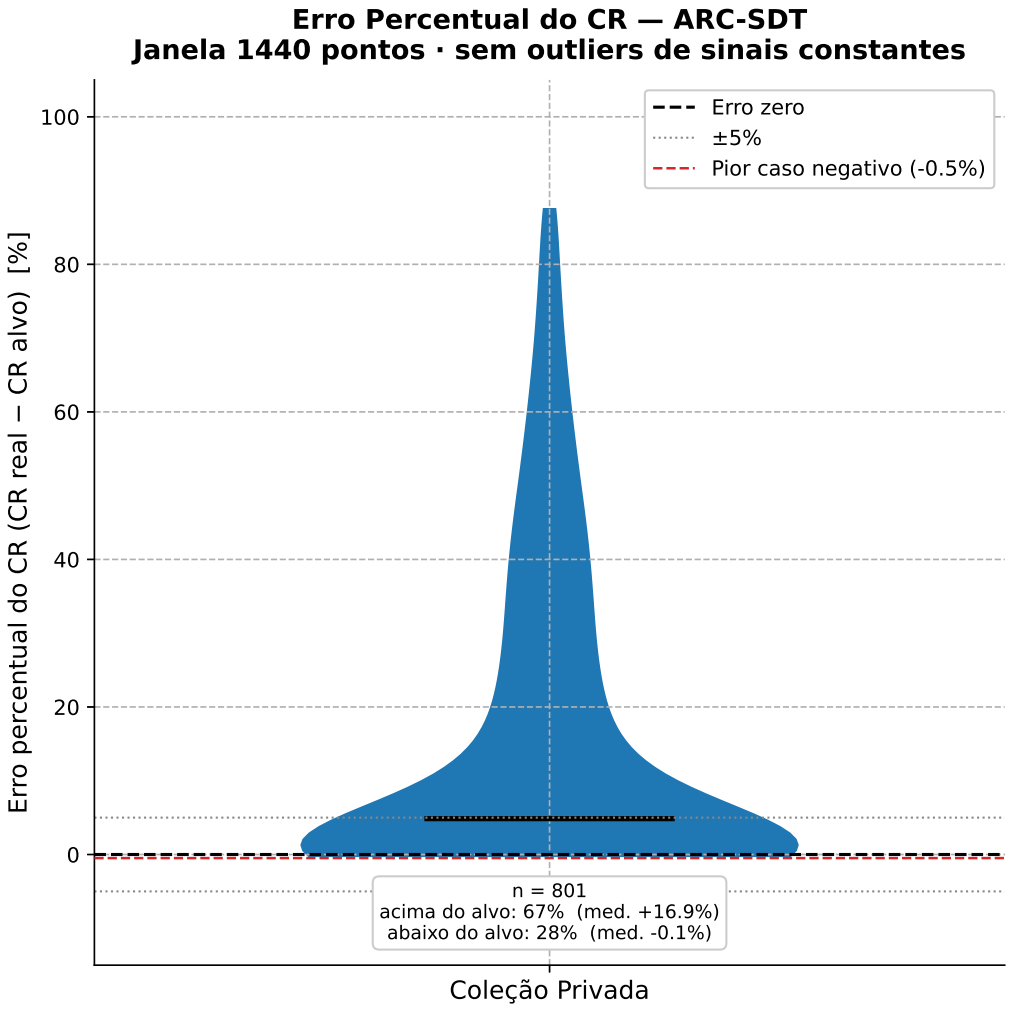
A principal afirmação do ARC-SDT é que o controlador PI ajusta o parâmetro de erro dinamicamente para que a taxa de compressão real se aproxime da taxa alvo. As Figuras 6.15 e 6.16 comparam a dispersão entre CR alvo e CR real para o ARC-SDT nas duas coleções.

Figura 6.15 — Distribuição do erro de controle do ARC-SDT (CR real − CR alvo): Coleção Pública, janela de 1440 pontos, excluindo séries de alta compressibilidade. O traço horizontal sobre cada violino marca a mediana. A linha tracejada indica erro zero; as linhas pontilhadas cobrem ±5%.



Fonte: elaborada pelo autor.

Figura 6.16 — Distribuição do erro de controle do ARC-SDT (CR real − CR alvo): Coleção Privada, janela de 1440 pontos.



Fonte: elaborada pelo autor.

Nas séries de alta compressibilidade, o ARC-SDT exibe comportamento análogo ao SDT: a baixa variabilidade faz com que o corredor, mesmo com `tol_rel` no limite mínimo, acomode longos trechos em poucos segmentos, e a taxa de compressão converge para valores próximos de 100% independentemente do parâmetro ajustado. O controlador detecta o desvio positivo e reduz `tol_rel` até o piso de 1%, mas a compressão permanece alta pelas características do próprio sinal. Esse é o mesmo comportamento vantajoso do SDT nesses perfis: ambos comprimem excepcionalmente bem, e o ARC-SDT não é exceção.

As Figuras 6.15 e 6.16 apresentam a distribuição do erro de controle excluindo essas séries, pois nelas o parâmetro `tol_rel` não determina o CR resultante (a estrutura do sinal determina) e incluí-las distorceria a avaliação da capacidade de controle em sinais de variabilidade típica. A análise considera

apenas a janela de 1440 pontos, o caso de uso natural do algoritmo.

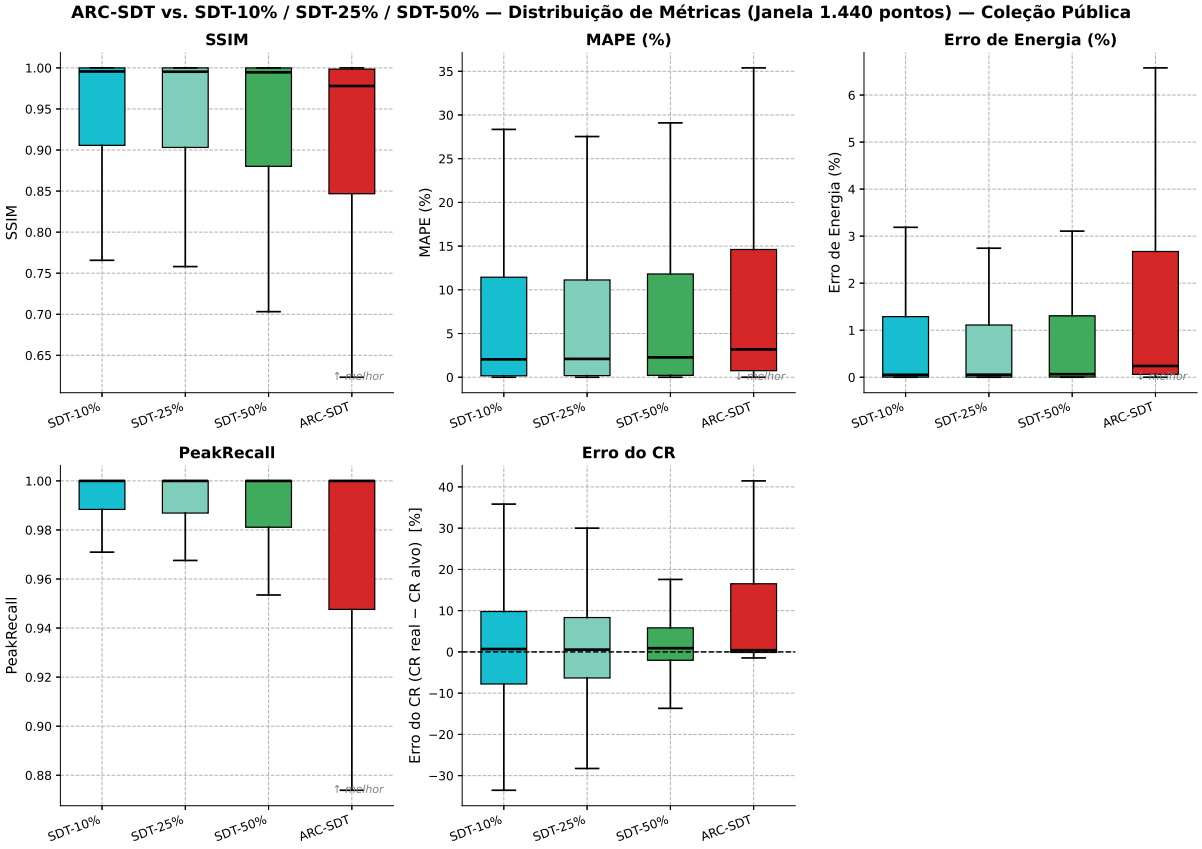
É importante destacar que os dois sentidos do erro têm consequências bem diferentes. Um valor positivo (CR real $>$ CR alvo) significa que o algoritmo comprimiu mais do que o necessário: o pacote transmitido é menor do que o limite disponível, situação aceitável. Um valor negativo (CR real $<$ CR alvo) significa que o algoritmo comprimiu menos do que o necessário: o pacote excede a carga útil disponível na rede, situação crítica.

Na coleção pública, 56% das execuções ficaram acima do alvo, 39% abaixo e os 5% restantes atingiram o alvo exatamente (mediana $+0{,}42\%$). Na coleção privada, 67% ficaram acima, 28% abaixo e os 4% restantes atingiram o alvo exatamente (mediana $+4{,}86\%$), uma mediana bem mais alta do que na pública. Esse deslocamento é consistente com a regularidade dos perfis industriais e comerciais desta coleção: em séries cujo consumo varia pouco ao longo do dia, o corredor acomoda trechos longos mesmo com `tol_rel` rebaixado até o piso de 1%, de modo que a compressão ultrapassa o alvo por característica do próprio sinal, e não por limitação do controlador. A mediana dos desvios negativos é de $-0{,}21\%$ na coleção pública e $-0{,}14\%$ na privada, e o pior caso registrado não passa de $-1{,}46\%$. Isso significa que, mesmo quando o ARC-SDT não atinge o alvo, o pacote gerado fica marginalmente acima do tamanho desejado, erro que pode ser absorvido por uma pequena margem de segurança no dimensionamento da carga útil.

6.6.2 Qualidade de Reconstrução: ARC-SDT vs. SDT

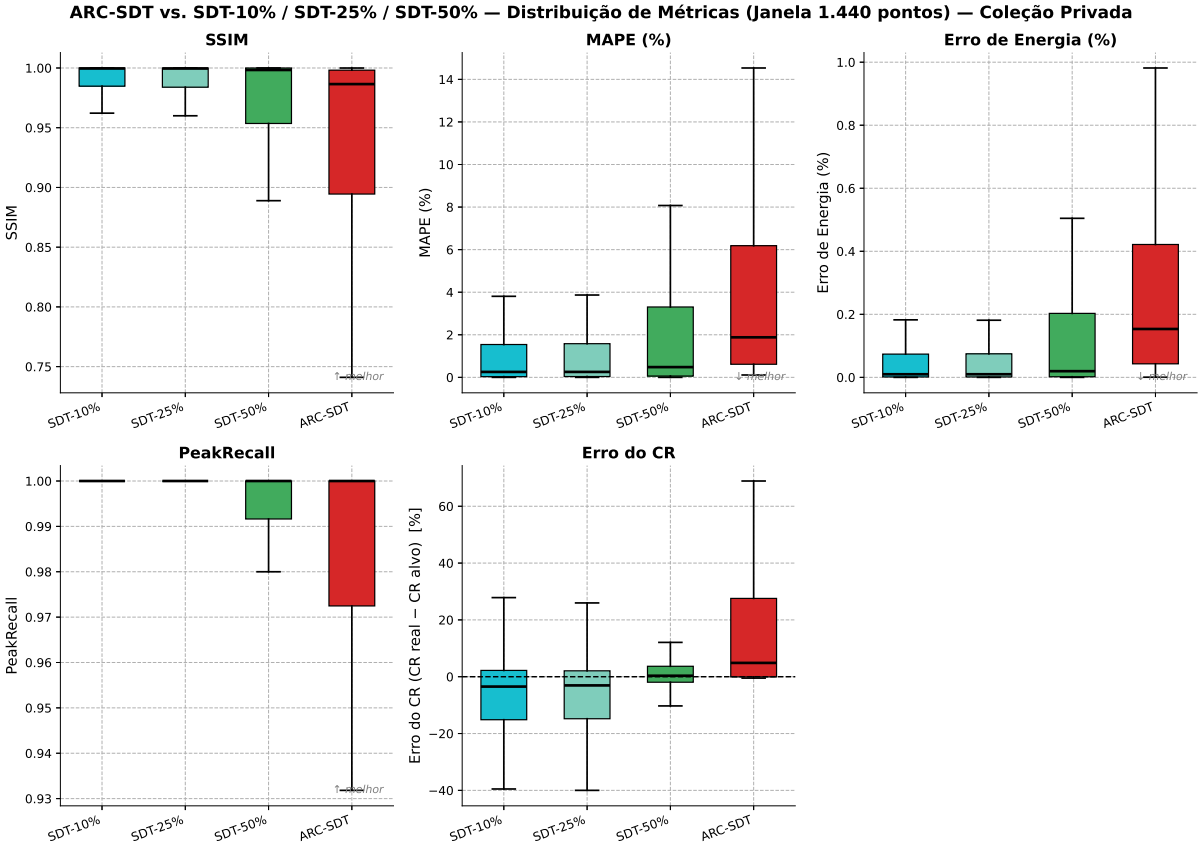
O controle preciso da taxa de compressão (propriedade ausente no SDT original) é obtido pelo ARC-SDT ao custo de ajustes dinâmicos na tolerância do corredor. Mas qual é o preço pago em qualidade de reconstrução? Responder essa pergunta exige calibrar o SDT para o mesmo CR alvo, e aqui surge uma dificuldade metodológica: calibrar por busca binária vendo a janela inteira é um processo realista, mas anula a vantagem que o SDT tem sobre os algoritmos em lote, o processamento amostra a amostra, já que a calibração exige ter a janela inteira disponível antes de começar a comprimir. Para manter essa comparação justa, o SDT foi calibrado por busca binária usando só uma fração inicial da janela (10%, 25% e 50% dos 1.440 pontos) e a tolerância resultante foi aplicada à janela inteira, simulando a mesma restrição que o ARC-SDT sempre enfrenta: não conhecer o futuro da série no momento de decidir a compressão.

Figura 6.17 — Distribuição de métricas de qualidade e do erro do CR (CR real − CR alvo) para ARC-SDT e SDT calibrado com frações iniciais da janela (10%, 25%, 50%): Coleção Pública, janela de 1.440 pontos, excluindo séries de alta compressibilidade.



Fonte: elaborada pelo autor.

Figura 6.18 — Distribuição de métricas de qualidade e do erro do CR (CR real − CR alvo) para ARC-SDT e SDT calibrado com frações iniciais da janela (10%, 25%, 50%): Coleção Privada, janela de 1.440 pontos.



Fonte: elaborada pelo autor.

Na coleção privada (instalações residenciais, comerciais e industriais, mais heterogêneas entre si), calibrar com frações pequenas da janela (10% ou 25% dos dados) faz o SDT errar o CR para menos, chegando a -40% no pior caso. Esse desvio se conecta a um achado já discutido neste capítulo: quanto maior o CR alvo, maior a degradação das métricas de qualidade. Sendo assim, se o SDT calibrado numa janela pequena está de fato comprimindo menos do que o alvo pedido, é esperado que suas métricas de qualidade apareçam melhores nesse comparativo, não porque o algoritmo seja superior, mas porque ele está comprimindo menos do que deveria. Aumentar a janela de calibração para 50% resolve em parte esse problema: o SDT passa a acertar bem o CR alvo, mas em compensação as métricas de qualidade degradam claramente. O ARC-SDT controla o erro do CR de forma bem mais confiável que qualquer uma dessas variantes do SDT, à custa de uma degradação parecida nas métricas, mas com uma vantagem que nenhuma variante testada do SDT tem: continua processando amostra a amostra, sem precisar de uma fase de calibração em lote antes de começar a operar.

Na coleção pública (majoritariamente residencial), esse efeito fica bem mais discreto: as métricas de

qualidade ficam praticamente equivalentes entre as quatro abordagens. Ainda assim, é possível observar a mesma tendência de fundo: aumentar a janela de calibração do SDT tende a piorar levemente as métricas, ainda perdendo a vantagem do processamento amostra a amostra que o ARC-SDT tem.

Diante desse panorama, o SDT com calibração em lote só atinge o CR alvo de forma confiável usando uma fração grande dos dados para calibrar (50%), perdendo justamente a vantagem que motivava seu uso em primeiro lugar, o processamento amostra a amostra. Além disso, do mesmo modo que a calibração numa fração do dia não garante o resultado para o dia inteiro, essa mesma calibração pode não se sustentar nos dias, meses ou anos seguintes, à medida que o perfil de consumo muda. O ARC-SDT resolve as duas questões ao mesmo tempo: atinge o CR alvo de forma consistente e mantém a operação amostra a amostra, pagando por isso uma queda moderada e mensurável nas métricas de qualidade.

6.7 Síntese dos Resultados

A consistência dos resultados entre as duas coleções (uma pública, de origem acadêmica, e outra privada, de instalações brasileiras) é parcial. Os padrões amplos se repetem: os algoritmos geométricos (SDT e RDP, ambos com calibração por busca binária) e as wavelets ortogonais (`haar` e `sym2`) permanecem, nas duas coleções, entre os de melhor fidelidade de forma, e a `bior5.5` permanece a mais fraca em ambas. Mas a posição relativa entre SDT e RDP se inverte: na coleção pública, o RDP fica à frente do SDT em praticamente toda a faixa de taxas; na privada, é o SDT quem tende a liderar na maior parte da faixa avaliada, na janela de 1.440 pontos. Essa diferença, discutida na seção anterior, é atribuída aos perfis de consumo mais regulares das instalações industriais e comerciais que compõem a coleção privada.

No que diz respeito à fidelidade de forma, o quadro é de maior equilíbrio entre os algoritmos do que se costuma esperar de métodos espectrais frente a métodos geométricos. O RDP se destaca sobretudo na preservação de picos de demanda em taxas de compressão agressivas: é o único algoritmo que mantém PeakRecall próximo de 1,000 mesmo em CR alvo de 90%, robustez que é sua vantagem mais consistente entre os algoritmos avaliados e dimensão crítica para o monitoramento energético. Em SSIM, no entanto, a wavelet `haar` alcança a maior mediana entre os sete algoritmos quando se considera o conjunto completo de janelas e taxas-alvo, à frente do RDP por uma margem estreita (cerca de 0,01 de SSIM), resultado direto da adoção do critério de seleção de coeficientes por magnitude (Capítulo 3), que garante a melhor aproximação possível em K termos para as famílias ortogonais de wavelet.

A DCT mantém erro de energia próximo de zero na mediana da grande maioria das configurações avaliadas, propriedade que se origina do fato de o coeficiente que codifica a média do sinal costumar

figurar entre os de maior magnitude nessas séries de consumo. Essa proximidade, contudo, não é uma garantia estrutural sob o *hard thresholding* por magnitude: em taxas de compressão agressivas (CR alvo de 80% e 90%), uma fração das séries apresenta erro de energia severo, quando esse coeficiente não sobrevive à seleção. A mesma ressalva vale para as wavelets ortogonais, que compartilham o critério. Entre as wavelets, a `bior5.5` é a exceção notável: seu suporte de filtro mais longo (12 amostras, contra 2 da `haar` e 4 da `sym2`, Capítulo 3) a torna muito mais sensível a janelas pequenas, com colapso severo de qualidade na janela de 15 pontos que não se observa nas famílias `haar` e `sym2`. A restrição prática de janelas muito pequenas para dispositivos de borda aplica-se, portanto, especificamente à `bior5.5`, não às Wavelets como grupo.

Diante deste contexto, o SDT (avaliado em sinais com variabilidade real, excluídas as séries de alta compressibilidade, com ajuste por busca binária, perdendo a característica amostra a amostra) apresenta SSIM comparável ou superior ao RDP, a depender da coleção, mas com fidelidade de pico consistentemente menor: o RDP continua sendo a escolha mais robusta quando a preservação de picos de demanda é o critério prioritário, sobretudo em taxas de compressão agressivas. O ARC-SDT, por sua vez, oferece controle adaptativo da taxa de compressão sem calibração manual, ao custo de uma queda mensurável em SSIM e PeakRecall em relação ao SDT base, reflexo da natureza do ajuste dinâmico de tolerância. Em termos de recursos computacionais, o SDT original, sem qualquer etapa de calibração (isto é, operando com uma tolerância já conhecida de antemão), tem o mesmo custo de memória constante do ARC-SDT, por ambos operarem de forma incremental. O SDT avaliado neste capítulo, porém, não está nessa condição: a calibração por busca binária sobre a janela inteira exige armazenar o sinal completo antes de iniciar a compressão, custo que o ARC-SDT nunca paga. Essa operação incremental torna os dois candidatos naturais para dispositivos embarcados com restrições severas de memória e processamento, com destaque para o ARC-SDT por permitir controle da taxa de compressão sem exigir esse acesso privilegiado ao sinal.

Capítulo 7

Conclusão e Trabalhos Futuros

7.1 Conclusão

O monitoramento contínuo de grandezas elétricas produz volumes de dados que exercem pressão simultânea sobre múltiplos recursos escassos em dispositivos de campo. Um medidor que registra 30 grandezas com resolução de um minuto gera, em um único dia, aproximadamente 43.200 valores de ponto flutuante de 32 bits, o que equivale a cerca de 172 kB de dados brutos sem considerar cabeçalhos, *timestamps* ou identificadores de dispositivo. Para redes com restrições severas de banda, como o LoRaWAN em áreas rurais e industriais, esse volume é intransmissível em tempo real sem alguma forma de redução; para dispositivos com memória *flash* de poucos kilobytes, é igualmente inviável armazená-lo localmente por períodos prolongados; e para sistemas que processam e agregam dados diretamente na borda, transmitir ou persistir dados brutos representa um consumo desnecessário de energia e processamento. Diante desse conjunto de pressões, a compressão no próprio dispositivo deixa de ser uma otimização opcional e passa a ser uma condição de viabilidade em múltiplos contextos de implantação.

Neste trabalho, o caso concreto estudado é o mais restritivo desses ambientes: dispositivos de monitoramento instalados em áreas industriais com cobertura LoRaWAN, onde os limites de carga útil por pacote variam de 11 bytes (SF10, configuração padrão de tempo de permanência de 400 ms) a 222 bytes (SF7), podendo ser inferiores ao tamanho de uma única leitura bruta não comprimida [35].

Para responder a esse problema, realizamos uma avaliação comparativa de seis variantes de algoritmo de compressão com perda (SDT, RDP, DCT e três famílias de DWT: haar, bior5.5 e sym2), avaliados sobre 256 séries temporais de consumo elétrico de instalações públicas e privadas, em nove taxas de compressão alvo e cinco tamanhos de janela. As métricas selecionadas cobrem quatro dimensões distintas: fidelidade de forma do sinal (SSIM), preservação de picos de demanda (PeakRecall), erro percentual

relativo ponto a ponto (MAPE) e fidelidade energética (EnergyError).

Os resultados são consistentes entre as duas coleções em seus padrões amplos, com uma diferença notável discutida adiante. Em fidelidade de forma, o RDP só assume a liderança em SSIM nas taxas de compressão mais agressivas, ficando praticamente empatado com a wavelet `haar` em toda a faixa abaixo delas, e é o algoritmo mais robusto na preservação de picos de demanda nessa mesma faixa agressiva, mantendo PeakRecall próximo de 1,000 mesmo em CR alvo de 90%. A DCT mantém erro de energia próximo de zero na mediana da maioria das configurações, mas essa proximidade não é uma garantia estrutural sob o *hard thresholding* por magnitude: em taxas de compressão agressivas, uma fração das séries apresenta erro de energia severo quando o coeficiente que codifica a média do sinal não sobrevive à seleção (Capítulo 3). Ao contrário do que se poderia supor de uma abordagem espectral, as Wavelets ortogonais (`haar` e `sym2`) rivalizam com os algoritmos geométricos em fidelidade de forma: a `haar`, inclusive, alcança a maior mediana de SSIM entre os sete algoritmos avaliados quando se considera o conjunto completo de janelas e taxas-alvo. Apenas a `bior5.5`, cujo suporte de filtro mais longo a torna especialmente sensível a janelas pequenas, mantém desempenho consistentemente inferior às demais abordagens.

O SDT, por sua vez, reúne propriedades atraentes para uso embarcado (processamento amostra a amostra e custo de memória constante), mas apresenta uma limitação operacional para cenários com carga útil fixa: quando operado com tolerância fixa, não oferece controle direto sobre a taxa de compressão resultante, e seu comportamento em relação à preservação de picos se torna imprevisível em altas taxas de compressão. Esse comportamento decorre da natureza da busca binária em lote: a tolerância ótima para atingir um CR alvo é aplicada uniformemente ao sinal inteiro, e o resultado para os picos depende de características locais do sinal que variam entre séries.

Para resolver essa limitação, este trabalho propôs e avaliou o ARC-SDT, uma adaptação que expressa a tolerância como fração do RMS local do sinal e emprega um controlador PI para ajustar essa fração a cada nova amostra, buscando aproximar o CR resultante de um alvo configurável. Os experimentos mostram que o desvio do controlador (CR real menos CR alvo) tem mediana de $+0{,}42\%$ na coleção pública e $+4{,}86\%$ na privada; restrito aos casos de desvio negativo, a mediana é de $-0{,}21\%$ e $-0{,}14\%$, respectivamente, com pior caso de $-1{,}46\%$ entre todas as execuções avaliadas. O desvio negativo é o único com implicação prática: ele indica que o pacote gerado é ligeiramente maior do que o limite de carga útil alvo, situação absorvível por uma pequena margem de segurança no dimensionamento. O desvio positivo, que pode ser expressivo em sinais de alta compressibilidade natural, é aceitável: significa que o sinal admitiu compressão além do alvo, gerando um pacote ainda menor, o que não compromete os objetivos da transmissão.

O custo-benefício dessa controlabilidade é mensurável em todas as métricas de qualidade. Em CR de 80% e janela de 1.440 pontos (o caso de uso natural do ARC-SDT), o algoritmo apresenta SSIM mediano de $0{,}861$ contra $0{,}896$ do SDT, PeakRecall de $0{,}892$ contra $0{,}960$, e EnergyError de $1{,}31\%$ contra $0{,}43\%$. A queda mais pronunciada está no PeakRecall: como o controlador PI ajusta a tolerância com base no RMS acumulado do sinal (uma medida global de amplitude) e não na posição dos eventos de pico, ele tende a sacrificar picos para atingir a taxa alvo. Esse é o custo estrutural do controle adaptativo amostra a amostra: sem acesso ao sinal completo, o algoritmo não consegue distribuir os descartes de forma a preservar os pontos geometricamente mais relevantes. É importante ressaltar que o SDT na avaliação comparativa é executado com busca binária sobre a série completa, o que lhe confere acesso privilegiado ao sinal para calibrar a tolerância (condição que o ARC-SDT, por operar em tempo real sem armazenamento prévio do sinal, não dispõe). A diferença de qualidade entre os dois deve, portanto, ser interpretada como o custo da operação amostra a amostra em relação a uma calibração em lote com informação completa, e não como uma limitação do mecanismo de controle em si.

Quando todos os algoritmos são avaliados na mesma janela de 1.440 pontos e em CR de 80%, o ARC-SDT não lidera o conjunto: com SSIM mediano de $0{,}861$, fica atrás do RDP ($0{,}925$) e do próprio SDT base ($0{,}896$), e praticamente empatado com a wavelet `haar` ($0{,}867$). É importante destacar, contudo, que nessa mesma configuração o ARC-SDT supera a `sym2` ($0{,}852$), a `bior5.5` ($0{,}834$) e a DCT ($0{,}792$), de modo que a proposta se posiciona no meio do conjunto avaliado, e não atrás dele. Essa comparação, porém, ignora uma diferença estrutural fundamental entre as abordagens: a DCT, o RDP e as Wavelets são algoritmos em lote, que precisam armazenar a janela inteira em memória antes de iniciar a compressão, enquanto o SDT e o ARC-SDT processam cada amostra à medida que ela chega, com custo de memória constante e independente do tamanho da janela (Seção 4.5). Comparar os dois grupos na mesma janela só é justo sob duas suposições simultâneas, ambas otimistas para os cenários que motivam este trabalho. A primeira é de memória: o dispositivo precisaria armazenar 1.440 pontos antes de iniciar qualquer compressão. A segunda, igualmente decisiva, é de latência: um algoritmo em lote com janela de 1.440 pontos só produz e transmite um pacote comprimido a cada 1.440 amostras (um dia inteiro, na resolução de um minuto adotada neste trabalho), latência incompatível com qualquer forma de monitoramento em tempo próximo ao real, a mesma restrição já identificada para a compressão sem perda no Apêndice A. O SDT e o ARC-SDT não sofrem dessa limitação: cada *keypoint* é transmitido assim que é determinado, independentemente de o horizonte de 1.440 pontos ser usado apenas para acumular estatísticas de calibração (Seção 6.5).

Diante deste contexto, é preciso deixar clara uma diferença mais profunda entre os dois grupos, que afeta qualquer comparação feita por tamanho de janela. Para os algoritmos em lote, a janela é de fato

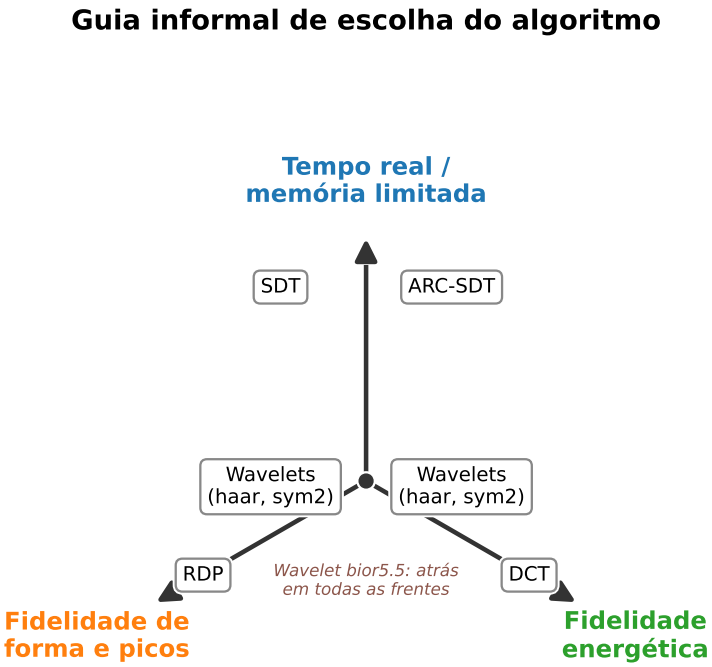
um parâmetro de projeto: define quanto contexto temporal o algoritmo enxerga e, sobretudo, quanta memória o dispositivo precisa reservar antes que a compressão sequer possa começar. Para o SDT e o ARC-SDT, a janela não é uma configuração do algoritmo. Ambos processam o sinal de forma contínua, com estado de tamanho fixo, e podem operar indefinidamente sem nunca reiniciar. As execuções com janelas menores reportadas no Capítulo 6 correspondem a reiniciar artificialmente o acumulador `ms` e o controlador a cada bloco, procedimento adotado unicamente para que todos os algoritmos compartilhassem a mesma grade experimental (Seção 6.5). Sendo assim, avaliar o ARC-SDT em uma janela de 15 pontos e compará-lo à DWT na mesma janela não descreve nenhum cenário real de implantação: impõe à proposta uma restrição que ela simplesmente não tem, e o número resultante mede o custo de um reinício que, em campo, nunca aconteceria. Por outro lado, um algoritmo em lote que precisasse operar com latência comparável à do ARC-SDT seria obrigado a reduzir sua janela para algo próximo do intervalo de transmissão desejado, e essa sim é uma restrição real, imposta pelo hardware e pelo requisito da aplicação. É exatamente essa comparação, com o ARC-SDT operando como de fato opera e os algoritmos em lote confinados às janelas que a memória permite, que revela a vantagem prática da proposta.

Concretamente, a pergunta que importa para um dispositivo real de memória limitada é quanto de contexto temporal cada algoritmo consegue efetivamente utilizar dentro do orçamento disponível. Um dispositivo capaz de armazenar apenas 15 ou 60 pontos limita a DCT, o RDP e as Wavelets a essas janelas; o ARC-SDT e o SDT, por não precisarem de *buffer*, continuam operando com o horizonte de 1.440 pontos sob a mesma restrição de memória. Nessa comparação, o resultado se inverte de forma acentuada: em CR de 80%, o ARC-SDT na janela de 1.440 pontos (SSIM de $0{,}861$) supera com folga qualquer algoritmo em lote mesmo na janela de 60 pontos (a melhor delas, o RDP, atinge SSIM de apenas $0{,}701$), e a diferença se torna drástica contra a janela de 15 pontos, onde a DCT desaba para SSIM de $0{,}096$ e PeakRecall de $0{,}000$. O mesmo se aplica ao SDT (SSIM de $0{,}896$ na janela de 1.440 pontos). Essa é, em última análise, a vantagem prática central dos algoritmos amostra a amostra avaliados neste trabalho: para um orçamento de memória fixo, eles preservam mais informação do sinal do que qualquer alternativa em lote obrigada a operar sobre uma janela igualmente pequena.

Sendo assim, a escolha entre os algoritmos depende dos requisitos da aplicação e, sobretudo, do orçamento de memória disponível no dispositivo. Para cenários com memória severamente restrita, carga útil estritamente limitada e operação em tempo real (o caso típico de LoRaWAN embarcado que motiva este trabalho), o ARC-SDT é a opção mais adequada: processa o sinal com memória constante, independentemente de quanto contexto temporal está efetivamente disponível, e oferece controle de taxa sem calibração manual. Para dispositivos com memória suficiente para armazenar janelas grandes, onde a comparação direta com os algoritmos espectrais é justa, o RDP se destaca pela preservação de picos em

taxas agressivas, e a DCT e as wavelets ortogonais pela fidelidade de forma e energética. A Figura 7.1 resume, de forma informal, essa orientação de escolha.

Figura 7.1 — Guia informal de escolha do algoritmo, segundo o requisito prioritário da aplicação. Não é uma projeção matemática das métricas; a posição de cada algoritmo reflete qualitativamente as conclusões deste capítulo.



Fonte: elaborada pelo autor.

Embora o caso de uso deste trabalho seja o monitoramento de consumo elétrico, nem a metodologia de avaliação comparativa nem o ARC-SDT dependem de alguma característica específica desse domínio. Qualquer série temporal produzida por um dispositivo embarcado com restrições de memória, banda ou energia, e que precise de controle direto sobre a taxa de compressão, enquadra-se no mesmo problema: sensores de vazão de água ou gás, monitoramento de vibração em equipamentos industriais, variáveis agrícolas (umidade do solo, temperatura) ou qualquer outra grandeza amostrada em intervalos regulares são candidatos diretos à mesma avaliação e à mesma adaptação do SDT.

7.1.1 Respostas às Perguntas de Pesquisa

Retomadas as três perguntas formuladas na Seção 1.2, os resultados apresentados nos capítulos anteriores permitem respondê-las de forma direta.

Sobre a viabilidade da compressão com perda (Pergunta 1). A viabilidade não é propriedade absoluta do algoritmo, mas a relação entre a distorção introduzida e a tolerância que a aplicação admite. Um critério de tolerância que pode ser utilizado como exemplo está na classificação de medidores de energia em índices de classe, segundo a Norma Brasileira (NBR) 14519 da Associação Brasileira de Normas Técnicas (ABNT) [8], sendo 2% o valor na classe mais permissiva. Este limite se aplica ao erro do próprio medidor e não à distorção por compressão, de modo que empregá-lo aqui é apenas uma analogia. Na métrica de erro de energia, em janela de 1.440 pontos e CR alvo de 80%, as sete variantes avaliadas mantêm erro de energia mediano abaixo de 2% nas duas coleções. Já no MAPE, critério mais restritivo por medir o erro relativo ponto a ponto, o mesmo patamar é respeitado na mediana até CR alvo de 50%, com exceção da DCT, que só o satisfaz até 20%.

Sobre a existência de um algoritmo superior (Pergunta 2). Nenhum algoritmo domina o conjunto em todas as dimensões de fidelidade, e a liderança muda conforme a métrica e a taxa de compressão alvo. O RDP é o mais robusto na preservação de picos nas taxas agressivas; a wavelet `haar` alcança a maior mediana de SSIM do conjunto avaliado; e a DCT mantém o erro de energia próximo de zero na mediana, ainda que sem garantia estrutural sob o *hard thresholding* por magnitude. Apenas a `bior5.5` fica consistentemente atrás das demais. Podemos inferir, portanto, que a escolha do algoritmo é uma decisão de projeto guiada pelo requisito dominante da aplicação, e não a adoção de um vencedor único.

Sobre o destaque em dispositivos embarcados (Pergunta 3). Aqui a resposta é afirmativa. Avaliados todos sobre a mesma janela, os algoritmos amostra a amostra não lideram o conjunto. Já com a restrição real em campo, o quadro se inverte: um dispositivo capaz de armazenar apenas 60 pontos (ou uma aplicação que precise de atualizações hora a hora) limita a DCT, o RDP e as Wavelets a essa janela, enquanto o SDT e o ARC-SDT seguem operando com horizonte de 1.440 pontos sem *buffer* algum. Nessa condição, o ARC-SDT supera em SSIM qualquer algoritmo em lote, diferença ainda mais acentuada em janelas de 15 pontos. Entre os dois algoritmos amostra a amostra, o ARC-SDT é o que responde à pergunta de forma completa, por ser o único a oferecer controle direto da taxa de compressão sem depender de calibração prévia sobre o sinal inteiro, ao custo em PeakRecall discutido nesta seção.

7.2 Trabalhos Futuros

Os resultados deste trabalho abrem caminhos concretos para investigações futuras. Algumas direções se destacam como as mais relevantes.

Validação em dispositivo embarcado real. Toda a avaliação comparativa foi conduzida em ambiente controlado de software. O próximo passo natural é implementar o ARC-SDT em um microcontrolador com recursos típicos de um medidor de campo (como o ESP32 ou o STM32) e validar o comportamento do controlador PI sob restrições reais de *clock*, memória e aritmética de ponto flutuante de precisão reduzida. É possível que o desempenho do controlador se altere em ambientes com inteiros de 16 bits ou sem unidade de ponto flutuante (FPU) dedicada, o que exigiria adaptações na implementação.

Extensão da avaliação a outras grandezas elétricas. Este trabalho utilizou séries de potência ativa como representante do sinal elétrico. Grandezas como corrente, tensão, fator de potência e THD apresentam características espectrais distintas (a THD, por exemplo, pode ter componentes de alta frequência relevantes que algoritmos geométricos tendem a descartar). Avaliar os mesmos algoritmos nessas grandezas permitiria uma recomendação mais completa sobre qual algoritmo usar para cada variável em um dispositivo de monitoramento trifásico completo.

Generalização a outros domínios de aplicação. Este trabalho e a metodologia de avaliação empregada não são específicos de séries de consumo elétrico; aplicam-se a qualquer série temporal amostrada em intervalos regulares por um dispositivo embarcado com restrições semelhantes de memória e banda. Um passo natural é replicar a avaliação comparativa, incluindo o ARC-SDT, em outros domínios, como sensoriamento ambiental, monitoramento de vibração industrial ou medição de vazão, verificando se os mesmos padrões de custo-benefício entre taxa de compressão e fidelidade se mantêm em sinais com características espectrais diferentes das grandezas elétricas estudadas aqui.

Compressão conjunta de múltiplas grandezas. A compressão foi avaliada sobre o par (*timestamp*, potência ativa), produzindo, independentemente da técnica empregada, um conjunto comprimido de tamanho específico. Ao considerar uma segunda grandeza, essa contabilidade muda: como o *timestamp* é compartilhado, a entrada cresce por um fator de 3/2 (um instante e dois valores, no lugar de um instante e um valor), enquanto a saída tende a dobrar, pois cada grandeza é comprimida de forma independente e produz seu próprio conjunto comprimido, sem que o *timestamp* possa mais ser compartilhado. Mantida a mesma taxa de compressão para cada grandeza, a taxa de compressão global deixa de coincidir com a da grandeza isolada e precisa ser ajustada a partir do desempenho individual de cada uma. Nesse ponto, técnicas baseadas em tolerância a erro, como o RDP, o SDT e o ARC-SDT, tendem a levar vantagem: aplicadas a sinais comportados, comprimem muito, de modo que grandezas de baixa variação (como a tensão) contribuem com taxas de compressão altas que compensam o crescimento da entrada. Formali-

zar essa contabilidade e verificá-la empiricamente permitiria estender ao empacotamento simultâneo de várias grandezas a recomendação obtida aqui para uma grandeza isolada.

Controlador ciente de picos no ARC-SDT. O controlador PI atual ajusta a tolerância com base exclusivamente no CR acumulado, sem qualquer consciência sobre a natureza dos pontos descartados. Uma extensão promissora seria incorporar ao controlador um detector de eventos de pico (por exemplo, baseado em derivada local ou limiar de amplitude), de modo que a tolerância seja reduzida automaticamente nas proximidades de picos detectados, protegendo-os de forma explícita em vez de deixar isso acontecer por acaso. Isso poderia reduzir o custo em PeakRecall observado neste trabalho sem comprometer o controle de taxa.

Avaliação sob diferentes políticas de transmissão LoRaWAN. Este trabalho assumiu um cenário de transmissão por leitura, com intervalo de um minuto. Em muitas implantações reais, é mais eficiente agregar múltiplas leituras em um único pacote. Seria relevante avaliar como os algoritmos se comportam com janelas correspondentes a múltiplos minutos, e como a escolha do intervalo de transmissão afeta o custo-benefício entre taxa de compressão e qualidade (especialmente para o ARC-SDT, cujo controlador PI foi projetado para operar em janelas de tamanho fixo).

Validação estatística das tendências observadas. A comparação entre algoritmos neste trabalho é descritiva, baseada em medianas e na distribuição das métricas, e identifica tendências de desempenho por dimensão de fidelidade. Um passo natural é submeter essas tendências a testes de inferência estatística, acompanhados de tamanhos de efeito e do controle de comparações múltiplas, para confirmar quais diferenças são estatisticamente distinguíveis e quais permanecem dentro da variabilidade dos dados.

Bibliografia

[1] IEEE standard definitions for the measurement of electric power quantities under sinusoidal, nonsinusoidal, balanced, or unbalanced conditions. Technical Report 1459-2010, IEEE, 2010.

[2] Agência Nacional de Energia Elétrica (ANEEL). Resolução normativa ANEEL nº 414, de 9 de setembro de 2010, 2010.

[3] Agência Nacional de Energia Elétrica (ANEEL). Procedimentos de distribuição de energia elétrica no sistema elétrico nacional (PRODIST) — módulo 8: Qualidade da energia elétrica. `https://www2.aneel.gov.br/cedoc/aren2021956_prodist_modulo_8_v13.pdf`, 2021.

[4] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, 23(1):90–93, 1974.

[5] Kiam Heong Ang, Gregory Chong, and Yun Li. PID control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, 13(4):559–576, 2005.

[6] Juan David Arias Correa, Alex Sandro Roschildt Pinto, and Carlos Montez. Training swinging door trending compression algorithm for IoT devices. In *2024 IEEE 10th World Forum on Internet of Things (WF-IoT)*, 2024.

[7] Juan David Arias Correa, Alex Sandro Roschildt Pinto, Carlos Montez, and Erico Meneses Leão. Swinging door trending compression algorithm for IoT environments. In *Anais Estendidos do IX Simpósio Brasileiro de Engenharia de Sistemas Computacionais (SBESC Estendido 2019)*, 2019.

[8] Associação Brasileira de Normas Técnicas. NBR 14519: Medidores eletrônicos de energia elétrica (estáticos) - especificação. Norma técnica brasileira, baseada na IEC 60687:1992 e IEC 61036:1990. Incorpora Errata n. 1, fev. 2005, 2000.

[9] K. J. Åström and T. Hägglund. *PID Controllers: Theory, Design, and Tuning*. Instrument Society of America, 2nd edition, 1995.

[10] Aloÿs Augustin, Jiazi Yi, Thomas Clausen, and William Mark Townsley. A study of LoRa: Long range & low power networks for the Internet of Things. *Sensors*, 16(9):1466, 2016.

[11] Sean Barker, Aditya Mishra, David Irwin, Emmanuel Cecchet, Prashant Shenoy, and Jacobus Albrecht. Smart*: An open data set and tools for enabling research in sustainable homes. In *Proceedings of the Workshop on Data Mining Applications in Sustainability (SustKDD)*, 2012.

[12] Nipun Batra, Manoj Gulati, Amarjeet Singh, and Mani B. Srivastava. It's different: Insights into home energy consumption in India. In *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings (BuildSys)*, 2013.

[13] Monika Borova, Michal Prauzek, Jaromir Konecny, and Karolina Gaiova. A performance analysis of edge computing compression methods for environmental monitoring nodes with LoRaWAN communications. In *IFAC-PapersOnLine*, volume 55, pages 387–392, 2022.

[14] E. H. Bristol. Swinging door trending: Adaptive trend recording. In *Proceedings of the ISA National Conference*, pages 749–753, 1990.

[15] Edgar H. Bristol. Data compression for display and storage, maio 1987.

[16] Gang Chen and Li Li. An optimized algorithm for lossy compression of real-time data. In *2010 IEEE International Conference on Intelligent Computing and Intelligent Systems (ICICISYS)*, pages 187–191, 2010.

[17] Giacomo Chiarot and Claudio Silvestri. Time series compression survey. *ACM Computing Surveys*, 55(10):1–32, 2023.

[18] J. C. Clegg. A nonlinear integrator for servomechanisms. *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry*, 77(1):41–42, 1958.

[19] Arnaud de Myttenaere, Boris Golden, Bénédicte Le Grand, and Fabrice Rossi. Mean absolute percentage error for regression models. *Neurocomputing*, 192:38–48, 2016.

[20] Chacko John Deepu, Chun-Huat Heng, and Yong Lian. A hybrid data compression scheme for power reduction in wireless sensors for IoT. *IEEE Transactions on Biomedical Circuits and Systems*, 11(2):245–254, 2017.

[21] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[22] Jens Engel, Andrea Castellani, Patricia Wollstadt, Felix Lanfermann, Thomas Schmitt, Sebastian Schmitt, Lydia Fischer, Steffen Limmer, David Luttropp, Florian Jomrich, Rene Unger, and Tobias Rodemann. A real-world energy management dataset from a smart company building for optimization and machine learning, 2025.

[23] ETH Zurich. ECO data: Electricity consumption and occupancy dataset. ETH Zurich Research Collection, 2014.

[24] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid—the new and improved power grid: A survey. *IEEE Communications Surveys & Tutorials*, 14(4):944–980, 2012.

[25] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, 2020.

[26] IAWE. Indian appliance level energy dataset. `http://iawe.github.io/`, 2014.

[27] IoT Analytics. LPWAN market 2024: Licensed technologies boost their share among global 1.3 billion connections as LoRa leads outside China. `https://iot-analytics.com/lpwan-market/`, 2024.

[28] IoT Analytics. State of IoT spring 2024: 10 emerging IoT trends driving market growth. `https://iot-analytics.com/state-of-iot-10-emerging-iot-trends-driving-market-growth/`, 2024.

[29] Jack Kelly and William Knottenbelt. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Scientific Data*, 2:150007, 2015.

[30] Manel Khadiche, Bachir Boucheham, and Salah Bougueroua. Ramer-Douglas-Peucker dynamic time warping (RDP-DTW): A novel data reduction based dynamic time warping method for time series classification. In *2024 International Conference on Advances in Electrical and Communication Technologies (ICAECOT)*, pages 1–5, 2024.

[31] Jaromir Konecny, Monika Borova, and Michal Prauzek. Self-learning wavelet compression method for data transmission from environmental monitoring stations with a low bandwidth IoT interface. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, 2021.

[32] Gregory R. Lee, Ralf Gommers, Filip Waselewski, Kai Wohlfahrt, and Aaron O'Leary. PyWavelets: A Python package for wavelet analysis. *Journal of Open Source Software*, 4(36):1237, 2019.

[33] Yan-hua Liu and Wei-qing Chen. Line simplification algorithm implementation and error analysis. In *2011 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, pages 64–68, 2011.

[34] LoRa Alliance. LoRaWAN 1.1 specification. Technical specification, LoRa Alliance, 2017.

[35] LoRa Alliance. LoRaWAN regional parameters RP002-1.0.5. Technical specification, LoRa Alliance, 2022.

[36] Stephen Makonin, Z. Jane Wang, and Craig Tumpach. RAE: The rainforest automation energy dataset for smart grid meter data analysis. *Data*, 3(1):8, 2018.

[37] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[38] Wes McKinney. Data structures for statistical computing in Python. In Stéfan van der Walt and K. Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56–61, 2010.

[39] Andrea Monacchi, Dominik Egarter, Wilfried Elmenreich, Salvatore D'Alessandro, and Andrea M. Tonello. GREEND: An energy consumption dataset of households in Italy and Austria. In *Proceedings of the 5th IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Venice, Italy, 2014.

[40] David Murray, Lina Stankovic, and Vladimir Stankovic. An electrical load measurements dataset of United Kingdom households from a two-year monitoring campaign. *Scientific Data*, 4:160122, 2017.

[41] National Instruments. The PID controller & theory explained. `https://www.ni.com/en/shop/labview/pid-theory-explained.html`, 2025.

[42] Marco Pasetti, Emiliano Sisinni, Paolo Ferrari, Paolo Bellagente, and Dario Zaninelli. Comprehensive evaluation of lossless compression algorithms in a real use case for smart grid applications. *Sustainable Energy, Grids and Networks*, 36:101238, 2023.

[43] U. Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244–256, 1972.

[44] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. Low power wide area networks: An overview. *IEEE Communications Surveys & Tutorials*, 19(2):855–873, 2017.

[45] Przemyslaw Skibinski. lzbench: In-memory benchmark of open-source LZ77/LZSS/LZMA compressors. https://github.com/inikep/lzbench, 2023.

[46] Stéfan J. van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: Image processing in Python. *PeerJ*, 2:e453, 2014.

[47] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, 2020.

[48] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[49] Weatherford. CygNet swinging door compression. https://softwaredocs.weatherford.com/cygnet/94/Content/Topics/History/CygNet%20Swinging%20Door%20Compression.htm, 2024.

[50] Denghui Yao, Xiaoyong Zhang, and Zhengbo Sun. Research on application of data compression in signal correlation processing. In *2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT)*, 2021.

[51] W. A. Yasodya, S. M. L. Arampola, M. S. K. Nisakya, V. Logeeshan, S. Kumarawadu, and Chathura Wanigasekara. Self-adaptive deep learning framework for non-intrusive load monitoring: Addressing aging appliance challenges with transfer learning and pseudo labeling. *IEEE Access*, 13:106524–106539, 2025.

Apêndice A

Avaliação de Algoritmos de Compressão sem Perda

Como etapa preliminar desta pesquisa, foi realizado um experimento de compressão sem perda aplicado a dados de grandezas elétricas coletados no Laboratório de Sistemas Distribuídos (LSD) da Universidade Federal de Campina Grande (UFCG). O objetivo foi reproduzir a metodologia de Pasetti et al. [42] em um conjunto de dados diferente, verificar se os resultados se generalizam e quantificar até onde a compressão sem perda consegue reduzir o volume de dados de monitoramento elétrico com amostragem de 1 minuto. Os resultados deste experimento são parte da proposta de qualificação de mestrado que precedeu esta dissertação e são apresentados aqui como motivação para a investigação de algoritmos com perda desenvolvida nos capítulos anteriores.

A.1 Configuração Experimental

A grandeza utilizada foi a potência ativa total do sistema trifásico, registrada com intervalo de amostragem de 1 segundo e agregada em 1 minuto pelo medidor inteligente *LiteMe*, instalado no LSD. O período de coleta foi de 12 a 17 de maio de 2024, totalizando seis dias completos com 8.640 amostras por série. A ausência de períodos com potência nula ou constante (característica da sala de servidores em funcionamento ininterrupto do LSD) conferiu um comportamento estocástico aos dados, resultando em razões de compressão mais uniformes ao longo do dia em comparação com os dados do artigo original.

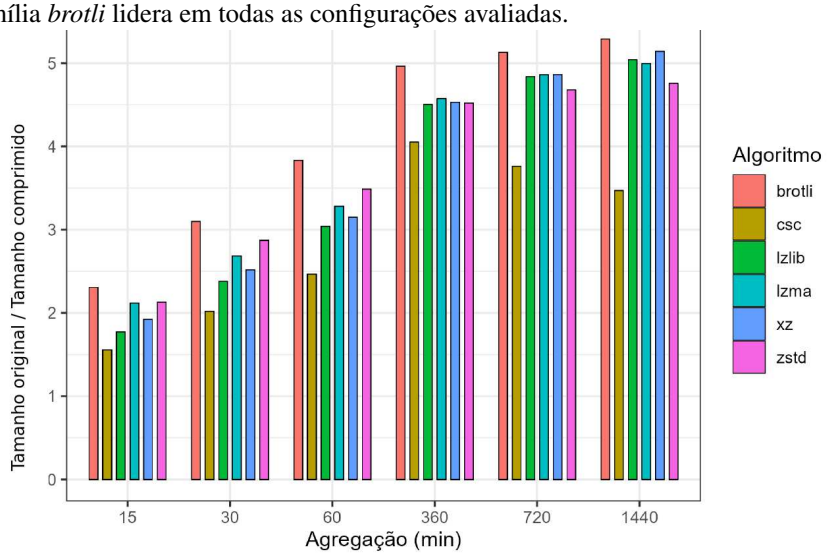
Para avaliar o impacto da escala temporal sobre a compressibilidade, os dados foram particionados em arquivos CSV com diferentes intervalos de agregação $\tau \in \{15, 30, 60, 360, 720, 1440\}$ minutos. Cada arquivo foi submetido a 56 algoritmos de compressão disponíveis na ferramenta *lzbench* [45], que

executa compressores baseados nas estratégias LZ77/LZSS/LZMA com o mesmo compilador e otimizações, garantindo uma comparação justa entre as técnicas. A métrica adotada é a razão entre o tamanho original e o tamanho comprimido do arquivo: quanto maior essa razão, maior a redução obtida. Nota-se que essa definição difere da métrica homônima utilizada nos capítulos desta dissertação, que expressa a redução como porcentagem do sinal original.

A.2 Resultados dos Algoritmos

Dentre os 56 algoritmos avaliados, seis famílias apresentaram desempenho notavelmente superior aos demais: *brotli* (e variantes brotli22, brotli24), *csc*, *lzlib*, *lzma*, *xz* e *zstd* (e variantes zstd22, zstd24). A Figura A.1 apresenta a razão de compressão total para esses algoritmos em cada nível de agregação.

Figura A.1 — Razão de compressão total para os algoritmos sem perda com melhor desempenho, por intervalo de agregação τ. A família *brotli* lidera em todas as configurações avaliadas.



Fonte: elaborada pelo autor.

É possível observar uma tendência consistente de crescimento da razão de compressão com o aumento de τ. Para $\tau = 15$ minutos, o *brotli* obtém razão de 2,3 entre tamanho original e comprimido, equivalente a $CR = 56{,}5\%$ na definição adotada nesta dissertação; para $\tau = 1440$ minutos (um dia completo), essa razão chega a 5,3, equivalente a $CR = 81{,}1\%$. Sendo assim, percebemos que a agregação temporal aumenta substancialmente a compressibilidade: janelas maiores expõem mais regularidades ao compressor, especialmente as semelhanças entre leituras consecutivas. Outro ponto de destaque é o comportamento do *csc*: ao contrário dos demais algoritmos, seu desempenho diminui a partir de $\tau = 360$ minutos, revelando um intervalo ótimo de agregação específico para esta técnica. Por fim, o custo com-

putacional de todos os algoritmos mostrou-se desprezível frente ao intervalo de coleta: o maior tempo observado foi de 325 ms para o *xz* com $\tau = 15$ min, correspondendo a apenas 0,036% do intervalo de amostragem.

A.3 Comparação com os Limites do LoRaWAN

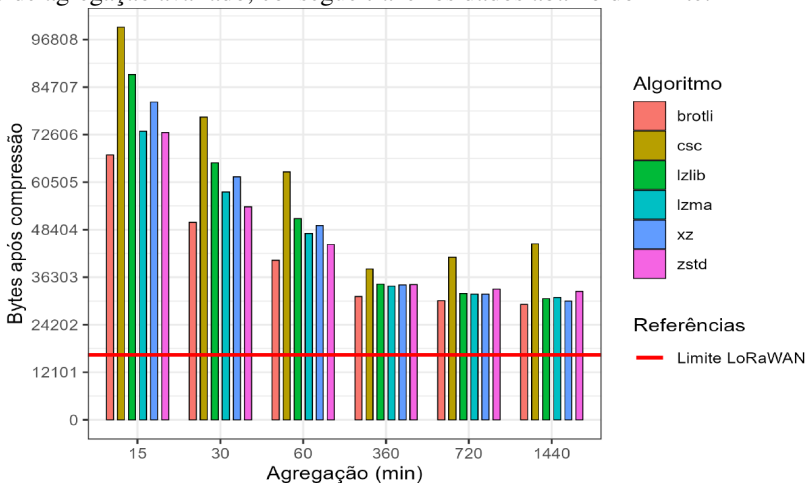
Para avaliar se os algoritmos sem perda são suficientes para viabilizar a transmissão por LoRaWAN, essas razões de compressão foram aplicadas a três configurações de carga útil por leitura, considerando coleta a cada minuto:

- **Configuração I**: todas as grandezas do sistema de monitoramento (potências ativa e reativa por fase e total, fator de potência por fase e total, tensões e correntes por fase, neutro, THD de tensão e corrente, frequência): $l(I) = 120$ bytes por leitura, totalizando $120 \times 60 = 7200$ bytes/hora e 172.800 bytes/dia;
- **Configuração II**: apenas as grandezas relacionadas a potência (ativa, reativa e fator de potência por fase e total): $l(II) = 56$ bytes por leitura, totalizando 3360 bytes/hora e 80.640 bytes/dia;
- **Configuração III**: configuração mínima com as cinco grandezas de maior interesse imediato (potência ativa total, potência reativa total, fator de potência trifásico, tensão mínima e corrente máxima): $l(III) = 28$ bytes por leitura, totalizando 1680 bytes/hora e 40.320 bytes/dia.

Como referência de limite LoRaWAN, foi adotado um plano público com *DataRate* = 3 (configuração intermediária, representativa de ambientes urbanos com interferência eletromagnética moderada), que permite carga útil de 115 bytes por mensagem e no máximo 6 mensagens por hora (144 por dia), totalizando $115 \times 144 = 16.560$ bytes por dia. As Figuras A.2, A.3 e A.4 apresentam o volume diário de dados após compressão para cada configuração, em comparação com esse limite (linha vermelha).

Diante deste contexto, os resultados são conclusivos: a Configuração I permanece acima do limite em todos os cenários, mesmo com *brotli* e $\tau = 1440$ minutos (o resultado é aproximadamente 30.000 bytes por dia, quase o dobro do limite). A Configuração II apresenta comportamento similar. Somente a Configuração III, com o melhor algoritmo e agregação de no mínimo 6 horas, consegue ficar abaixo do limite para *DataRate* = 3, mas com um custo operacional elevado: transmitir dados agrupados em intervalos de 6 horas torna inviável qualquer forma de monitoramento em tempo próximo ao real. Em ambientes com menor cobertura, onde *DataRate* < 3 e a carga útil cai para 51 bytes, mesmo a Configuração III com os melhores algoritmos e maior agregação não satisfaz o limite.

Figura A.2 — Bytes diários após compressão sem perda para a Configuração I (todas as grandezas de monitoramento). A linha vermelha indica o limite diário de um plano LoRaWAN com *DataRate* = 3. Nenhum algoritmo, em nenhum intervalo de agregação avaliado, consegue trazer os dados abaixo do limite.



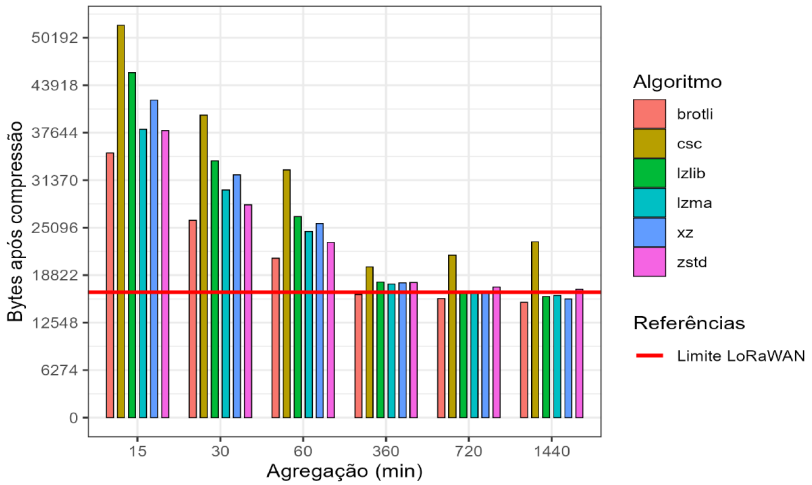
Fonte: elaborada pelo autor.

A.4 Conclusão

Os resultados estabelecem o limite superior da compressão sem perda para este caso de uso: a razão entre tamanho original e comprimido fica entre 2 e 5, dependendo do algoritmo e do intervalo de agregação, o que corresponde a CR entre 50% e 80% na definição adotada nesta dissertação. Para os cenários mais relevantes na prática (transmissão a cada 15 a 60 minutos com o conjunto completo de grandezas), essa faixa de redução está longe de ser suficiente, e a lacuna entre o volume gerado e o limite da rede exige uma redução muito maior do que as técnicas sem perda conseguem atingir.

Existe, porém, um caso em que a compressão sem perda consegue cruzar a barreira do LoRaWAN: a Configuração III com *brotli* e intervalos de agregação de 6 horas ou mais. Ainda assim, esse caso de exceção carrega três ressalvas que limitam sua utilidade prática. A primeira diz respeito ao comportamento do sinal: a compressibilidade depende das características estatísticas da grandeza monitorada, e séries com alta variabilidade estocástica (como as da sala de servidores do LSD) tendem a se comprimir menos do que sinais com padrões mais regulares; não há garantia de que a razão de compressão observada se replique em outras instalações ou condições de operação. A segunda ressalva é o custo de memória: para comprimir uma janela de 360 minutos na Configuração III, o dispositivo precisa manter em RAM um *buffer* de $360 \times 28 = 10.080$ bytes antes de iniciar a compressão, além da memória interna do próprio algoritmo; em hardware embarcado com dezenas ou centenas de kilobytes disponíveis, esse requisito pode ser proibitivo. A terceira é o impacto no monitoramento em tempo real: transmitir dados agrupados em janelas de 6 horas significa que variações de tensão, sobrecargas ou picos de demanda seriam detectados

Figura A.3 — Bytes diários após compressão sem perda para a Configuração II (grandezas de potência). Nenhum algoritmo consegue ficar abaixo do limite LoRaWAN em nenhum dos intervalos avaliados.

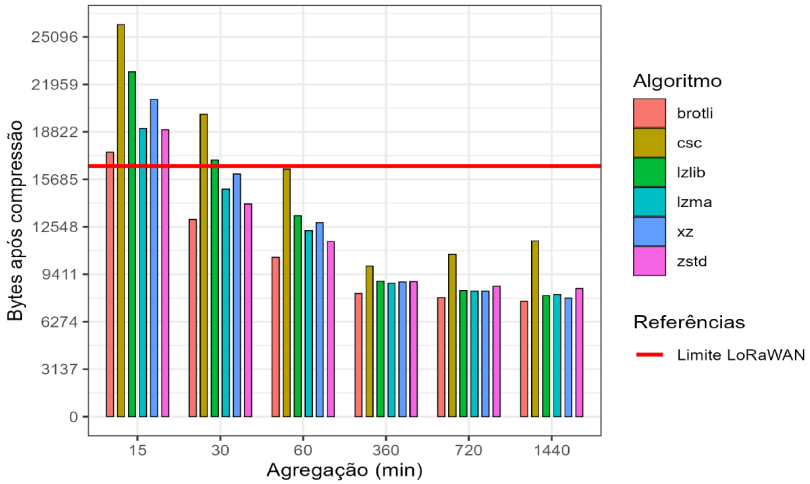


Fonte: elaborada pelo autor.

com horas de atraso, tornando o monitoramento operacional inviável. Aliadas a essas três restrições, há ainda a questão da cobertura do cenário: o resultado satisfatório ocorre apenas para a configuração mínima de grandezas e apenas nos maiores intervalos de agregação avaliados. Qualquer ampliação do conjunto monitorado leva a Configuração III de volta para além do limite.

Será que a compressão com perdas vai conseguir mudar esse cenário? É exatamente essa questão que motivou a avaliação comparativa de algoritmos com perda desenvolvida no corpo principal desta dissertação.

Figura A.4 — Bytes diários após compressão sem perda para a Configuração III (cinco grandezas mínimas). O *brotli* fica abaixo do limite apenas a partir de $\tau = 360$ minutos, exigindo agregação em janelas de 6 horas ou mais.



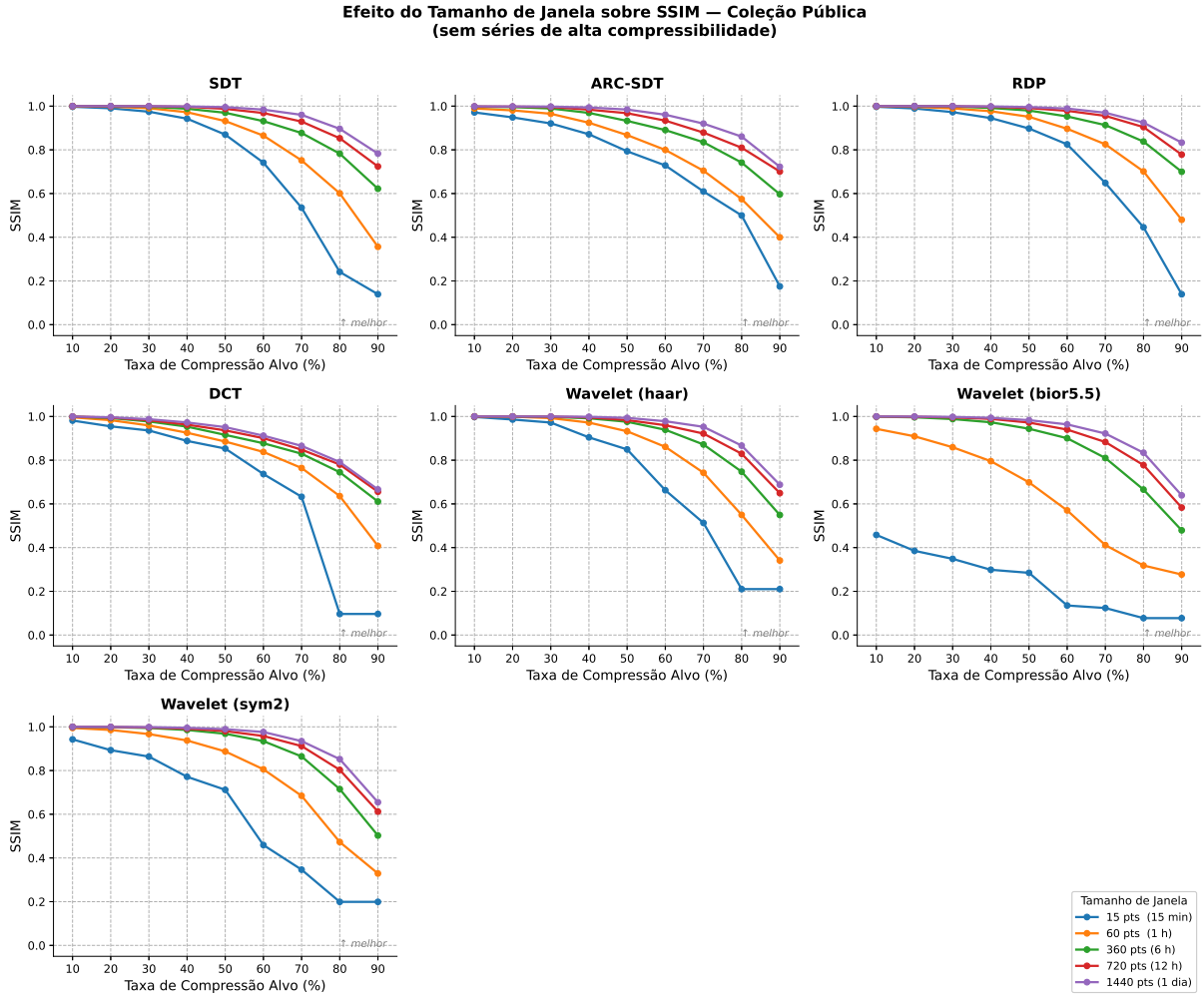
Fonte: elaborada pelo autor.

Apêndice B

Efeito do Tamanho de Janela por Algoritmo e Métrica

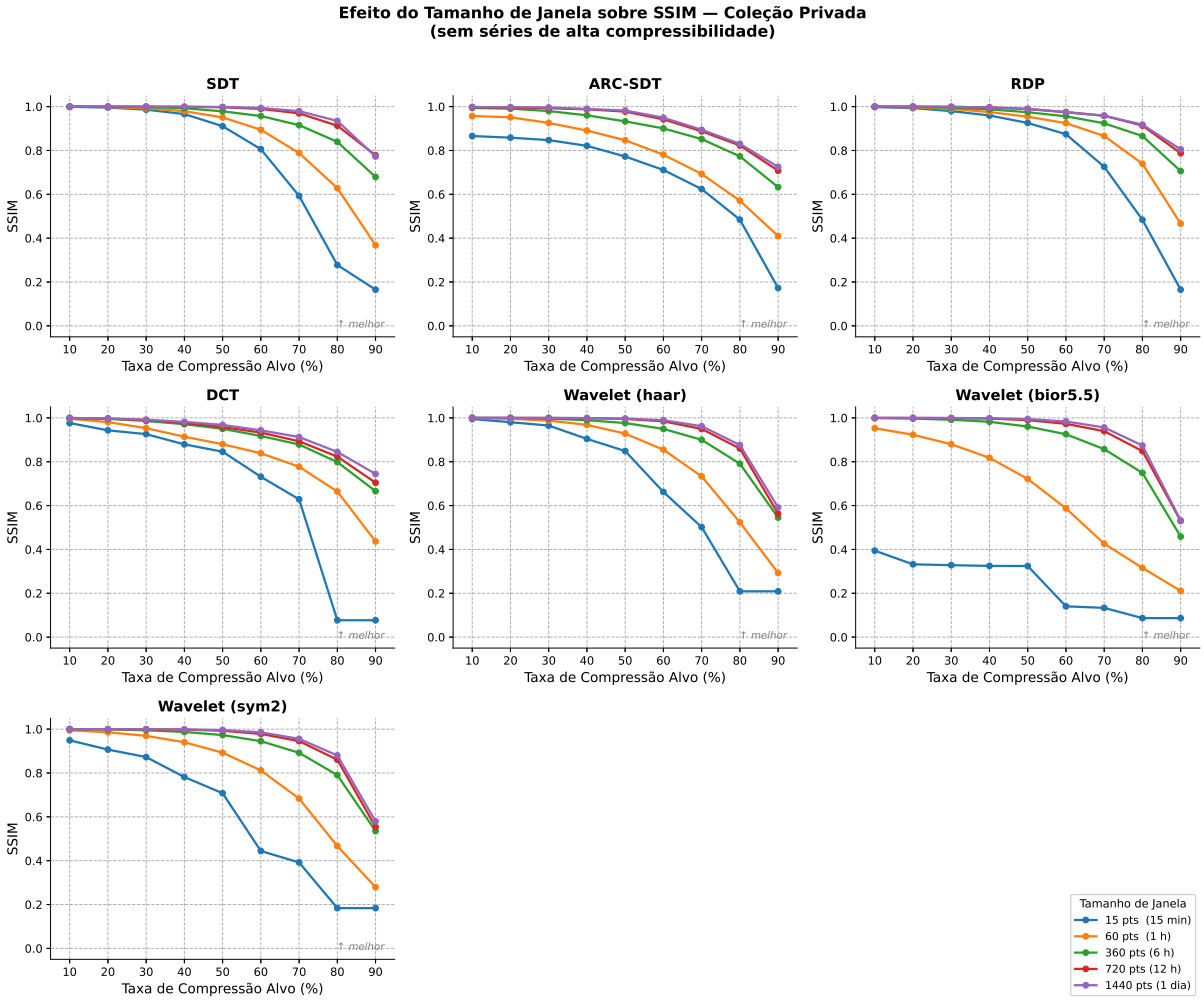
Este apêndice apresenta o detalhamento por algoritmo do efeito do tamanho de janela sobre as quatro métricas avaliadas. Cada figura contém um subplot por algoritmo, mostrando a mediana da métrica em função do CR alvo para cada tamanho de janela. A visão condensada, com todos os algoritmos em um único gráfico, é apresentada na Seção 6.5 do capítulo de avaliação.

Figura B.1 — Efeito do tamanho de janela sobre o SSIM mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade).



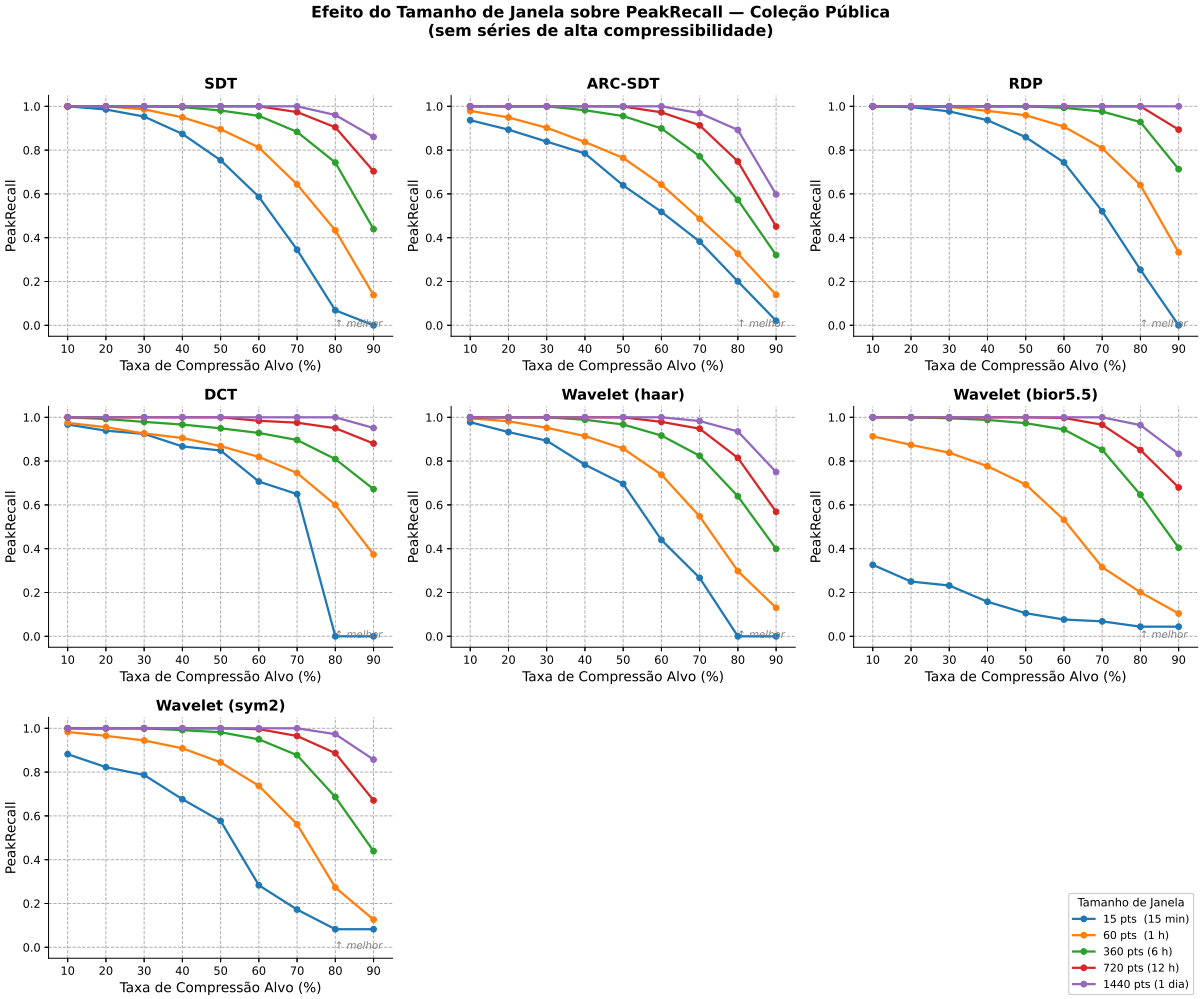
Fonte: elaborada pelo autor.

Figura B.2 — Efeito do tamanho de janela sobre o SSIM mediano por algoritmo: Coleção Privada.



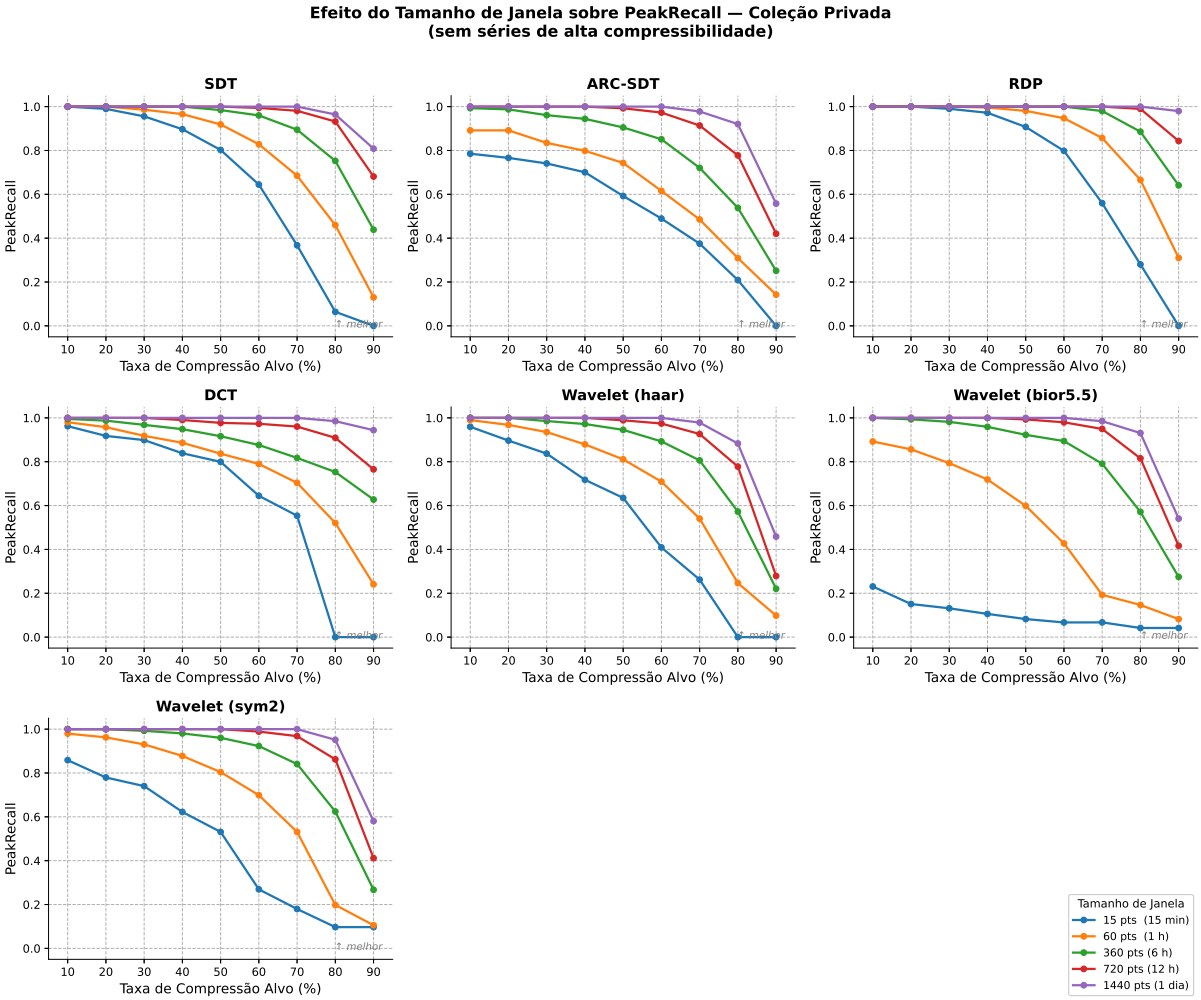
Fonte: elaborada pelo autor.

Figura B.3 — Efeito do tamanho de janela sobre o PeakRecall mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade).



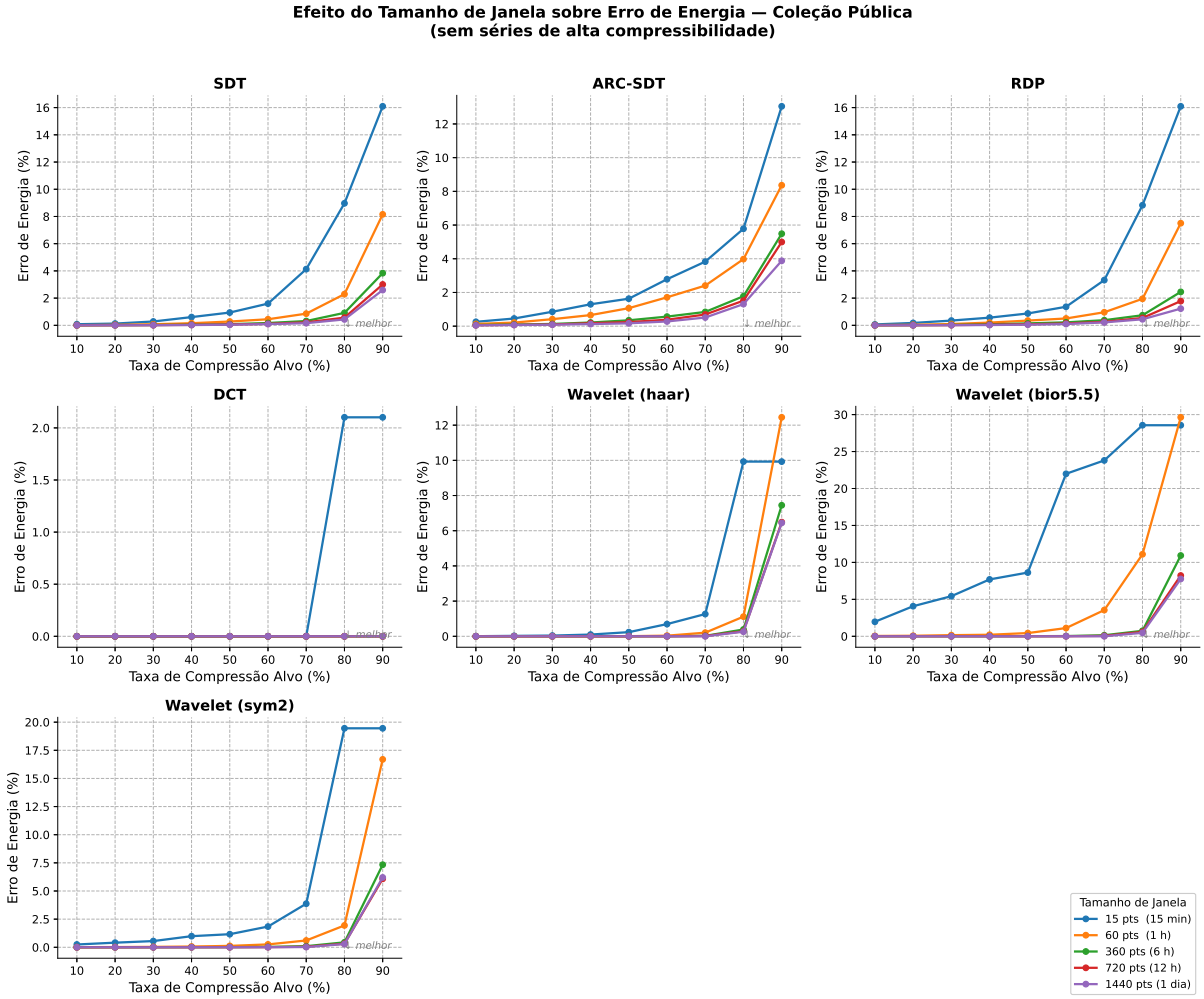
Fonte: elaborada pelo autor.

Figura B.4 — Efeito do tamanho de janela sobre o PeakRecall mediano por algoritmo: Coleção Privada.



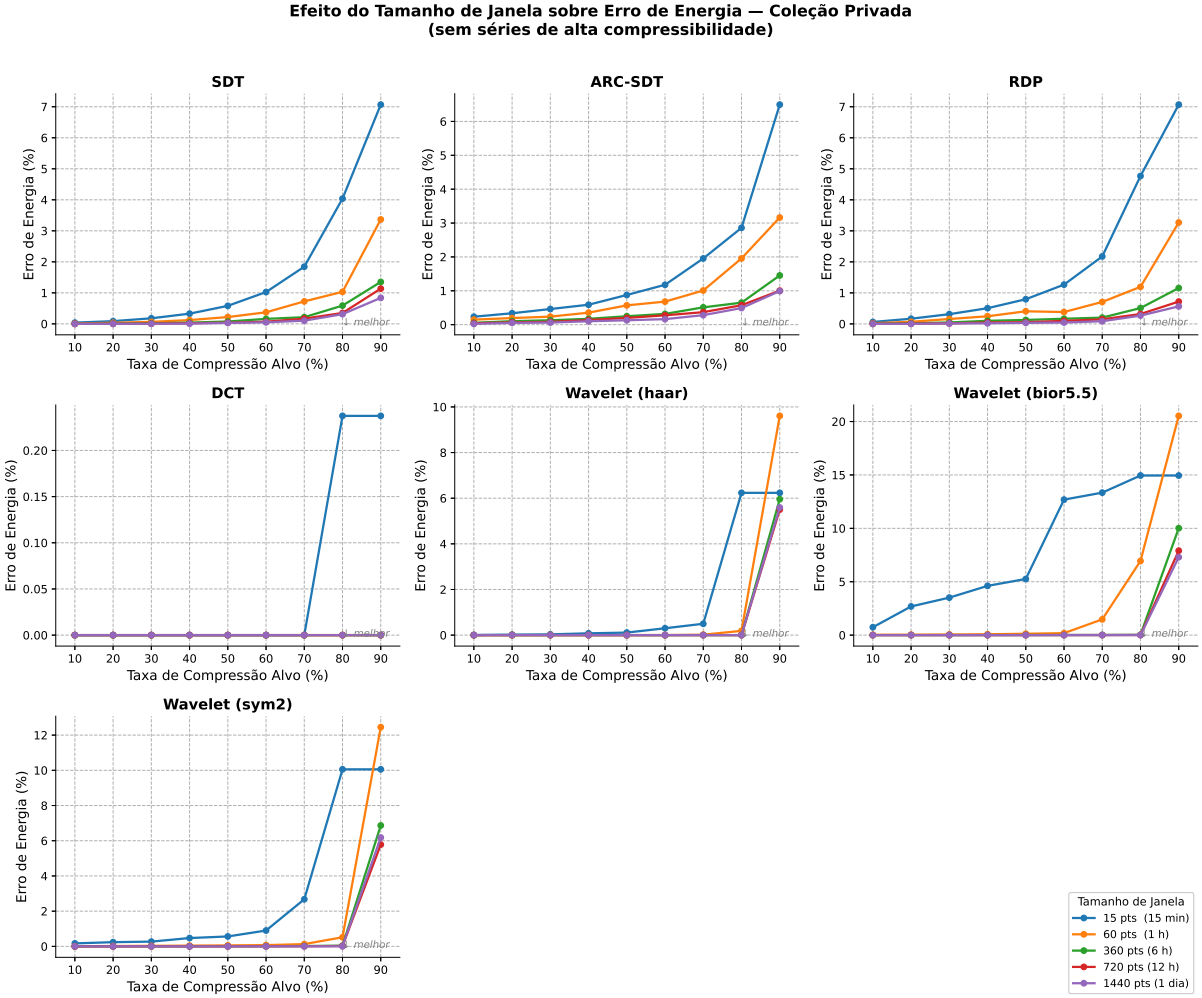
Fonte: elaborada pelo autor.

Figura B.5 — Efeito do tamanho de janela sobre o Erro de Energia mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade).



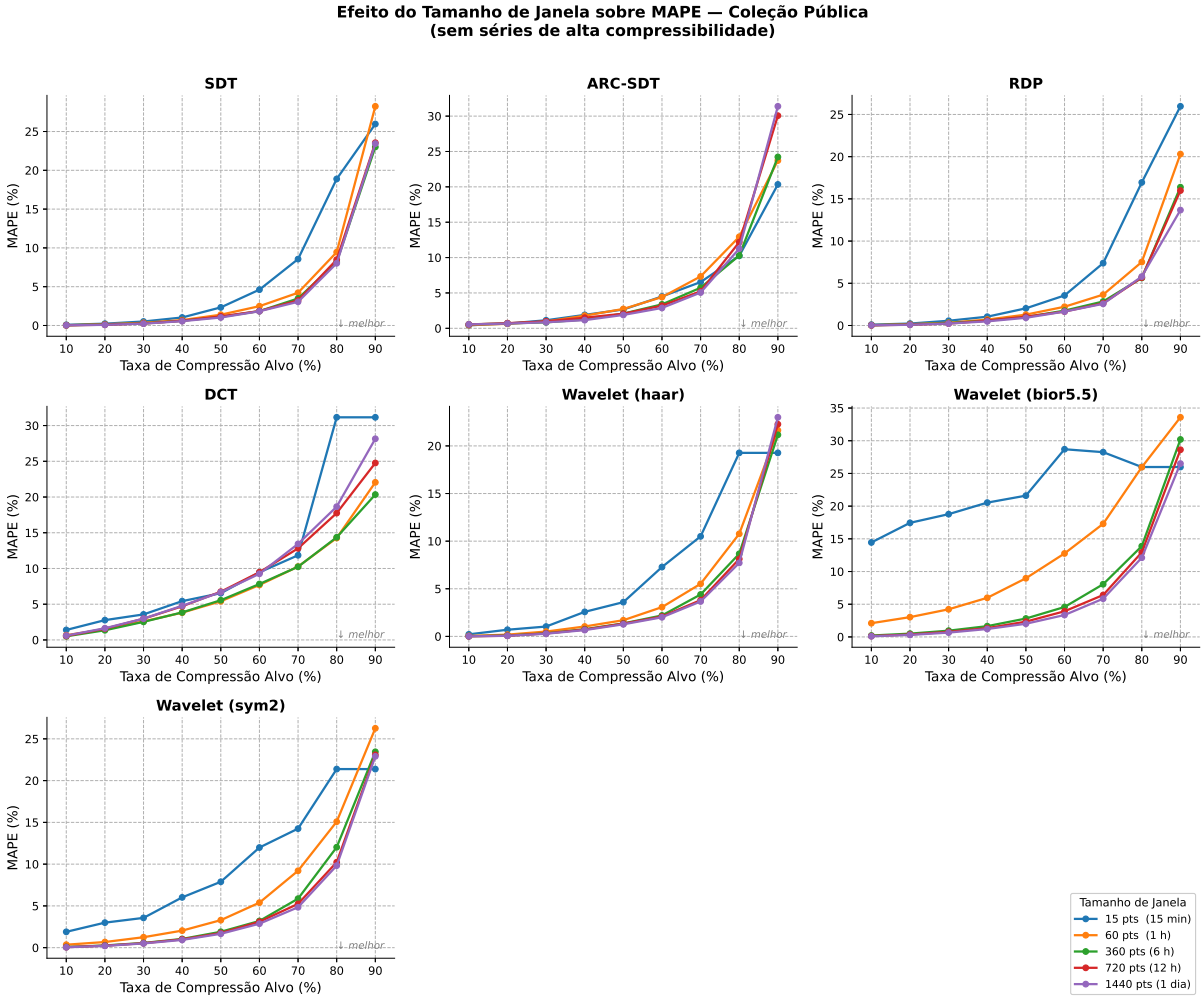
Fonte: elaborada pelo autor.

Figura B.6 — Efeito do tamanho de janela sobre o Erro de Energia mediano por algoritmo: Coleção Privada.



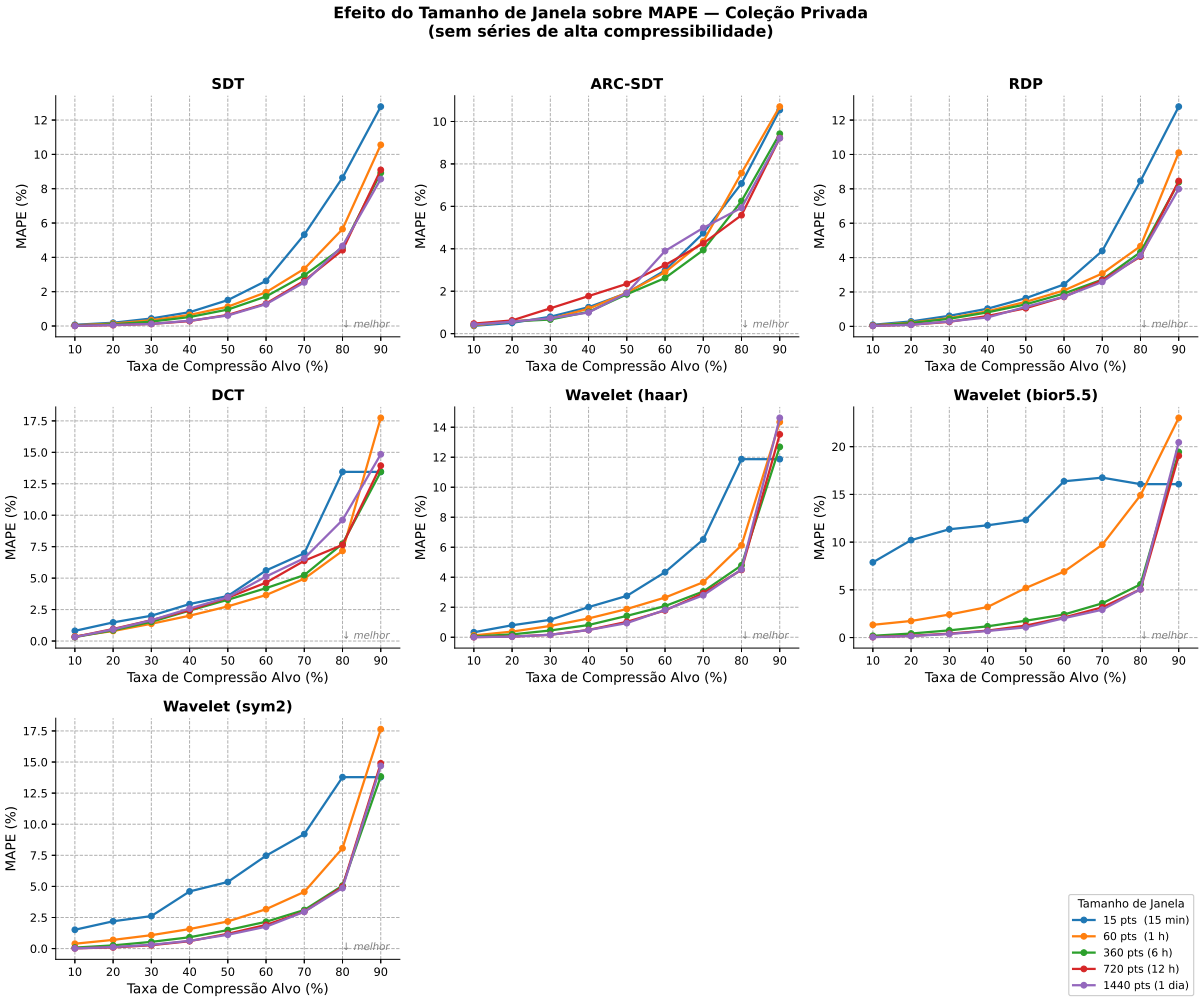
Fonte: elaborada pelo autor.

Figura B.7 — Efeito do tamanho de janela sobre o MAPE mediano por algoritmo: Coleção Pública (excluindo as séries de alta compressibilidade).



Fonte: elaborada pelo autor.

Figura B.8 — Efeito do tamanho de janela sobre o MAPE mediano por algoritmo: Coleção Privada.



Fonte: elaborada pelo autor.

Apêndice C

Compressibilidade da Tensão

Como discutido na introdução, a tensão foi apontada como um problema semelhante ao da potência ativa, porém mais simples, com sua análise sistemática deixada como trabalho futuro. Ainda assim, é possível oferecer uma demonstração inicial de que a tensão é, de fato, altamente compressível. Este apêndice aplica o SDT às séries de tensão das instalações da coleção privada e observa a taxa de compressão obtida.

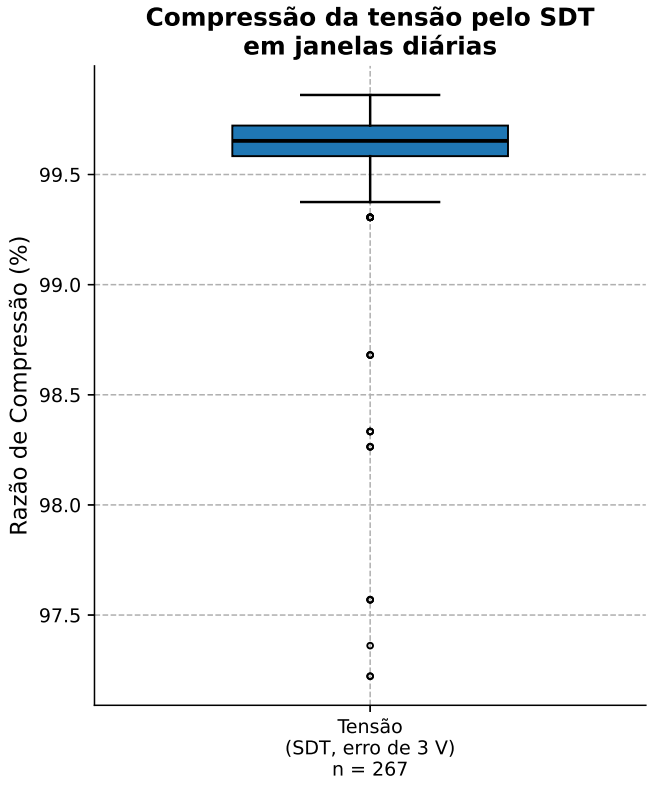
O experimento utiliza os quinze conjuntos de dados brutos da coleção privada, cada um com as três tensões de fase amostradas a cada minuto. A tolerância de erro do SDT foi fixada em 3 V. A faixa adequada de tensão em regime permanente definida pela ANEEL no Módulo 8 dos Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional (PRODIST) [3] vai de 202 a 231 V para a tensão nominal de 220 V, e de 117 a 133 V para 127 V; a menor de todas essas margens é de 6 V, entre 127 e 133 V. Adotou-se metade dessa menor margem, 3 V, como critério deliberadamente rigoroso, aplicado igualmente a todos os níveis nominais presentes nos dados. A compressão foi aplicada apenas a dias completos, de 1.440 leituras, o que resultou em 267 janelas diárias somando as três fases.

O resultado é contundente. A Figura C.1 mostra que a taxa de compressão se concentra em torno de 99,65%, com as 267 janelas entre 97,22% e 99,86%. Uma taxa de 99,65% corresponde a reter cerca de cinco pontos para representar um dia inteiro de 1.440 leituras: bastam poucos segmentos de reta, dentro da tolerância adotada, para descrever a tensão ao longo de vinte e quatro horas. É uma compressão muito acima dos 75% exigidos no cenário mais restritivo (Tabela 2.2) e muito acima do que a potência ativa alcança nas mesmas condições.

Mesmo o pior caso confirma essa leitura. A Figura C.2 apresenta a janela de menor compressão de todo o experimento, uma fase com razão de 97,22%: mesmo diante de uma tensão ruidosa e com quedas acentuadas ao longo do dia, o SDT a reproduz com apenas quarenta pontos, ante as 1.440 leituras

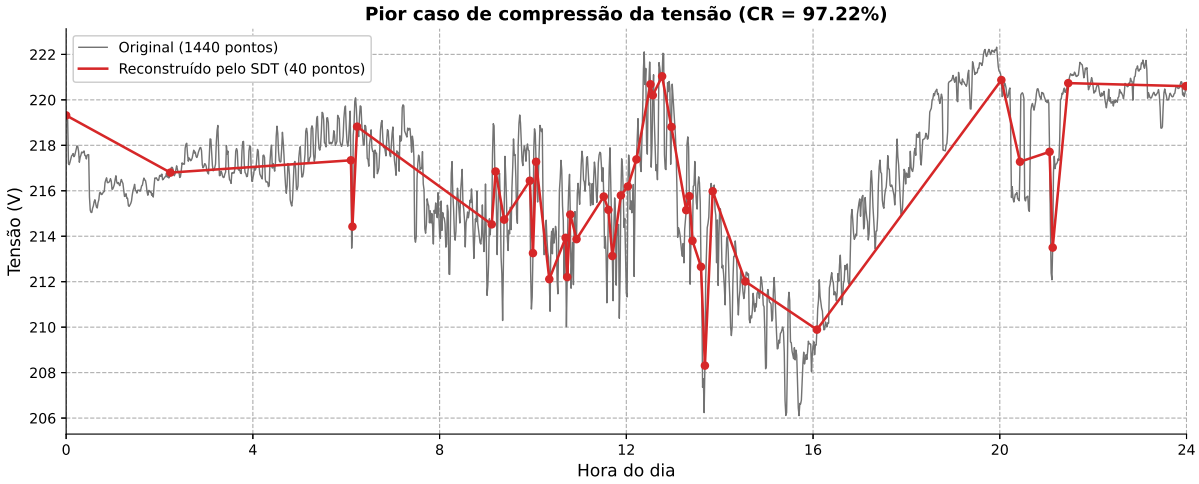
originais. A ressalva feita na Metodologia, de que a tensão é comprimida com folga por praticamente qualquer técnica, encontra aqui sua confirmação empírica.

Figura C.1 — Taxa de compressão do SDT aplicado à tensão, com tolerância de 3 V, sobre 267 janelas diárias das três fases de quinze instalações. A distribuição concentra-se em torno de 99,65%.



Fonte: elaborada pelo autor.

Figura C.2 — Pior caso de compressão do experimento, com razão de 97,22%: a série original de um dia completo (1.440 pontos) e sua reconstrução pelo SDT com quarenta pontos.



Fonte: elaborada pelo autor.